Migrating WebSphere Liberty 21.0.0.x to Payara Server Enterprise 5

The Payara® Platform - Production-Ready,
Cloud Native and Aggressively Compatible.

Migration Guide

Contents

Introduction 1
WebSphere Products 2
Releases 3
Enterprise Standards 4
MicroProfile 4
Clustering 5
Operations 6
Monitoring 7
Diagnostics 8
Deployment 9
Cloud 9
IDE 10
Support 10
The Features You Use in WebSphere Are Available in the Payara Products 10
How is Payara Server Enterprise Better Than WebSphere Liberty? 11
About Payara Enterprise 12

Introduction

Migrating applications from WebSphere Liberty to Payara Server 5 can be a simple and straight-forward process because both servers rely on the Jakarta EE (Java EE) specifications. However, there are differences in many areas because many Java EE APIs in WebSphere and Payara Server Enterprise are implemented by different components. Moreover, the configuration of certain aspects like external resources, high availability, and deployment is not covered by any specification and is, in fact, very different in both servers.

This guide gives you an overview of the differences between WebSphere Liberty and Payara Server, which components and features in Payara Server are equivalent to those in WebSphere, and how to configure some frequently-used resources and features in Payara Server compared to how you're used to them in WebSphere.

If you are running WebSphere Application Server, have a look at our guide where we discuss the migration for this product in more detail.

Payara Server is based initially on GlassFish Server, and they both still share a lot of code, concepts, and ecosystem tools. Since Sun Microsystems first developed GlassFish as the reference implementation (RI) of Java EE, it has been used by both developers in its open source form and by Sun - then, later, by Oracle - in production fully supported. Oracle GlassFish fared well in a competitive market and became a popular choice, with the vast majority using the GlassFish Server Open Source Edition. In late 2013, Oracle announced the end of commercial support for GlassFish.

One year after the original announcement of the end of support from Oracle, Steve Millidge - a Java EE expert and the founder of Payara Services Ltd - announced the arrival of Payara Server and, with it, a reintroduction of a fully supported option with a very similar platform and the added reassurance of regular releases, bug fixes, patches, and enhancements.

Payara Server evolved rapidly from just a "supported GlassFish alternative" to a fully open source product that can run Java EE and Jakarta EE applications. It contains several unique features in monitoring and security, for example, and was the **first runtime that integrates Java EE and Eclipse MicroProfile®**.

Starting in June 2020, Payara Server split into two editions:

- **Payara Server Community** - geared towards innovation and rapid development, is entirely free to use in development environments. This edition aims to allow the general developer community to rapidly adopt new features that might not be suitable for production environments until they are stable enough. This edition can be compared with WebSphere OpenLiberty, but there are some differences.
- **Payara Server Enterprise** - tailored for use in production environments for extended periods with guaranteed stability. A Payara Enterprise subscription is needed to run the server in production environments, and customers with an active subscription get access to premium support and specific feature sets that are not available in the Community Edition. This edition adopts innovative features at a much slower pace since they will only be integrated when they are stable enough. This edition is directly comparable to the WebSphere products, as they are premium products with commercial support. Commercially licensed Payara Server Enterprise focuses on the requirements and needs of the customers to optimize their deployment and management scenarios and the system stability at runtime

The rest of this document will strictly reference Payara Server Enterprise Edition since it is the recommended product to run in a production environment.

WebSphere Products

Looking at the product page on the IBM website, there are three products with the name WebSphere.

WebSphere Application Server (WAS) is the typical application server that implements the Java EE specifications. The current version 9.0.5 supports the Java EE 8 specification, but there seems to be no intent to evolve the standards in the product. WebSphere Application Server is not listed as a compatible implementation of the Jakarta EE specification on the website, the continuation of the Java EE specification under the governance of the Eclipse Foundation. Support for running on a JDK 11 runtime and within containers is added, but this product seems to be discontinued regarding new functionality.

WebSphere Liberty is the product where new development efforts can be found. The product's architecture and operational aspects are different from the Application Server version. For example, you can download a version that supports the Jakarta EE specification. Still, the idea behind Liberty is that you have a modular runtime where you need to define what specification and features you want to be available at runtime. It either downloads it when you start the server, or you can prepare a specific runtime upfront. The product page mentions this is specifically for microservices, although it can also be used to run more traditional applications.

Payara Server and Payara Micro don't have this modular approach but instead have very tight integration between the different specifications and features in our platform. This allows us to achieve various functionalities, like MicroProfile tracing support for remote EJB calls and better performance since we don't have an abstraction layer if some modules are not available at runtime.

The final product is **WebSphere Application Server Network Deployment** and has the following exciting description "server runtime environment is for large scale and mission-critical application deployments." It is based on the Application Server version and thus again uses the older Java EE specification. There are additional features to use the runtime in a clustered setup, and the mission-critical notion suggests that the other versions don't have these capabilities.

The Payara Platform products have a more straightforward approach and can be used interchangeably without learning an entirely new way of managing your environment.

Payara Server can be identified with the more traditional application server you first install, perform the required configuration, and deploy the applications in the last step. This product still follows the latest standards from the industry and is compatible with Jakarta EE and MicroProfile specifications.

Payara Micro is a different packaging of the same codebase that brings you the Jakarta Web profile and MicroProfile specification as a runtime. You can start the executable jar file containing Payara Micro and, on the fly, perform the configuration through the same commands that work on Payara Server. You can bundle the web application in the same jar file and keep them separated as that is preferred for many environments.

Both are also targeted for mission-critical environments and have support for clustering.

Releases

The WebSphere Liberty product has monthly releases as it is based on the OpenLiberty product, which is also released monthly. This monthly release cadence is the same as the Payara products and contains bug fixes, security fixes, and new features.

But it seems IBM is pushing you to a monthly update of the Liberty product. If you create a starter project or a MicroProfile Starter demo project, the version is actually a range, which means that you download the latest version each month. This is not the enterprise vision we have at Payara, where you can decide if you want to upgrade with each release or not. You can upgrade, but you do not need to do it. And the version range also contradicts the requirement to have reproducible builds of your application.

Enterprise Standards

The WebSphere Liberty product supports the Jakarta EE 8 standards, just as Payara Server does. Since both products are Jakarta EE certified implementations, application should behave identically on both products unless you use some framework-specific functionalities. The following table lists the differences in implementation of the main specifications.

Specification implementation	WebSphere Liberty 21.0.0.x	Payara Server 5
JAX-RS	Apache CXF, RestEasy in the future	Jersey
JSON	Jackson	Yasson
JPA	Hibernate	EclipseLink
JSF	MyFaces	Mojarra

Each of the implementations is certified against the corresponding specification and should behave the same. But each of those implementations also provides some additional functionality that is not yet available within the specification. When a developer uses these additional features, deploying the Jakarta EE application unaltered on Payara Server is no longer possible.

Support for the new Jakarta namespace within Jakarta EE 9. x is available with the beta releases of the product. Both Payara and WebSphere support this new namespace in a pre-release version as there is no new functionality compared to Jakarta EE 8. With Payara we clearly indicate the fact of this breaking change by a change in the major version number. With WebSphere Liberty, the numbering is based on the year and does not give you that information.

MicroProfile

The WebSphere Liberty product follows up on new development in the standards world and thus also supports the latest version of MicroProfile. Since the Payara products also support the newest version, a migration of an application using the MicroProfile specifications should not pose any problem.

The WebSphere Liberty product also implements standalone specifications like Long Running Actions and GraphQL. Payara Server doesn't have an implementation for these optional specifications of MicroProfile at this moment. If you use one of these specifications, there is currently no replacement available within Payara.

Clustering

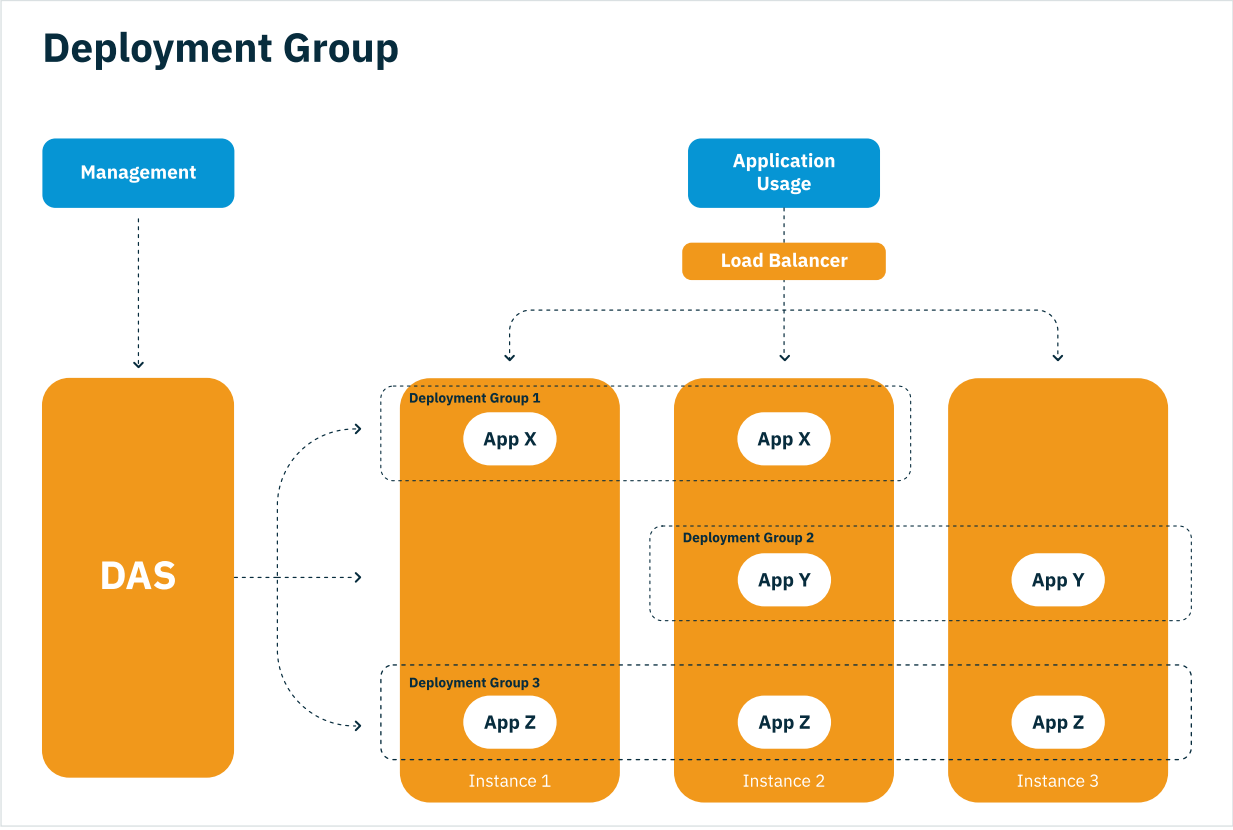
The WebSphere products allow you to define a clustered environment to distribute the user requests over different resources to achieve higher throughput. You need to add a specific feature to the server to have clustering capabilities.

The clustering works in the traditional way, having a specific machine joining the cluster and running all your applications.

You can create the same environment using Payara Server, where each instance in the cluster runs the applications. If you are using a clustered setup with WebSphere, it will be straightforward to create a similar environment with Payara Server.

But Payara Server also has a very flexible alternative called the Deployment Groups. A Deployment Group is a logical grouping of instances, not a physical one. This means that a Payara Server instance can be part of one or more Deployment Groups. And instead of defining the resources, like the database connections, and the applications to an instance, you can assign it to a Deployment Group.

Payara Server ensures in this case that all instances that are part of the Deployment Group get all the resources. These Deployment Groups are also dynamic, instances can be added and removed, and the Payara Server Domain Server makes sure all deployments are performed or removed.



Related to clustering, a data cache option in those clustered situations is used many times. This way, you can easily share the information between the different instances of the cluster. The WebSphere products can use the IBM eXtreme Scale product that needs to be purchased separately. It is an elastic, scalable in-memory data grid.

As a developer, you can access the functionality of this product through the JCache specification, the Distributed Map option of WebSphere, or directly. The Distributed Map option can also store data locally, without the need for the in-memory data grid. This feature can be achieved by caching the outcome of the method on the instance.

While you have several options, the Payara solution is more straightforward and doesn't require purchasing an additional product. The Payara products are integrated with Hazelcast. The clustering capabilities of Payara are built on top of this framework, but you can also for data caching solutions. You can cache any value using the JCache specification. Your application doesn't need to know if it runs standalone or clustered. Hazelcast will make sure the data is available on all instances. And this functionality is available out of the box with all the Payara products installation, without needing an additional installation or configuration.

Operations

An essential aspect of maintaining your environment is the ability to use scripts to set up and change the installation.

With the WebSphere products, you can use the wsadmin tool to manage the server and the domain. Besides these basic commands, you can combine this client with some Jython scripts to unlock the full potential of the configuration capabilities.

You also can manage your environment through a graphical interface in the browser. You can view, change and configure your entire WebSphere environment from the admincenter application.

However, maintaining your environment through a graphical interface is not ideal. You cannot automate it and will make more mistakes. That is why the Payara Admin Console has the asadmin recorder that generates a script that performs the same actions you did manually.

These scripts use the asadmin CLI tool capable of performing all operations on your Payara Environment. It can set up the different servers and instances, deploy applications, and configure all aspects of the server and your applications. You only need to learn one tool and don't need to use a mixture of the wsadmin tool and the Jython scripts that are based on a very old and no longer supported Jython version.

When moving from WebSphere to Payara, you will need to learn all the capabilities of the asadmin CLI tool. Our documentation and the web Admin Console can help you with that.

Monitoring

The Performance Monitoring Infrastructure of WebSphere helps you keep an eye on the performance and health of your installation. You can follow up on the typical Java values like Heap usage, connection pool metrics like the JDBC one, Dynamic cache statistics, and various other metrics from other modules.

Based on these metrics, you can define alerts that notify you by email that your environment is not running optimally and potentially some actions are required.

The metric information can also be integrated with external tools like Tivoli, which is also an IBM product, but also exported to Prometheus and visualized with Grafana.

The Payara products have similar capabilities around monitoring and metrics and have more options regarding the notifications in case an alert is raised. Besides the SMTP option, you can also send the notifications through JMS, CDI event bus, XMPP, and products like Slack, Datadog, and New Relic. We also have the notifiers API so that you can create your notification channel in case the default ones are not suited for your case.

Within Payara, you also have a monitoring service where you can configure the level of monitoring for the different subsystems. Once active, many metrics are gathered and can be consulted using Payara tools or through some integration. Since data are stored as JMX beans, any JMX client can be used to have a look at the metrics. But we also have a graphical UI, Payara InSight, our monitoring console, that gives you an insight into your environment.

Another option is to expose the data through Prometheus and use, for example Grafana.

Monitoring of the Payara products needs to be set up differently than on WebSphere, but Payara has the same capabilities regarding monitoring along with many additional possibilities with Payara.

Diagnostics

Besides the monitoring, some additional diagnostics tools are helpful to analyze potential issues in your environment.

As part of its MicroProfile support, WebSphere Liberty has the MicroProfile OpenTracing implementation. This allows you to follow a user request through the different services to assemble the response for the user.

Besides this, there are traditional tools like analyzing the logs or taking thread and heap dumps from the JVM itself.

But the diagnostic capabilities of Payara go much further. Payara has an implementation of MicroProfile OpenTracing but extends the tracing capabilities also to Servlet, remote EJB calls, SOAP calls, Batch job creation, and Tasks executed by the managed executors. This will give you a much more detailed insight into the different aspects of the user requests.

But Payara Platform has more diagnostics tools that you can use. You can define, for example that SQL queries are logged that take longer than a predefined time. This will help you to identify the slow responses. You can also get notifications when connections are not returned in a timely fashion to the pool. This also helps you to identify the areas that need attention to increase performance.

A similar diagnostic is available for threads that are not returned to the managed executor. These Stuck Threads checks can also help you to identify performance issues but this time around, the code execution.

When migrating to Payara, you will discover that you have even more diagnostics tools at your disposal to make sure your environment runs at its optimal.

Deployment

The WebSphere products support the usage of variables so that an application can be deployed unaltered in different environments. The differences in environment can be captured in different values for WebSphere variables or environment variables. With placeholders in the WebSphere configuration files like server.xml, the actual values are used for the environment.

The Payara Products have the same capabilities as you can also use environment variables in the runtime configuration. For sensitive data, you can use the password alias option of Payara Server so that passwords, for example are not visible within the configuration files themselves.

But the support of Variables within Payara is also available in many other descriptors files like web.xml, persistence.xml, and annotations of the Jakarta EE specification like @EJB, @DataSource, @WebServlet, etc....

It shouldn't be a problem to use the same abstraction of values for the implementation within Payara regarding WebSphere's capabilities.

Cloud

Official Docker images exist for the WebSphere Liberty products. There is also a Kubernetes Operator to handle the installation of WebSphere Liberty on Kubernetes.

The same functionality is available for Payara Products so that you can migrate easily from a WebSphere-based setup to Payara using cloud and containerized environments. The usage of containerized environments is probably much easier if you use Payara Micro to run your Jakarta Web profile-based or MicroProfile based application. Payara Micro can automatically form a cluster within Kubernetes without the need for any additional tool or framework. You don't need a Kubernetes Operator and can use standard Kubernetes functionality like Kubernetes Services and Horizontal scalers to achieve a dynamically scaling environment.

IDE

IDE plugins exist for Eclipse IDE and Visual Studio Code, and a plugin for IntelliJ is under development. There is no plugin available for NetBeans. With these plugins, you can control the WebSphere server (start and stop, for example) and help you run and configure the server from within the IDE.

Payara has plugins for all four major IDEs and allows the same functionality as the WebSphere ones. You can control and configure the Payara instances and start a new project with all the Payara public dependencies.

Support

The Licensing cost for the WebSphere products is based on how the product is installed and used. It is based on Processor Value Unit, a value based on the number and type/brand of the CPU cores, or Virtual Processor Core, the vCPU values in virtual and containerized environments.

Payara Enterprise has a much easier licensing model, which is just based on the number of cores (physical or virtually) that can be used by the Payara process. Within the subscription, there is also support included for your environment. This relates to the setup and operation of the environment itself but also includes questions about application-specific best practices or possibilities and all systems related to the Payara installation like a Load balancer.

The Features You Use in WebSphere Are Available in the Payara Products

Besides a few standalone MicroProfile specifications, all features available on the WebSphere Application Server and WebSphere Liberty product have a corresponding counterpart within the Payara Products. After migration, you will have even more monitoring, diagnostic, and clustering tools available that make operating your environment easier.

Also, the similarity between Payara Server and Payara Micro makes it easy to switch from one runtime to the other, even without changing the application or major changes into the operational aspects since they share the same core.

How is Payara Server Enterprise Better Than WebSphere Liberty?

	WebSphere Liberty 21.0.0	Payara Server Enterprise 5
License	Proprietary	Open Source with EULA
Release Frequency	Monthly	Monthly
Production Support	✓	✓
Supported IDEs	Eclipse IDE, VS Code	Eclipse IDE, NetBeans, IntelliJ IDE, VS Code
Java EE 8 Compatible	✓	✗
MicroProfile Support	✓	✓
Microservices Edition	✓ (Liberty is positioned as Microservices only)	✓ (Payara Micro)
Data Grid (Caching)	✗	✓ Included
Clustering	✓	✓ Included
Scripting Tool	Wsadmin and jython scripts	asadmin
Admin Command Recording and Auditing	✗	✓
Deployment Flexibility	Variables (Environment, WebSphere variables, MicroProfile config)	Variables (System, Environment, MicroProfile config)
Notification Channels Monitoring	• JMS Beans • SMTP	• JS Beans • Log Files • SMTP • SNMP • JMS • CDI • Slack • New Relic • Datadog • Teams Channels
Custom Notifiers	✗	✓

About Payara Enterprise

Payara Platform Enterprise is stable, supported software for enterprise designed for mission critical production systems and containerized Jakarta EE (Java EE) and MicroProfile applications.

Quickly and easily deliver Jakarta EE (Java EE) apps in any environment: on premise, in the cloud, or hybrid with enterprise-grade security. A 10-year software lifecycle lets you decide when to migrate from one release to the next and a choice of 24x7 or 10x5 production support ensures you're fully covered when you need it.

Choose Payara Server for reliable and secure deployments of Jakarta EE and MicroProfile applications in any environment, or Payara Micro for containerized microservices deployments with no installation, configuration, or code rewrites required.

Talk to us and request Payara Enterprise or find out about our Migration & Project Support option to help you migrate to the Payara Platform.

sales@payara.fish **+44 207 754 0481** **www.payara.fish**

Payara Services Ltd 2022 All Rights Reserved. Registered in England and Wales; Registration Number 09998946
Registered Office: Malvern Hills Science Park, Geraldine Road, Malvern, United Kingdom, WR14 3SZ