E-Book

Microsoft

How Microsoft applies Java

THE INSIDE STORY

By Bruno Borges, Principal PM Manager, Java Engineering Group
& Theresa Nguyen, Senior Product Manager, Java Engineering Group.

About the authors

Bruno Borges is a Principal PM Manager in the Java Engineering Group at Microsoft. He joined the company in 2018 after a six-year career as a Product Manager for Java related technologies at Oracle, and later-on in a Developer Engagement and Experience role on Oracle Cloud. Bruno has been involved in Java development and the Java community since 2001.

Theresa Nguyen is a Senior Product Manager in the Java Engineering Group at Microsoft. She joined the company in 2018 after ten years at Caucho Technology and Tomitribe, two Java EE application server vendors. Theresa has been involved with Java and the Java community since 2008.

How Microsoft applies Java

Introduction

Over the last decade, we've witnessed Microsoft's evolution from a focus on Windows and .NET to a mindset that embraces open-source software and philosophy. Today, Microsoft employs thousands of people focused on Java across engineering, developer relations, customer support, and other parts of the business.

When we talk to customers and developers, we're often surprised at how so many are unaware of the extent to which Microsoft applies, implements, and supports Java:

1. We use Java. A lot. We're using Java to run major parts of Microsoft—and empower thousands of customers to do the same.
2. We're listening to feedback from the Java developer community and making meaningful contributions to existing community frameworks.
3. Our teams across Microsoft employ thousands of Java professionals who innovate and integrate Java into Microsoft products and technologies and empower Java developers to do more.

In this paper, we'll look at:

- A historical look at the ways in which Microsoft has adopted Java over the years.
- The recent history of Microsoft's investments in Java.
- What we're doing today to help Java developers succeed.
- How various groups within Microsoft are using Java internally.

The Evolution of Java at Microsoft

Microsoft opens up to open source

Starts investing in Java resources and expertise

2008	• Microsoft Bing, the company's search engine is deployed across multiple datacenters, spanning hundreds of thousands of JVMs on more than 50,000 servers.
2009	• Starts building 1st class support for Java on Microsoft Azure.

Microsoft commits to shaping Azure as an open platform

Makes additional investments in Java partnerships

2011	• Commits to shaping Azure as an open platform.
2013	• Azure becomes the first major cloud platform with included OpenJDK commercial support.

Microsoft Azure's transformation to an open platform gains rapid momentum under Nadella's leadership

2014	• Releases Azure Toolkit for Eclipse.
2015	• Releases Azure Toolkit for IntelliJ. • Releases support for Java on Azure App Service.
2016	• Partners with Pivotal/VMWare around Spring and Azure. • Microsoft joins Eclipse Foundation.
2018	• Releases scalable services for open-source databases: MySQL, PostgreSQL, MongoDB, Cassandra

Microsoft and Java - in it for the long term...

2019	• Acquires jClarity, the leading contributor to the AdoptOpenJDK project. • Releases Azure Spring Apps in collaboration with VMware.
2020	• Releases Red Hat JBoss EAP support for Java EE on Azure App Service.
2021	• Becomes founding member of the Eclipse Adoptium Working Group. • Releases Microsoft Build of OpenJDK. • Microsoft joins the Java Community Process (JCP) program.
2022	• 2 million+ JVMs run in production on internal systems across Microsoft. • Microsoft joins the Eclipse Jakarta EE and MicroProfile Working Groups. • Releases Apache Tomcat 10 and Java 17 support on Azure App Service.

"Microsoft's positive approach towards open-source ecosystems and specifically towards Java demonstrates their unbridled passion for enabling developers to achieve more. At jClarity, our team was focused on the good of the overall Java open-source ecosystem via the AdoptOpenJDK project, supporting Java User Groups, and helping developers better understand Java performance. After being acquired by Microsoft, our efforts to empower the Java community have gained even more momentum and we are incredibly proud to be part of what is now a Java at Microsoft steam train!"

-- Martijn Verburg, Principal Group Engineering Manager - Java Engineering Group, Microsoft

Committed to developers and the Java community

Microsoft actively supports Java community organizations to help promote the future of Java, including ongoing development of the Java language and the JVM. Let's take a closer look at the ways in which Microsoft contributes to the Java community's work as part of our commitment to the success of Java Developers.

Microsoft Build of OpenJDK

Released in mid-2021, the Microsoft Build of OpenJDK is based on the OpenJDK source code, built using Eclipse Adoptium build scripts, tested with Eclipse Adoptium AQAvit, and certified with Oracle's JCK (Java Compatibility Kit) for long-term supported versions. Today, it includes binaries for Java 11, and Java 17 on x64 environments on macOS, Linux, and Windows, AArch64/ARM64 on Linux and Windows, binaries for macOS on Apple Silicon (ARM64), and musl-libc compiled binaries for Alpine Linux on x64.

The Microsoft Build of OpenJDK is a drop-in replacement for other builds of OpenJDK. We built it to support both our internal Java workloads and the significant growth we're seeing in the number of Java-based customer workloads on Azure. In addition, running our own build gives us a deeper connection with the Microsoft teams that operate various Azure services that support Java, enabling us to work more closely together in addressing bugs, performance regressions, and other issues that matter to our customers. We also see our own build of OpenJDK as an avenue for even more active participation in the Java ecosystem and increased contribution to the Java community.

Tools for the Java developer

We're constantly listening to feedback from the community and work closely with other teams at Microsoft to deliver the tools that Java developers need and love.

In November 2021, Red Hat released version 1.0 of the Language Support for Java on Visual Studio Code, a multi-year collaboration between Microsoft, Red Hat, and the Java community. This release includes a series of changes, including the stability that the more than 1 million Java developers who use Visual Studio Code have come to expect. Other highlights in the release include Java 17 support.

In parallel, we're working closely with GitHub teams to ensure great support for Java, such as GitHub Codespaces for Java and the ability to use Java in GitHub Actions.

The Java developer community also really enjoys the ease of IntelliJ Azure Toolkit. Azure also works seamlessly with other tools such as Maven, Gradle, Eclipse Toolkit, etc.

Eclipse Foundation and Java Community Process

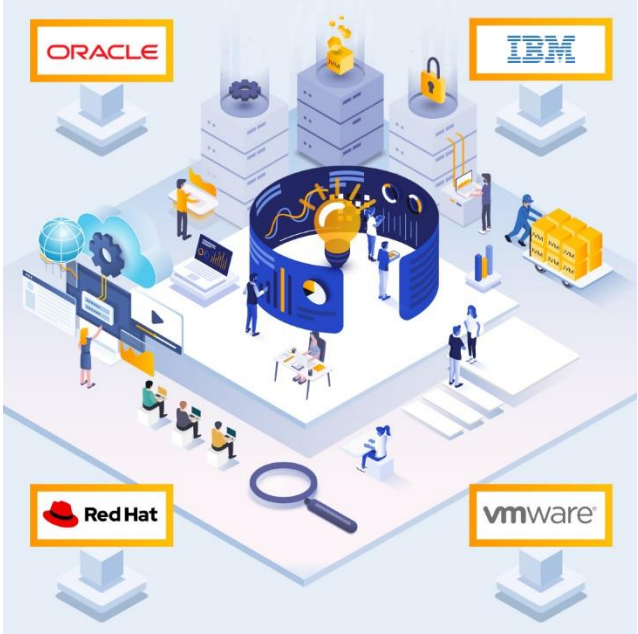
In August 2021, we expanded our participation in the Eclipse Foundation to the role of Strategic Member. Today we're actively participating in Eclipse Adoptium and other projects hosted by the foundation, including the Eclipse Jakarta EE, MicroProfile, IDE and others. A full list of the Eclipse projects to which Microsoft is actively contributing can be found on the Eclipse Foundation website.

In November 2021, we took another major step when Microsoft officially joined the Java Community Process (JCP). For the past 20+ years, JCP has served as the mechanism for collaboration between individual developers, customers, and vendors in developing standard technical specifications for Java to ensure stability and cross-platform compatibility.

Strategic partnerships: VMWare, Red Hat, IBM, Oracle, and more

Over the past few years, we've forged several strategic partnerships with major vendors in the Java ecosystem. These partnerships empower developers to code and deploy without worrying about infrastructure.

Azure Spring Apps, a fully managed service, was jointly developed by Microsoft and Pivotal / VMWare. It abstracts away the complexity of managing infrastructure and Spring Cloud middleware, enabling Java developers to focus on building cloud-native applications while Azure takes care of dynamic scaling, security, high availability, and so on. Azure Spring Apps also offers native integrations with third-party application performance monitoring (APM) tools from New Relic, App Dynamics, Dynatrace, and Elastic.



Next, we partnered with Red Hat to deliver Red Hat JBoss Enterprise Application Platform (EAP) on Azure App Service - enabling Java developers to deploy their Java EE and Jakarta EE applications into Azure App Service with integrated support from both companies. JBoss EAP is also available on Virtual Machines and Virtual Machines Scale Set - through Azure Marketplace mages, for fast deployment and horizontal scaling of Java clusters.

We also partnered with IBM and Oracle, launching jointly developed solutions for WebSphere Application Server, WebSphere Liberty, and Open Liberty on Azure and for Oracle WebLogic Server on Azure VMs and on Azure Kubernetes Service. Finally, in terms of recent partnerships, we worked with Confluent Cloud to help them deliver Apache Kafka for Confluent Cloud, an Azure Marketplace offering that provides Apache Kafka as a fully managed service.

"In 2009, we embarked on the journey to enable Java on Azure and Java has since been an unstoppable force at Microsoft. Today, Azure is the home for thousands of mission-critical, enterprise Java workloads scaled across the globe; the implementations help us learn so we can help developers and customers achieve more. We are grateful for how far we have come and for where we are heading in terms of meeting the needs of Java developers. I am super thrilled to have been a part of this journey since 2009 and I am energized by what lies ahead."

-- Asir Selvasingh, Principal Architect for Java on Microsoft Azure

---

Developer and customer success is our success

Java developers can use a wide range of development and DevOps tools—including Visual Studio Code, IntelliJ, GitHub, Maven and Gradle—to bring their Java apps to the cloud while collaborating effectively and saving time and money. Microsoft has offerings that are jointly developed and supported with its partners to allow developers to keep using their favorite frameworks, tools, and services while benefiting from all that the Azure cloud has to offer.

Our engineers and support teams work closely with our customers to effectively run and scale their Java apps. Customer success stories, options for customer support, and consistent programming of technical content on Java are all ways Microsoft helps developers succeed.

Improvements in technologies, services, and tooling

In addition to our solutions, we work to provide our customers with access to information and resources that help ensure their success. At a high level, those efforts can be broken down into three main areas:

- Supported Azure services that customers can easily consume—such as Azure App Service, Azure Spring Apps, and others.
- Developer tools and services for Java developers —such as the Microsoft Build of OpenJDK, Java extensions for Visual Studio Code, Azure SDK, Spring Cloud Azure libraries, GitHub, Playwright, etc.
- Documentation and support—such as the Microsoft for Java Developers documentation center, Azure docs for Java developers and Microsoft Learn.

Empowering our customers to run and scale their Java apps

Over the past several years, thousands of Microsoft customers have moved to Java on Azure, including well-known companies like Adobe, AIA, Bosch, Daimler, FedEx, J.B. Hunt, Kroger, Maersk, Mercedes Benz, and Swiss Re. We've worked closely with many of these customers, working side-by-side with their engineers to ensure success. Similarly, we work closely with the internal groups at Microsoft who use Java, lending our own expertise while learning from theirs. Through such efforts, we've amassed a substantial body of insight and expertise toward helping new customers build Java applications on Azure or move existing ones to the cloud—including, for example, how moving to the latest versions of Java can help increase performance and reduce costs.

FedEx launched a platform that dynamically predicts estimated time of arrival on more than 16 million shipments a day

"We are committed to Java and are helping organizations modernize their Java applications with Java Spring Apps Enterprise, which we built as part of our expansive partnership with VMWare. FedEx, for example, is using the service to run all their Java apps at massive scale, predicting the arrival time of millions of shipments."

-- Satya Nadella, CEO, Microsoft

"Exposure and consumption of these insights are critical [for our business], and we chose Azure Spring Apps as our principal compute for high-volume processing because of its ease of use, scale, and integrations into the Azure ecosystem. Our collaboration with Microsoft and VMware has delivered an amazing service for fast-moving Java teams to deliver impactful solutions for our customers."

-- Tony Kreager, Senior Vice President, Data Engineering & Data Science, FedEx Datawork

Mercedes-Benz R&D creates 'container-driven cars' powered by Microsoft Azure

"The Azure ecosystem gave us useful capabilities that the developers worked into their system, such as Azure Container Registry tasks, AKS integration, alerting, monitoring, and logging. Plus, Azure Pipelines made it possible to handle deployment and other tasks in a repeatable way, giving the platform team centralized control and standards enforcement."

-- Rodrigo Nunes, Principal Software Engineer, Mercedes-Benz R&D

We rely on Java. And use it. A lot.

Microsoft uses Java extensively. We're using Java to run significant parts of Microsoft—and empower thousands of customers to do the same. After the acquisition of jClarity in 2019, we kicked off a dedicated effort to optimize internal Java workloads and contribute our learnings to OpenJDK.

Today, we now have more than 2 million JVMs running in production on internal systems across the company. We collaborate closely across various businesses at Microsoft, sharing best practices, lessons learned, and expertise running Java in enterprise scenarios and funneling insights back into our customer-facing offerings as well as our contributions to the broader Java community.

Bing

People are often surprised to hear that Bing—Microsoft's web search engine, which also powers the search feature in the Windows Start menu—uses Java. In 2016, with machine learning (ML) and deep learning (DL) being incorporated into the Bing search stack, the offline compute team at Bing began building a platform that could support both big data processing and ML/DL workloads.

Today, that offline compute platform is based on Apache Kafka and Apache Hadoop, with data processing driven by Apache Spark jobs running in JVMs. Most of the environment is currently based on Java 8, with the remainder running Java 11. The platform is deployed across multiple Bing datacenters, spanning hundreds of thousands of JVMs on more than 50,000 servers. About 40 percent of the JVMs run on bare metal, while the other 60 percent run on Azure VMs. Cluster management is based on Yarn, with HDFS used for distributed file storage.

The Bing engineering team has hundreds of Java developers, with about half using IntelliJ and the other half using Visual Studio Code. Today, across Bing, more and more "old" big data processing is being migrated to the new platform, with both batch and stream processing running on Java VMs.

If you'd like to learn more about Bing's use of technology, check out the Bing Engineering blog.

Azure Infrastructure: Control Plane

When we tell customers that while provisioning resources, most of the interactions that they have with Azure go through a JVM, they're caught by surprise. This is due to Azure Control Plane operations within the Azure infrastructure, which are used to manage the resources within an Azure subscription—such as the creation of a virtual machine, storage account, or database. Two foundational services within the control plane include a pub/sub messaging service built on Apache Kafka and a leader election/configuration management service built on Apache ZooKeeper. These control plane services run on bare-metal servers in every Azure datacenter, of which there are more than 160 today.

Both services are written in Java and Scala, and run on the JVM. The pub/sub service built on Kafka uses the Microsoft Build of OpenJDK 11 today, and the leader election/configuration management service based on Zookeeper uses Eclipse Temurin 8. After tuning exercises performed by our Java Engineering Group and the Control Plane teams, they have opted to use the G1GC garbage collector for both services.

Across all Azure datacenters worldwide, these Kafka clusters for the pub/sub service in Azure infrastructure produce more than 500 billion messages and more than 200 terabytes of data every day.

LinkedIn

When Microsoft acquired LinkedIn in 2016, it also acquired a huge Java shop. More than 46 percent of all LinkedIn source code is written in Java, with more than 43 million lines of Java code contained in more than 400,000 files. About 60 percent of all LinkedIn engineers are Java developers, using Java tools such as Eclipse, IntelliJ, and Visual Studio Code.

Almost all production services at LinkedIn run within a Java Virtual Machine (JVM). Altogether, the company's three production datacenters host hundreds of thousands of JVMs, with some 60 percent running Java 11 while the rest run Java 8. Today, close to 90 percent of all JVMs are running the Microsoft Build of OpenJDK, which is discussed in detail later in this eBook. Those JVMs run a range of server technologies, including Jetty (~55%), Play Framework (~25%), and Apache Samza (~20%). LinkedIn also employs several garbage collectors, including G1 (~55%), CMS (~30%), Parallel GC (~5%), and others.

In the past, LinkedIn had a monolithic architecture. Today, it's fully microservices-based, with more than 2,000 Java microservices—a number that's growing every day. Key microservices within the company's architecture include:

- A service that drives the "People you may know" feature of LinkedIn, which has the largest heap across all back-end services at LinkedIn, at 1.2 TB.
- A service for the LinkedIn member-to-member blocking feature, which supports more than 2,000 queries per second per instance. It's one of the lowest tail-latency services across all back-end services at LinkedIn, at 99.9 percent < 1 millisecond.

- A routing service-layer for LinkedIn's own distributed, fault-tolerant NoSQL database, called Espresso. Its 4,500 production instances support a total of more than 22 million queries per second.

Over the years, LinkedIn has built several in-house Java solutions to drive its data pipeline, many of which are open-source and are used by Java developers worldwide. Apache Kafka, LinkedIn's pub/sub messaging system, is one such example, with the company's Kafka cluster producing more than 5 trillion messages and handling more than 5 petabytes of data per day. Apache Samza, the company's distributed stream processing engine, processes over 1 trillion messages per day. Apache Pinot—a distributed, real-time OLAP data store—and Espresso were also developed at LinkedIn.

If you'd like to learn more about LinkedIn's use of Java, check out the LinkedIn Engineering website.

Yammer

Yammer, an enterprise social networking platform, was acquired by Microsoft in 2012 and is now a fully integrated part of Microsoft Teams. The Yammer service is largely Java-based; some parts are written in Ruby, and the Yammer team is looking at adopting JRuby. The Yammer team has hundreds of Java developers, who use a mix of IntelliJ and Visual Studio Code.

Yammer is powered by some 70 different microservices, which run on thousands of JVMs hosted on Linux machines. Most of those JVMs are based on Eclipse Temurin (formerly AdoptOpenJDK) 8 and 11, with plans to move to the Microsoft Build of OpenJDK 17 soon. Elasticsearch clusters run on Ubuntu's OpenJDK 8. Dropwizard—an open-source framework for rapid development of REST APIs—was developed in-house at Yammer and is still deployed widely.

Other open-source projects in use at Yammer include:

- Apache Kafka for asynchronous (pub/sub) inter-service communication
- Apache HBase and Apache Hadoop (hosted on Azure Databricks) for big data
- Apache Superset for visualization
- Apache ZooKeeper for consensus management across service mesh and PostgreSQL (master/read replica) cluster

Altogether, the Yammer service supports up to 100,000 requests per second.

Minecraft Java Edition

Although Minecraft has been ported to many platforms, the original version of the game—now called Minecraft: Java Edition—remains immensely popular. As the name suggests, it was written in Java and remains a Java desktop application to this day. Today, Minecraft: Java Edition bundles the Microsoft Build of OpenJDK 17 and reaches millions of gamers around the world.

Minecraft Realms, which are personal, Microsoft-hosted multiplayer Minecraft servers, are also based on Java—including both the personal servers themselves and the backend infrastructure that supports them. The personal servers reside within ephemeral VMs, which are Docker containers that wrap the Minecraft Realms Java service and the dedicated game server. Minecraft Realms runs more than 100,000 ephemeral VMs per day via Azure PlayFab, a fully managed platform for building and operating live games.

Within each ephemeral VM, the Java service handles commands and input from the Minecraft client, communication with the dedicated Minecraft game server, and communication with backend Minecraft Realms services for status, payments, and so on. There are typically 50-75 instances of such backend services, which are hosted on Azure VMs running Ubuntu Linux.

Other services that support the backend infrastructure include Azure Service Bus for messaging, Azure Blob Storage for storing hundreds of petabytes of large files, Geneva and Kusto for logging, and Grizzly for HTTP services. MySQL is used for all database needs, although the Minecraft Realms team is considering a move to Azure Table Storage and Azure Cosmos DB.

Committed to continuous improvement

Committing to developer and customer success is a continuous process, one that we take very seriously. Some of our key focus areas include:

- Listening to our customers who use Java—both external and internal—and delivering what they tell us they need.
- Packaging the best practices and lessons learned from our efforts into an easily accessible and consumable knowledge base designed for developer success.
- Working with the thousands of Java developers and Java champions across our engineering and product teams toward improving our Java products.
- Contributing to OpenJDK and engaging with the Java community through the Eclipse Foundation, Eclipse Adoptium, Jakarta EE and MicroProfile Working Groups, Java Community Process, and the industry partnerships we've formed.
- Actively sponsoring, attending, and contributing to relevant industry events and conferences.
- Recruiting passionate, experienced Java professionals to join our product and engineering teams.

"Microsoft's commitment to customer success starts with understanding what our customers need. Through extensive customer engagement, we developed services specifically for customers with Java-based workloads, such as Azure Spring Apps for Spring Boot applications, a managed service offering for Red Hat JBoss EAP. To address customer requests for more idiomatic and native Java support, we have incorporated that support across our entire portfolio: Azure services, SQL Server, Big Data, and developer tools."

-- Scott Hunter, VP of Engineering Product Management at Microsoft

Talk to us

If you'd like to learn more about Java development at Microsoft our Java documentation home page is a good starting point. If you're interested in diving deeper into some of the subjects we've covered in this eBook, below are some helpful sources of information:

Blog articles, forums, and other ad-hoc engagement

Beyond the formal training and documentation venues described above, we also try to keep Java developers informed on what we're doing in more informal, ad-hoc ways. Blog articles are a good example, with the Java at Microsoft blog, Tech Community Apps on Azure blog, and Azure Blog providing news, updates, and insights for Java development across Microsoft tools, Azure services, and OpenJDK. We're also on Twitter at @JavaAtMicrosoft.

Training and technical enablement

We recently added a Get Started guide on Java on Azure on our online learning platform, Microsoft Learn. We also recently launched a Java for Beginners channel on YouTube, where Microsoft employees from around the world are sharing their knowledge of Java and all you can do with it.

We have formal programs to help customers deploy Java on Azure, as well. One such program is FastTrack for Azure, which customers can use to move their Java apps to Azure quickly and efficiently, with customized guidance on architectures and best practices from our own engineers. FastTrack engagements are free to customers with eligible Azure subscriptions, and typically last a few weeks.

We'd love to hear your feedback. Please reach out to us via Twitter: @JavaAtMicrosoft.

For more information on Java development visit developer.microsoft.com/java