Liferay®

# Jakarta Migration Field Guide

# Version History

| Date | Version | Changes |
|---|---|---|
| Aug 06 2025 | 1.0 | Initial draft of the field guide. |
| Aug 18 2025 | 1.1 | Incorporating Lessons from Mastering Liferay project and from Dave Neginber's blog post (Matheus Monteiro) |
| Sept 9 2025 | 1.2 | Updated formatting, introductions |
| Dec 12 2025 | 1.3 | Add guidance for upgrading REST Builder projects |
| | | |
| | | |

# Table of Contents

# Purpose of this document

This field guide is a living document, capturing lessons learned from Liferay internal Jakarta upgrade projects. It will be continually updated and will serve as the foundation for a future Jakarta Migration course on learn.liferay.com coming in 2026. Additional resources covering the Jakarta upgrade can be found on the Jakarta information page on learn.liferay.com.

This document provides both an explanation and a guide to the steps required to move a Liferay DXP application from Java EE to Jakarta EE.

It outlines:

- What is Jakarta EE and what is impacted
- What changes must be made
- How to prepare for the upgrade to Jakarta EE
- How to make the required changes, and the tools that are provided to help
- What to look for post-upgrade to ensure that applications are functioning properly
- Common issues or pitfalls

# Introduction

## What is Jakarta EE?

Java EE (Java Platform, Enterprise Edition) was the long-established standard for developing enterprise Java applications. It defined APIs for building web, REST, messaging, and transactional systems, and was originally maintained by Sun Microsystems and later Oracle.

In 2017, Oracle transferred the stewardship of Java EE to the **Eclipse Foundation**, marking a major milestone in the evolution of the platform. This transition gave rise to **Jakarta EE**, a community-driven and open-source continuation of Java EE.

Due to naming rights issues, the latest release of the framework was forced to move away from the Java EE name and use Jakarta EE in its place.

## Why is this change necessary?

Due to naming rights issues, the latest version of the Java Enterprise framework has been renamed from Java EE to Jakarta EE.

This impacts any Java-based application that moves to Jakarta because the underlying API is also renamed - from 'javax' to 'jakarta'.  Accordingly, any code that uses the core APIs must be updated from 'javax' to 'jakarta'.

Liferay DXP is moving from Java EE 8 to Jakarta EE 10 from the 2025.Q3 release. The move is beneficial for all customers because it brings enhanced features, improved performance, more robust security features and better stability.

## Impact on Liferay

Liferay's own code has been updated starting from the 2025.Q3 release, and as such, any customer using this version of the platform or any later version, must also make changes to their custom code to ensure it is compatible, and also to third-party libraries used in that custom code.

Liferay is providing tooling to make this process as easy as possible, along with guides to enable customers to complete the process and successfully transition to Liferay DXP 2025.Q3 or later.

# Migration Steps

## Preparing your Migration Environment

Before making any code or data changes, it's crucial to establish a stable and isolated foundation. This section walks you through preparing your codebase and database to ensure a smooth and reversible upgrade process.

### Prereqs

- Git (or VCS) available
- Workspace builds clean on current version
- Working Liferay DXP version on current version
- No unresolved dependencies

### Steps

1. Checkout all your code
2. Create full backup from Production (PRD) data (including Document Library (DL))
3. If you use git, create a different branch based on master (or whatever you use in PRD)

### Expected

- Isolated branch for all upgrade changes
- Ability to diff/revert easily

### Checklist

- ☐ Branch created
- ☐ Baseline committed or backup archived

## Migration Setup

With your environment prepared, the next step is to configure your Liferay workspace to recognize the new target version and incorporate the necessary upgrade tools provided by Liferay.

### Prereqs

- Supported version of Blade and Gradle

- Workspace root path known

### Steps

1. Remove the old bundle
2. Upgrade the workspace version in settings.gradle to **14.0.0** at a minimum
3. In your gradle.properties file, set liferay.workspace.target.platform.version to the latest patch release for the quarterly release you are targeting, e.g. **dxp-2025.q3.2**
   **Important: Do not use "latest".**
4. Initialize the target bundle.

### Expected

- Formatter knows your target release
- A fresh **Tomcat 10-based bundle** is downloaded for the version target specified

### Checklist

- ☐ bundles/ regenerated for the target

## (Optional) Migrate REST Builder Projects

If you have REST Builder projects in your workspace, you must re-configure them for Jakarta.

1. In your rest-config.yaml file, specify you now use the Jakarta namespace:
   a. javaEEPackage: "jakarta"
2. Run REST Builder:
   a. blade gw buildREST

See the Jakarta upgrade documentation for more details and an example.

## Run Jakarta Transform

This is the core automated step of the migration. Here, you will use Liferay's tooling to perform the bulk of the code modifications, primarily updating the javax.* package namespace to the new jakarta.* standard across your projects.

### Prereqs

- Workspace compiles on current baseline.

## Steps

1. Run the **JakartaTransform** tool
   blade gw upgradeJakarta
2. Commit changes.

## Expected

- javax.* → jakarta.* updates across Java, annotations, tags (DB templates may require manual follow-up).

## Checklist

- ☐ JakartaTransform finished

# Review Transformation

The Jakarta Transform tool is only aware of libraries used by Liferay, so there is a chance that the transformation process may not identify all references to javax.*. It is recommended that before continuing the migration process you review for any remaining references.

## Prereqs

- JakartaTransform process has completed successfully

## Steps

1. Search for 'javax' across the entire codebase
2. For any identified references, determine the correct replacement.
3. Commit changes

## Expected

- There are no remaining references to javax in the codebase.

## Checklist

- ☐ Manual review and replacement finished

# Third-Party Dependencies

Your custom code often relies on external libraries, and these must also be compatible with Jakarta EE. This section addresses the critical task of identifying, updating, or transforming these third-party dependencies.

## Prereqs

- Inventory of libraries / Knowledge of Customizations

## Steps

1. **Substitute or Replace Obsolete Jars with Jakarta Counterparts:** Adopt official Jakarta-compatible versions of your dependencies (and transitive deps). Optionally, contact maintainers for Jakarta variants if none are published.
    a. If necessary, transform jars using the Eclipse Transformer to rewrite javax→jakarta for a quick unblock. This page will guide you through the necessary steps.

## Expected

- Build unblocked now; sustainable path defined for later.

## Checklist

- ☐ Decide: transform now vs upgrade now
- ☐ Transitives mapped
- ☐ Document your changes and plan (read.me file or any other relevant document for your company/organization)

# Build & Deploy to Bundle

After transforming your code and updating dependencies, you must verify that the project still compiles and deploys without errors. This step confirms that the foundational changes have been successful before you proceed to runtime testing.

## Prereqs

- Bundle initialized for the target release.
- All prior steps committed.

### Steps

1. Clean build & deploy modules:

```
 blade gw clean deploy
# Windows:
..\..\gradlew.bat clean deploy
```

### Expected

- All modules build and deploy to bundles/.

### Checklist

- clean deploy completes
- All compile errors resolved

## Database Upgrade

To align with the new Liferay DXP version, your database schema must also be upgraded. This section covers running the dedicated tool to bring your data into compliance with the target platform.

### Prereqs

- Full DB backup.
- Access to Documents & Media filestore.

### Steps

Follow the steps listed in the official Liferay documentation for using the Database Upgrade tool.

### Expected

- DB upgraded to match target bundle.

### Checklist

- Backup verified
- DB upgrade completed cleanly

## First Run

With the code and database upgraded, it's time for the initial system startup. This section guides you on what to look for in the server logs and which key functionalities to test to catch any major issues immediately.

### Prereqs

- Successful build & DB upgrade.

### Steps

1. Start Liferay and scan logs for:
    - Unresolved OSGi references / circular refs.
    - Template errors (e.g., PropsUtil moved to com.liferay.portal.kernel.util.PropsUtil).
2. Navigate functional areas:
    - Portlets, JSPs, JS behaviors.
    - FreeMarker templates

It is highly recommended to check the Liferay official breaking changes document for your target version:

- 2025.Q2 Breaking Changes
- 2025.Q3 Breaking Changes

There might be unexpected classes, API changes that will affect your templates, fragments and you might need to deal with it.
Analyze the logs and you might find a clue in there.

### Expected

- No severe startup errors.
- Core user functionalities working.

# Conclusion

The migration to Jakarta EE represents a significant evolution for Liferay DXP, bringing enhanced features, improved performance, and a more robust and secure platform. This field guide has outlined the essential steps, from preparing your environment and transforming code to updating dependencies and performing initial system checks, all aimed at facilitating a smooth transition. By following these guidelines and leveraging the provided tooling, you can successfully upgrade your Liferay DXP applications to Jakarta EE.

This is a living document, and your feedback is invaluable. If you have any questions, encounter issues not covered here, or have suggestions for improvement, please reach out to jakarta-questions@liferay.com. Your contributions will help us refine this guide and the future Jakarta Migration course.