Dev Blog:

OpenCms goes Jakarta

The Java ecosystem has completed its move to Jakarta EE, and OpenCms is moving with it. OpenCms 22, planned for October 2026, runs on Tomcat 11 and Jakarta EE, with a legacy variant so existing installations can follow at their own pace.

By Alkacon Software
Jul 14, 2026

If you have followed OpenCms development for a while, you may have wondered when OpenCms would move to Jakarta EE. That time has come, and the story of why it is now, and not earlier, is mostly a story about dependencies.

Like every mature Java application, OpenCms builds on many external libraries. Over the last years, more and more of them made the move to Jakarta, but two of the biggest did not: Apache Solr, which powers the OpenCms search, and Vaadin, the framework behind the Workplace UI.

Vaadin was the harder case of the two. The OpenCms Workplace is built on a Vaadin version whose open source line ended a few years ago; the final open source releases only support the old servlet API, and after the license change we had given up hope for a properly converted Jakarta version. Replacing Vaadin entirely would mean rewriting large parts of the Workplace, a huge effort that we wanted to avoid.

For a time, waiting was simply the right call. If we are honest, we half hoped the namespace split would eventually blow over. And Jetty 12 gave us a modern, fully supported runtime that can still host javax based applications, so we made it our default. There was more than one blocker, and little pressure to solve the hardest one.

That picture has changed. With Solr 10, released this March, the last of our major dependencies besides Vaadin completed the switch. And when we took a closer look at today's migration tooling, the Vaadin problem turned out to be solvable too: using a Gradle plugin that builds on the Apache Tomcat Migration Tool for Jakarta EE, we could convert the Vaadin dependencies to the new namespace automatically. With that, the Workplace keeps running on modern servlet containers without a rewrite. From there, the rest of the migration was manageable work. In fact, the biggest part is already behind us: the opencms-core sources have been converted, active development is now Jakarta based, and everything runs as expected.

The namespace change in a nutshell

Tomcat 11 is based on Jakarta EE 11. The relevant change for OpenCms happened earlier, in Jakarta EE 9: the namespace migration. Code that previously used packages such as:

```
javax.servlet.*
```

now needs to use:

```
jakarta.servlet.*
```

This is more than a simple rename. A web application *has to be written* in the namespace expected by the servlet container. Therefore, the same OpenCms artifact cannot run unchanged on both Tomcat 9 and Tomcat 10 or 11. Jetty 12 blurs this line with its EE 8 compatibility layer, which is what allowed us to host the javax based OpenCms on a modern runtime for the last releases, but that layer is a bridge for existing applications, not a destination for new development.

One codebase, two variants

To make the move easy for existing installations, OpenCms 22 will be available in two variants:

- a Jakarta variant for Tomcat 10 / 11 based environments
- a legacy javax variant for existing Tomcat 9 based environments

Both variants come from one codebase. In the opencms-core repository, we converted the sources to the new namespace and made the result the new **main** branch, so all active development now happens on Jakarta, with the Vaadin dependencies transformed. For a release, the main branch is copied over the legacy **master** branch and converted back, and the Tomcat 9 variant is compiled from there. The original Vaadin dependency works as is for that build, so no transformation is needed in the backward direction.

What changes for developers

For many OpenCms projects, not much changes right away. The main question is which servlet container you want to run on.

Projects using standard OpenCms functionality can simply pick the variant matching their runtime environment. Custom modules that directly reference servlet APIs may need small adjustments when moving to the Jakarta variant, in most cases an import change from javax.servlet to jakarta.servlet.

There are advantages that come with that change. The Jakarta platform has kept evolving and offers new features like:

- **A cleaner Servlet API.** Servlet 6.x drops long-deprecated APIs and adds practical details like direct SameSite cookie support.
- **Modern JSP / EL.** The Expression Language now handles newer Java constructs such as records.
- **Ecosystem compatibility.** The big one in practice: current Java web libraries are jakarta-only, so custom modules can finally use up-to-date dependencies.

For new development, Jakarta EE is the recommended direction.

The upgrade path

OpenCms 22 gives projects time to move step by step:

- Continue running existing installations with the legacy variant if needed.
- Test custom modules and integrations with the Jakarta variant.
- Move to Tomcat 11 when the project is ready.

The following major release, OpenCms 23 (April 2027), will complete the transition and will be Jakarta-only.

Looking ahead

Jetty remains an excellent runtime for OpenCms, and we continue to like it a lot. But many projects and hosting environments prefer Tomcat, and with Jakarta EE 11 support OpenCms is aligned with the current Java web platform.

By moving the source code to Jakarta while providing a compatibility variant for one transition release, OpenCms 22 gives every project a clean technical path forward, without forcing anyone to migrate immediately.