

AI Low Code Platform

A12 Backstage Session - September 2026

Sneak Peek into selected development streams

openCode 

GitHub 

Basic information about A12

<https://www.a12.ai>

What is A12?

A12 is an open source platform for developing enterprise applications in complex IT landscapes. It relies on model-driven software engineering (MDSE) and brings the low-code principle to the world of enterprise software. As an open platform, A12 simplifies the integration of best-of-breed solutions and the use of AI at all levels. A12's Modeling Environment provides tools to

create parts of an application without programming knowledge and to maintain them over the long term as independent business logic modules. A12's Runtime Platform provides the flexibility needed to evolve business-critical applications with professional individual software development, AI support and system integration into fully integrated enterprise applications.

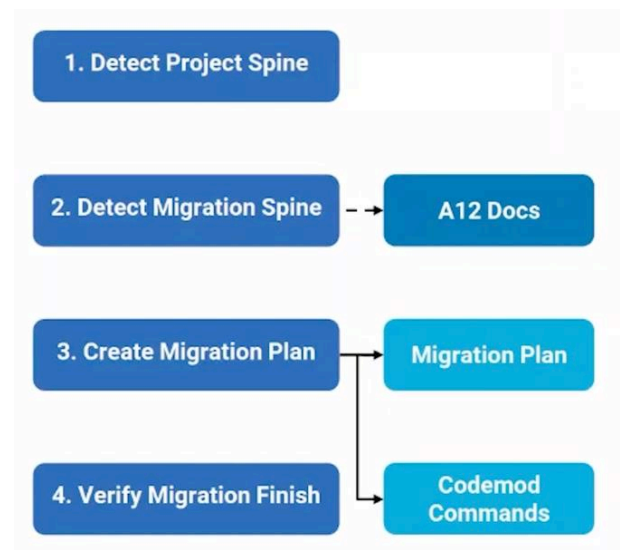
CLAUDE

OPENCODE

Planned for
2026.06-ext2

Agentic Engineering – Improved Migration Support, Plugins for OpenCode, and More

The tooling available for working with AI agents on A12 projects continues to grow. Migration support, for example, has been expanded. The Claude Code migration plugin, available since July, no longer helps only with moves to new major releases — drawing up a migration plan, generating codemod commands, and checking the migration for completeness. It now also supports minor release migrations, such as from version 2026.06-ext0 to 2026.06-ext1. In smaller projects, migrations have already been completed fully autonomously. In addition to the existing plugins for Claude Code, equivalent plugins for OpenCode will be provided going forward. This makes it possible to use self-hosted LLMs for AI agents, for instance — a key prerequisite for digitally sovereign development processes. Alongside the complete A12 documentation, the A12 Discourse forum has now been integrated into a12-knowledge-support as well, giving AI agents access to the community's entire body of knowledge. Since the first plugins were released, mgm has been developing accompanying benchmarks to assess the time, quality, and cost of AI-assisted tasks. In addition to the already extensive benchmarks for modeling tasks, benchmarks for further areas will follow — for example, around application architecture. New benchmarks are driven in particular by user feedback.



Business Scripting – First Experimental Version with Full-Stack Support

With Business Scripting, domain experts and business analysts gain access to additional A12 functionality beyond modeling – in the form of small TypeScript scripts with a fixed structure and a purpose-built API. A first version supporting backend and workflow operations has been available internally at mgm since June as part of research and development. With the A12 release 2026.06-ext2, this internal development version is being extended to the frontend. Scripts can now define actions for

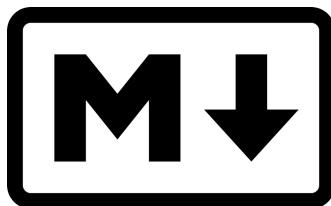
a button click, for example, meaning Business Scripting covers the entire stack for the first time. Backend support for Business Scripting is also becoming part of the regular 2026.06-ext2 release. It is available as an experimental feature to anyone who would like to try it out and share feedback. The experimental full-stack version is expected to become part of release 2026.06-ext4 in early 2027.

```
1 import type { Script, ScriptContext } from "@com.mgmt.a12.client/client-scripting";
2 import type { ScriptClientApi } from "@com.mgmt.a12.client/client-scripting/client-api";
3
4 type Args = { contractId: string };
5
6 export const openContractDetail: Script<Args, string> = {
7   name: "openContractDetail",
8   description: "Opens the selected contract in its own detail form.",
9
10  async main({ api, args }: ScriptContext<ScriptClientApi, Args>) {
11    const activity = await api.activity.launcher();
12    if (!activity) return "no activity to open from";
13
14    const result = await api.activity.startSub(
15      activity.id,
16      { model: "Contract_DM", instance: args.contractId },
17      { views: [{ name: "ContractDetailForm" }] },
18    );
19
20    return result.outcome;
21  },
22 };
```

Markdown Support: Rich Text for Humans and AI Agents

A12 is introducing end-to-end Markdown support, bringing a single, consistent way to work with formatted text across the platform. For users, this makes it easier to edit and review information in A12 applications in a structured, condensed form. And since LLMs handle Markdown well, it also opens up new possibilities for AI agents to read from and write to fields. Alongside two new widgets – the Core Markdown Widget and the Mark-

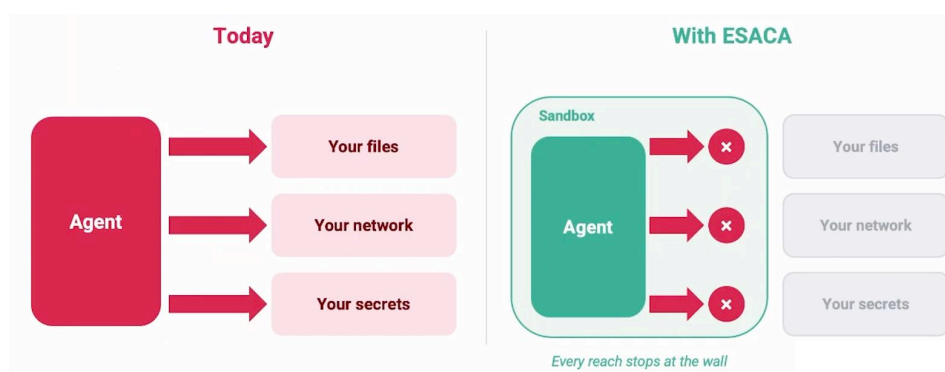
down Area Form Widget – integrations into the Form and Content Engine are underway. The Core Markdown Widget is a standalone React component built on Lexical, the open source text editor framework. It offers "view", "edit", and "source" modes and can be extended through a plugin SDK. More is planned, including real-time collaborative editing and integration with the A12 Print Engine.



ESACA: Agentic Coding with European Data Sovereignty

Coding agents speed up software development, but they bring security risks. An agent running on the host machine has full access to the file system, the network, and stored secrets. ESACA, the [European Sovereign Agentic Coding Appliance](#), solves this. The agent runs inside a sandbox, isolated from the host. No data leaves the sandbox uncontrolled: every outbound request goes through an egress proxy with a default-deny allowlist. Tokens and credentials stay in a local secret vault. They never enter the sandbox. The proxy adds them to outgoing requests at

runtime. Code, credentials, and data stay under your control. ESACA works with different coding agents. Claude Code and OpenCode are supported first. A12 is supported natively: a few clicks create a fully configured sandbox with the A12 plugins, the A12 Knowledge MCP, and matching A12 egress rules. ESACA comes as a desktop app and as a CLI, for Windows, macOS, and Linux. Both create the same sandbox. Teams can choose the graphical interface or the command line.



A12 App Factory: From Infrastructure to a Complete Project Environment

Faster time to value for A12 applications, with less effort along the way — that's the core goal of the A12 App Factory. With new features, the infrastructure and management platform now covers the entire journey: from business modeling to scalable deployment. While individual applications took center stage in the past, the focus now shifts to managing entire application landscapes - including secure AI support based on the ESACA sandbox. For some time now, users have been able to create environments for development and quality assurance at the click of a button using the A12 Project Template (with or without workflows), configure access permissions for team members, control build processes, and monitor the status of deployed envi-

ronments. New additions include support for production deployments, custom project templates, and a web-based instance of the Simple Model Editor for modeling; GitLab is now supported as a second repository alongside Bitbucket, which is particularly beneficial for open-source projects. A web-based IDE for traditional development tasks will follow later this year. Teams will then be able to work entirely in the cloud environment, making local machine configuration unnecessary. Additional AI features are on the roadmap for 2027, such as those for optimizing operations, as well as new governance features for the application landscape - for example, for setting up a data catalog and maintaining data flows between applications in a model-driven way.

On the User's Behalf: How UAA Securely Authenticates Machines with Personal Access Tokens and OAuth Discovery

Until now, UAA supported two basic types of authentication: interactive user login via OAuth2, LDAP/Local, or SAML, and login without a user interface via API key or certificate. As AI agents become more widespread, however, a third scenario is emerging: how does a machine authenticate when it is meant to act on behalf of a specific user? UAA closes this gap with two new features. The Personal Access Token (PAT) is a new authentication type: a long, randomly generated string that users create themselves and pass on to a client. Access is limited to a whitelisted set of endpoints, and the token either expires on a defined schedule or can be revoked as needed. The second addition, OAuth Discovery, takes the opposite route: instead of the user generating a token and handing it over, the client obtains access itself. To enable this, the server publishes machine-readable metadata at a standardized address, telling the client which identity provider is responsible, where the authorization and token endpoints are located, and which methods are supported. On that basis, the client starts the login flow, opens a browser window for sign-in, and then receives a short-lived token bound to the individual user. This allows clients such as Claude or Cursor to be integrated seamlessly, without long-lived credentials circulating.

New token

Token name

Valid for (days)

CANCEL GENERATE TOKEN

Token created

Copy this token now. It is shown exactly once and cannot be retrieved again.

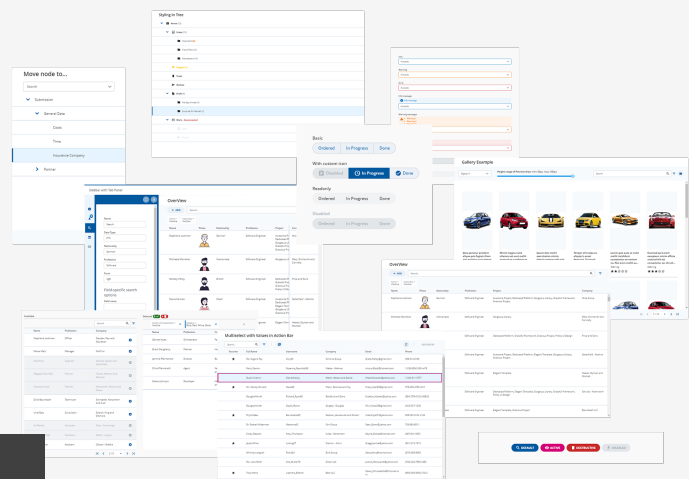
first PAT token

uaapat_b18oFDeqDBYvcqlyWmX1iq_7b8GzuvYb87lpR1rsdKPRZjxV...
COPY

DONE

A12 Widgets

A12 Widgets is a comprehensive collection of prebuilt components designed to provide a pleasurable and accessible user experience.



A12-Widget-Showcase

<https://www.mgm-tp.com/a12.htmlshowcase/>