Code, Deploy, and Scale Java Your Way

EMPOWERED JAVA APPLICATION DEVELOPMENT IN THE CLOUD

By Asir Vedamuthu Selvasingh, Principal Architect for Java on Microsoft Azure

About
the author

Asir Selvasingh is a Principal Architect for Java on Microsoft Azure, on-point for everything developers and customers need to build, migrate, and scale Java applications on Azure. Asir started his software engineering career in the early days of Java - in 1995 with JDK 1, and built enterprise products, applications, and open-source projects for many years. He joined Microsoft in 2005 after several years as a Java Architect at WebMethods. He has been leading the Java on Azure effort at Microsoft since its inception in 2009. He works closely with customer technical decision makers, business decision makers, and C-suite executives on strategies for application modernization and increasing developer velocity. Asir also works closely with the Java community, delivering sessions at Java conferences and fostering strategic relations that enrich the Java ecosystem. He loves to hear from customers and developers and can be reached on LinkedIn and Twitter.

Foreword

A chance meeting at the SpringOne conference in 2016 has led to one of the most important collaborations between two companies. After I met Asir at SpringOne in 2016, we both knew that we had to collaborate and innovate so Java developers could deploy their Spring applications on Azure with ease and confidence. Our respective teams have worked together on several Java efforts - from IDE integrations into Spring Tool Suite and VS Code, to Spring Boot Starters enabling seamless access to Azure services. And I am happy to report that today Azure is the best place for Java developers to deploy Spring applications.

The culmination of our efforts, Azure Spring Apps is a fully managed service for Spring Boot applications that lets you focus on building and running apps that run your business without the hassle of managing infrastructure. This idea has resonated with developers, and I am proud to say that today, many enterprise customers are running their mission critical production systems on Azure Spring Apps. We continue to listen to feedback and innovate, making it easier and quicker to get applications into production.

I have witnessed Microsoft's commitment to the Java ecosystem from the first row consistently for many years now. The company has advanced Java technologies on Azure to deliver the best outcomes for Java developers and customers. They offer an impressive breadth of supported technologies, managed offerings, and partner offerings. They are committed to supporting and contributing to the broader Java community as an active contributor to the OpenJDK, JCP, Eclipse Adoptium, and Jakarta EE.

Today, more and more Java developers are looking at how they can bring their existing Java applications to the cloud – or at how to build new cloud-native applications.

This e-book covers the entire journey for developers and operators to code, deploy, and scale with confidence.

It is a must read for every Java developer.

Best,

Ryan Morgan
Vice President, Software Engineering, VMware

Code, Deploy, and Scale Java Your Way

Introduction

Today, more and more Java developers are looking at how they can bring their existing Java applications to the cloud—or at how to build new cloud-native applications. In doing so, they want to know they'll be able to:

- Continue writing code using the Java tools and frameworks they already know and love.
- Deploy their Java applications using their preferred application servers and open-source software.
- Scale their Java applications easily and confidently including necessities such as security, supporting data and messaging services, caching, monitoring, and automation.

Microsoft Azure supports all these needs. As a company, Microsoft is committed to making Java developers as efficient and productive as possible, empowering them to use any tool, framework, and application server on any operating system. On the following pages, we'll examine how Azure delivers on this commitment, letting Java developers continue working the same way they do today—and continue using the tools and software of their choice—while leveraging the power of managed services in the cloud.

Code using the Java tools you know and love

As Java developers, we love the tools we use. We have our own unique way of working with them that gets us "into the flow" just as we have our shortcuts and secrets for getting things done faster and better. Whether we use IntelliJ, Eclipse, or VS Code for coding, or Junit for testing, or Maven or Gradle for dependency management and build automation, there is nothing that can compel us to toss aside our go-to tools and learn something new. That's why Azure empowers Java developers to bring their applications to the cloud on your favorite tools and frameworks and on the operating system of your choice. Let's take a closer look at some of these tools.

IDEs – VS Code, IntelliJ, and Eclipse

An ideal IDE includes tools for editing source-code, compilation, local build automation, testing, and debugging—along with controls and monitoring tools for backend services for data management, caching, messaging, and eventing. An integrated toolset that supports all these tasks makes developers more productive, enabling them to avoid having to learn and constantly switch between standalone tools for each task. IntelliJ, Eclipse, and Visual Studio Code are the popular Java IDEs.

Java on Visual Studio Code

Visual Studio Code (VS Code) is a lightweight, agnostic operating system that runs on Windows, macOS, and Linux. A powerful IDE, it provides a comprehensive toolset for Java development. It supports any Java Development Kit (JDK), including the Microsoft Build of OpenJDK, Amazon Corretto, Eclipse Adoptium, and Oracle Java SE. VS Code also integrates well with all Java frameworks, application servers, and other popular tools, including Tomcat, Spring Boot, JBoss EAP, WildFly, Quarkus, Open Liberty, Maven, and Gradle. It also supports other programming languages that are frequently used by Java developers—like JavaScript and SQL.
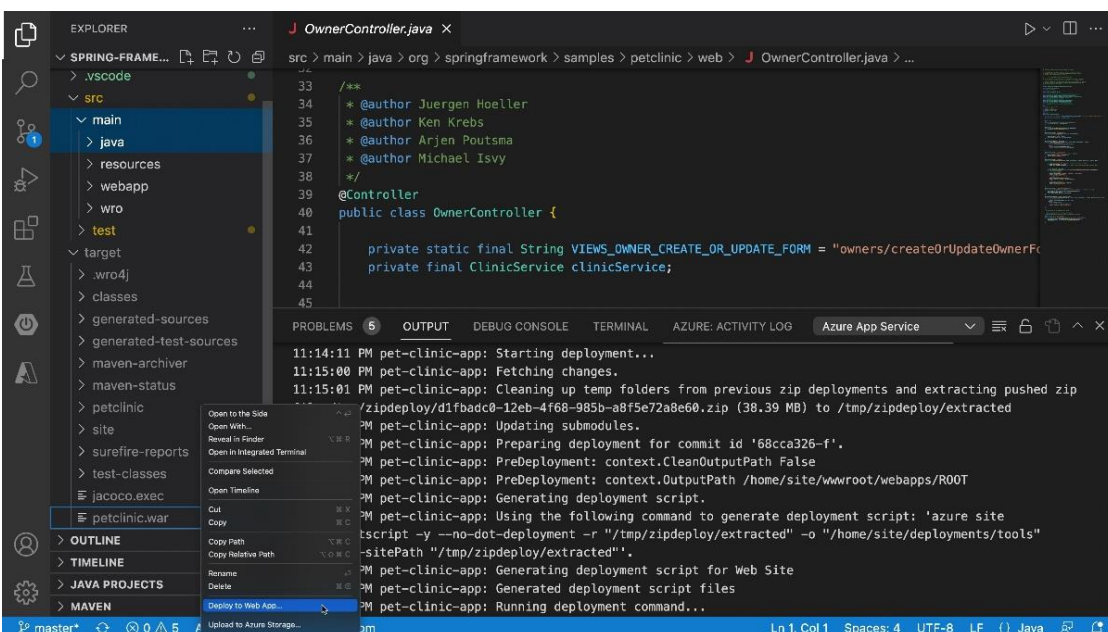
VS Code supports and streamlines Java development workflows through a broad range of extensions. There are several hundred for Java alone, which you can search for from within the IDE itself. We've packaged key extensions for fundamental Java development into the Extension Pack for Java, including those for project management, Maven integration, code editing, code completion, code navigation, refactoring, linting, formatting, debugging, running and debugging Junit/TestNG test cases, and more. There's also a Spring Boot Extension Pack for developing and deploying Spring Boot applications—including Spring Initializr Support for integration with Azure Spring Apps, a fully managed service for running Spring Boot applications on Azure.

The Azure Tools Extension Pack, built by Azure engineering teams, provides a rich set of extensions for discovering and interacting with all the Azure cloud services that help power your Java applications—all from within VS Code as you're writing, debugging, and testing your Java app. When you're ready to deploy your app, the Azure Tools Extension Pack supports one-click deployment to the various compute services that Azure provides for running Java applications.

Java in Visual Studio Code provides a good overview of the most popular Visual Studio Code extensions for Java development. It also provides instructions for getting started with Java development using Visual Studio Code, along with a walkthrough of the many ways it can help make Java developers more productive.

Getting Started with Java in VS Code provides a short tutorial that covers setting-up VS Code for Java Development, including how to write and run the Hello World program. Similarly, there are short tutorials that show how to build a Java application using Visual Studio Code and then deploy it with a single click into Azure App Service, Azure Spring Apps, Azure Container Apps, and Azure Functions. If you're new to Java on VS Code be sure to try out the "Java: Tips for Beginners" command in its main Command Palette.
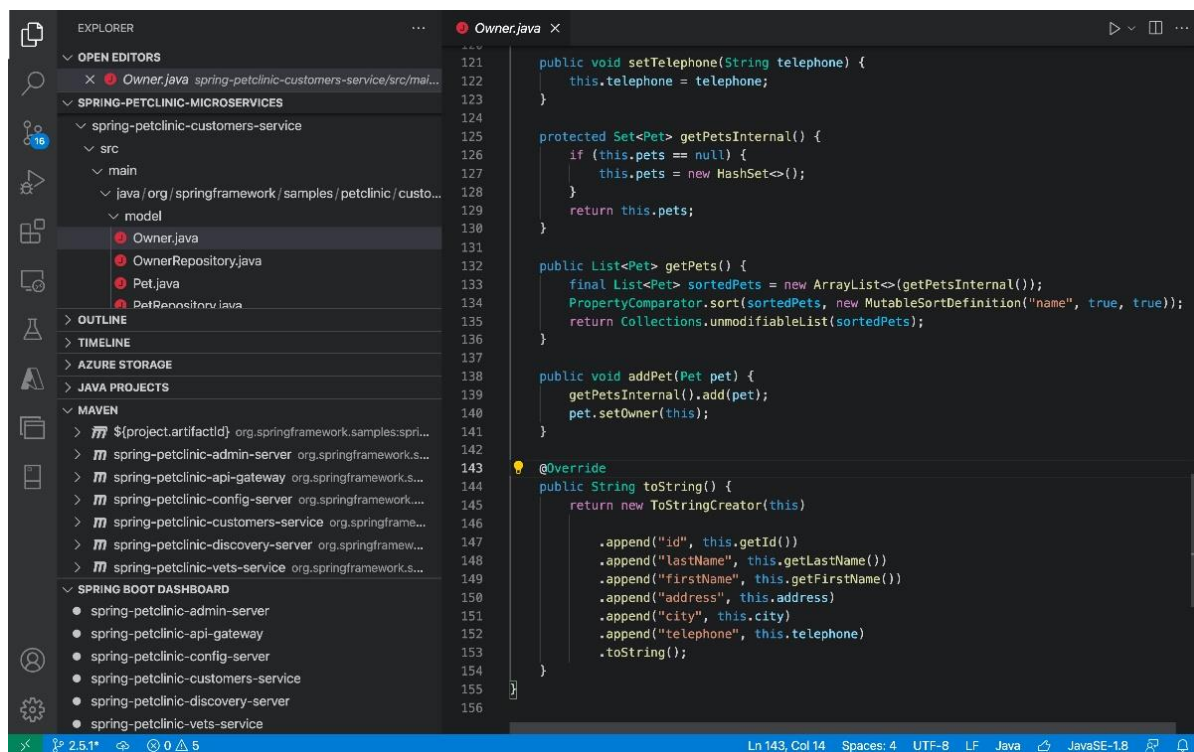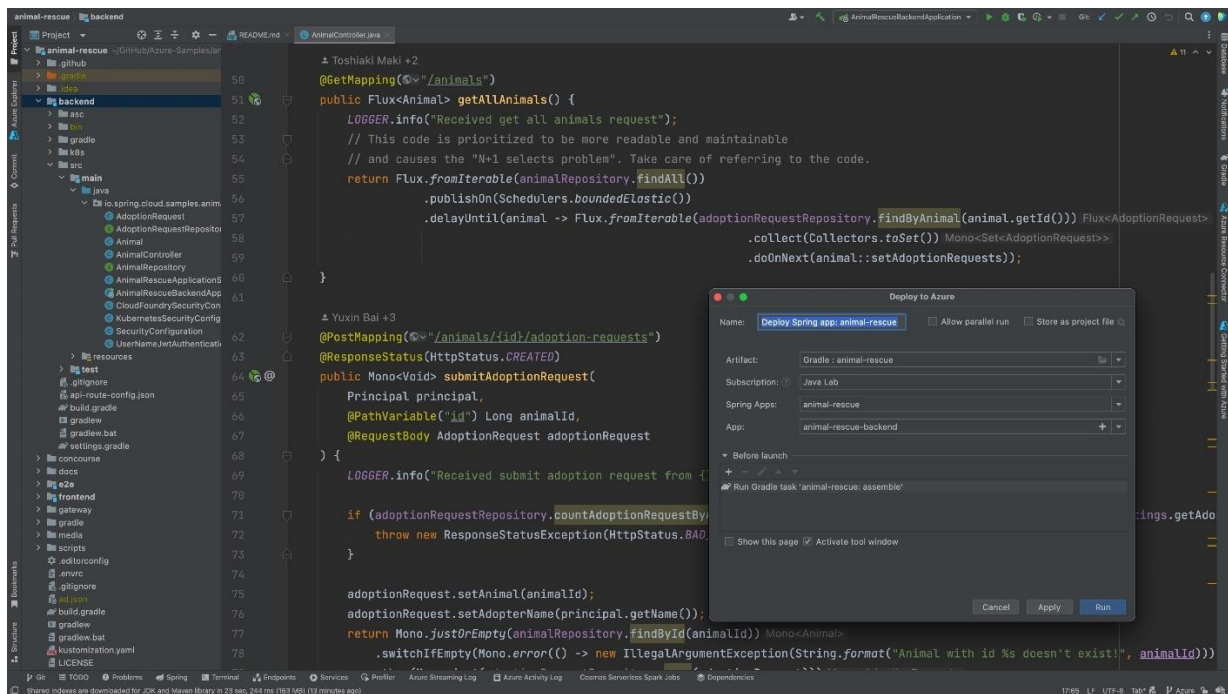
Azure Toolkit for IntelliJ

The Azure Toolkit for IntelliJ lets Java developers create, develop, test, and deploy Java applications to Azure using the IntelliJ IDE. For example, developers can use it to deploy Java Web applications to Azure App Service and custom containers in Azure App Service, deploy Spring Boot applications to Azure Spring Apps, or deploy serverless applications to Azure Functions—all of these are compute services for running Java on Azure, which we'll cover in more detail later in this eBook. Spring Cloud Azure integrations are provided through the Spring Initializr experiences in IntelliJ; simply add the appropriate Java libraries and drivers (including Azure SDK for Java) as dependencies in your Java project.

Microsoft is actively investing time and resources to provide additional functionality for IntelliJ, including new experiences for cloud-native development and deeper integration with Azure services—including integrations with Azure Kubernetes Service and Application Insights.

Azure Toolkit for Eclipse

The Azure Toolkit for Eclipse lets Java developers create, develop, test, and deploy Java applications to Azure using the Eclipse IDE. It includes key Java libraries and drivers, including the Azure SDK for Java. Developers can use the Azure Toolkit for Eclipse to Java Web Apps to Azure App Service and custom containers in App Service, deploy Spring Boot applications to Azure Spring Apps and deploy serverless applications to Azure Functions using Maven or Gradle plugins—all through the Eclipse IDE.

Dependency management and build automation – Maven, Gradle, and GitHub

Maven and Gradle are two popular project management, dependency management, and build automation tools for Java applications. These tools are well-integrated into popular Java IDEs, with one-click deployment to Azure supported through a set of plug-ins for each tool.

Maven Plugins for Azure Services

Maven plugins for Azure services let you extend your Maven development workflows to Azure, testing your Java applications locally and then deploying them to Azure services in a single step—in a way that integrates with Azure authentication methods and Azure Role-Based Access Control. The Maven plugin for Azure App Service helps you deploy Maven Java Web application projects to Azure App Service and to custom containers in App Service; the Maven plugin for Azure Spring Apps helps you deploy Maven Spring Boot application projects to Azure Spring Apps; and the Maven plugin for Azure Functions helps you deploy Maven serverless Java application projects to Azure Functions.
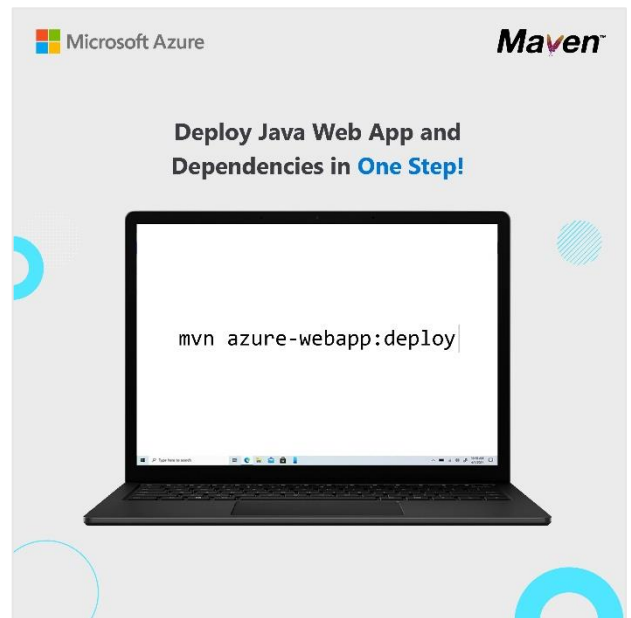
Gradle Plugins for Azure Services

Gradle plugins for Azure services are similar to those for Maven; they let you deploy your Java applications to Azure services in a single step - in a way that integrates with Azure authentication methods and Azure Role-Based Access Control. The Gradle plugin for Azure App Service helps you deploy Gradle Java Web application projects to Azure App Service and to custom containers in App Service, and the Gradle plugin for Azure Functions helps you deploy Gradle serverless Java application projects to Azure Functions.

GitHub

GitHub is a popular repository for Java applications, providing a DevOps environment for more than 3.5 million Java applications. Using GitHub Actions for Java, you can download and setup a requested version of Java; extract and cache a custom version of Java from a local file; configure runners for publishing using Maven, Gradle, or a GPG private key. You can also use GitHub Actions for Java to register problem matchers for error output and to cache dependencies managed by Maven or Gradle.

GitHub Actions makes it easy to automate all your Java software workflow using world-class CI/CD. You can build, test and deploy your code to Azure right from GitHub. Make code reviews, branch management, and issue triaging work the way you want. You can deploy to any of the Azure services for running your Java applications.

GitHub also supports development containers for Java, which you can access via GitHub Codespaces or VS Code Remote – Containers.

Jenkins Pipelines

Azure Pipelines, part of the Azure DevOps service, lets you continuously build, test, and deploy your Java applications to any platform and cloud. It works with GitHub (or Azure Repos) for source control, enabling you to build using Maven or Gradle and then deploy to any of the Azure services for running your Java applications.

Azure Pipelines

Azure Pipelines, part of the Azure DevOps service, lets you continuously build, test, and deploy your Java applications to any platform and cloud. It works with GitHub (or Azure Repos) for source control, enabling you to build using Maven or Gradle and then deploy to any of the Azure services for running your Java applications.

Azure Command Line Interface

The Azure Command-Line Interface (CLI) is a cross-platform command-line tool for creating, connecting to, and managing Azure resources—including the execution of terminal commands via command-line prompts or scripts. You can install the Azure CLI locally on Linux, macOS, or Windows-based machines; run it from within a container; or access the Azure CLI from a browser through Azure Cloud Shell.

```
# Deploy Payment Service
az spring app deploy --name ${PAYMENT_SERVICE_APP} \
    --config-file-pattern payment/default \
    --source-path apps/acme-payment

# Deploy Catalog Service
az spring app deploy --name ${CATALOG_SERVICE_APP} \
    --config-file-pattern catalog/default \
    --source-path apps/acme-catalog
```

(Azure CLI – deploy apps to Azure Spring Apps)

Microsoft believes in and respects your right to choose your own tools. You can build test, debug, and troubleshoot any Java application (including polyglot applications) using the machine of your choice, including Windows, macOS, Linux, and cloud-based machines—and you can deploy your application to Azure on any application server or with any embedded application server.
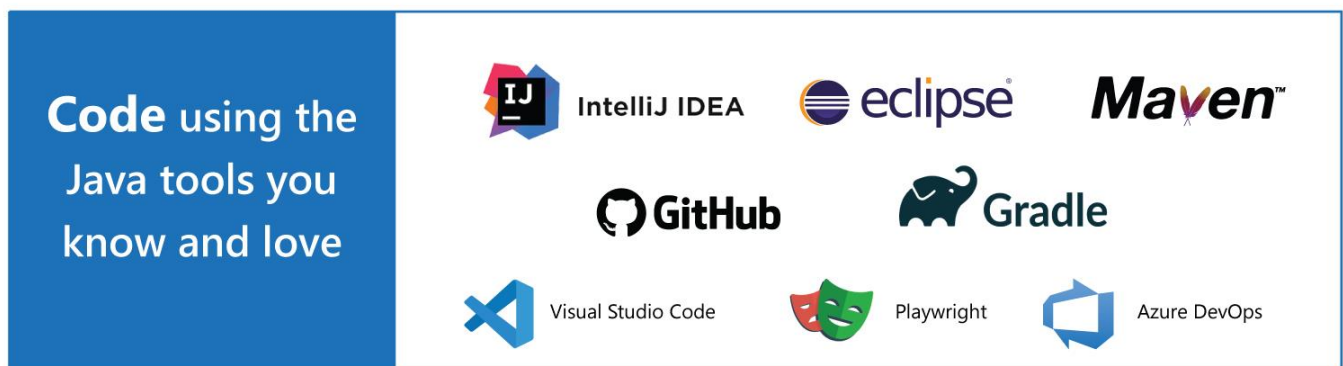
Deploy Java applications with confidence and ease

The Java ecosystem includes diverse technologies such as Java SE, Jakarta EE (successor to Java EE and J2EE), Spring, numerous application servers, and other frameworks. Whatever you're doing with Java—building an app, using a framework, and running an application server—Azure supports your workload with an abundance of choice. Similarly, Azure supports any application architecture—from monolithic applications running on VMs or in containers to cloud-native, microservices-based applications running on fully managed services.

Typically, to run your Java application, you'll deploy it to an application server—an instance of the Java Virtual Machine (JVM) that runs your applications. Or you can build a standalone application with an embedded application server. Either way, the application server provides common application infrastructure and functional capabilities, collaborating with Web containers to return a dynamic, customized response to a client request. The client request can be processed using software components that might include servlets, dynamic pages, enterprise beans, supporting classes, dependent libraries, and data drivers.

Tomcat, JBoss EAP, WildFly, WebLogic, and WebSphere are popular application servers. Similarly, Spring Boot, Quarkus, and Open Liberty are popular frameworks for building standalone applications with embedded application servers. Azure supports them all, enabling you to use any Java application server and deploy your Java application with confidence and ease.

Deploy Spring Boot or Java app to any application server – Tomcat and Jakarta EE

With Azure, you can run any version and any distribution of Java and any application server, without restrictions, and without having to manage your own physical infrastructure. You decide how much control you want, or how much day-to-day management you want Azure to handle for you with options like virtual machines, containers, and fully managed services. If you are using a commercially supported Java app servers or frameworks -- such as VMware Spring Runtime, Red Hat JBoss EAP, Oracle WebLogic Server, or IBM WebSphere, Liberty, or OpenLiberty -- Azure offers jointly developed and supported offerings for all of them.

Deployment options

Azure provides an abundance of deployment options for Java applications, including infrastructure-as-a-service (IaaS), containers-as-a-service (CaaS), and platform-as-a-service (PaaS) hosting services. You can lift-and-shift existing Java applications to virtual machines (VMs), containerize them in multiple ways, or deploy them onto fully managed PaaS services to optimize ease of management, developer and operational productivity, and total cost of ownership.

VMs and containers

You're free to use any distribution and version of Java—and any application server—when you deploy to virtual machines or containers on Azure. The choice is entirely up to you; just remember you'll need to manually configure the infrastructure

and its components. Deployment options that fall into this category include:

- [Azure Virtual Machines](#), which give you the flexibility of virtualization without having to purchase and maintain the physical hardware that runs it. However, you still need to maintain the VM by installing, configuring, and patching the software that runs on it.

- [Azure Virtual Machine Scale Sets](#), which lets you create and manage a group of load-balanced VMs. The number of VM instances can automatically increase or decrease in response to demand or a defined schedule.

- [Azure Kubernetes Service (AKS)](#), which simplifies deploying a managed Kubernetes cluster by handling all of the operational overhead for you—including critical tasks like maintenance and health monitoring. AKS supports elastic provisioning of capacity, including event-driven autoscaling and KEDA triggers.

- [Azure Red Hat OpenShift](#), which provides highly available, fully managed OpenShift clusters on-demand. OpenShift delivers added-value features to complement Kubernetes, making it a turnkey container platform that delivers significantly improved developer and operator experience.

While you can deploy any Java runtime on all these IaaS and CaaS services, we recommend that you use:

- [Microsoft Build of OpenJDK](#) for Java 11 or 17 – the [base container images](#) for which are supplied and maintained by Microsoft.

- [Eclipse Adoptium Temurin](#) for Java 8 – the Java runtime for which is provided by the Eclipse Adoptium project (formerly the OpenJDK project).
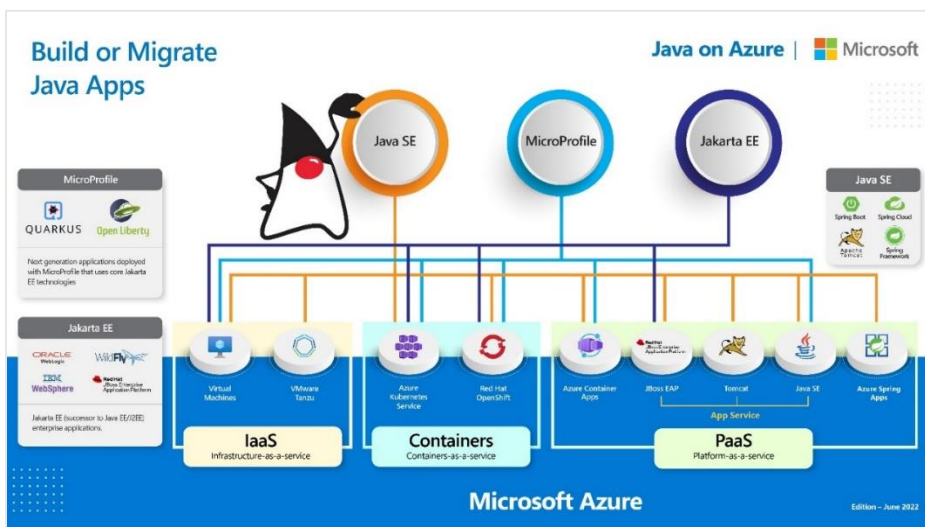
Both of these builds are available free-of-charge for local development and testing, and for building production-ready binaries using any platform or DevOps tools—without having to pay any licensing fees. We recommend them as a matter of convenience; when you use one of these builds, if you have any issues and have a qualifying support plan for Azure, you can simply open an Azure support ticket—without any additional costs. That said, it's worth pointing out that these recommendations are just that—the options that we recommend among various other freely-available builds of OpenJDK for ease-of-support.

All the above IaaS and CaaS deployment options let you easily deploy the Apache Tomcat application server. If you're using a commercial offering—such as Spring Runtime from VMware, JBoss EAP from Red Hat, WebLogic Server from Oracle, or WebSphere from IBM—Azure offers jointly developed and supported hosting options from those vendors as well. They're covered below, under [Jointly built and supported solutions with Java ecosystem partners](#).

Fully managed (PaaS) services

Fully managed PaaS services for running Java applications on Azure include the following:

- Azure Spring Apps, which makes it easy to deploy Spring Boot applications to Azure—without any code changes. The service manages all the infrastructure for Spring Boot applications, including comprehensive monitoring and diagnostics, configuration management, service discovery, security, application lifecycle, publishing logs and metrics, CI/CD integration, blue-green deployments, and more. Developed in partnership with Pivotal (now part of VMware), Azure Spring Apps is jointly operated and supported by Microsoft and VMware.

- Azure App Service, an HTTP-based service for hosting web applications, REST APIs, and mobile back ends—with built-in security, load balancing, autoscaling, and automated management. App Service also supports comprehensive DevOps capabilities, such as continuous deployment, package management, staging environments, custom domains, and TLS/SSL certificates.

- Azure Container Apps, which lets you run microservices and containerized applications on a serverless platform. Common uses include deploying API endpoints, hosting background processing applications, handling event-driven processing, and running microservices. Applications built on Azure Container Apps can dynamically scale based on HTTP traffic, event-driven processing, CPU, or memory load, or any KEDA-supported scaler.

Java runtimes for Azure Spring Apps and Azure App Service are supplied and maintained by Microsoft. They only support LTS distributions of OpenJDK, using Eclipse Adoptium Temurin for Java 8 and the Microsoft Build of OpenJDK for Java 11 and 17. That said, there are some caveats—for example, our jointly developed and supported partner offerings (discussed below) use their own runtimes.

For Azure Container Apps, since you'll need to build and manage your own container images from source code, you're free to use the distribution and version of Java—and application server—of your choice.

Serverless functions

Sometimes you don't need an entire Java application. For example, for real-time data processing, you might just need a small piece of code that can be triggered at scale—perhaps by millions and millions of events. Such events can be ingested via Azure Event Hubs, processed by event-driven serverless Java code running at scale in Azure Functions, and saved into a data store such as Azure Cosmos DB. FedEx, Best Buy, and UBS are great examples of real-time, event driven Java.

Jointly built and supported solutions with Java ecosystem partners

Microsoft has partnered with leading vendors in the Java ecosystem to deliver best-in-class solutions for running

Strong partner ecosystem

Red Hat Enterprise Linux, Ubuntu, debian, SUSE, CoreOS, CentOS	
ORACLE	Solutions for WebLogic on Azure VMs & AKS
IBM	Solutions for WebSphere on Azure VMs, AKS & ARO
elastic	Elastic Cloud with Azure native integration
CONFLUENT	Confluent Cloud with Azure Cache for Redis
redislabs HOME OF REDIS	Redis Enterprise in Azure Cache for Redis
vmware	Jointly built and operated service: Azure Spring Apps
Red Hat	Solutions for JBoss EAP on App Service, VMs & ARO

Java on Azure—ranging from jointly developed and supported managed services to Azure Marketplace offerings for popular Java application servers. We also integrated popular application monitoring tools, which are covered later in this eBook.

Azure Spring Apps (Pivotal/VMware)

Jointly developed by Microsoft and Pivotal, Azure Spring Apps is a fully managed service that solves many of the common challenges developers, IT operators, and DevOps teams face when running Spring Boot applications at scale. It abstracts away the complexity of managing infrastructure for running Spring-based applications and Spring Cloud middleware components, enabling Java developers to focus on their code while letting Azure take care of dynamic scaling, security patches, compliance standards, high availability, and so on.

> *"Azure Spring Apps builds on the rich ecosystems of Microsoft Azure, Spring, and Kubernetes to deliver a turnkey platform optimized for Spring-based applications and services. Spring and Azure Spring Apps let me deliver valuable software without worrying as much about the pager. They get me to production."*
>
> *—Josh Long, Spring Developer Advocate, VMware*

JBoss EAP (Red Hat)

Red Hat is a leading provider of open-source solutions for the enterprise. One such solution is JBoss Enterprise Application Platform (EAP), a popular application server platform that's Java EE Certified and Jakarta EE Compliant in both Web Profile and Full Platform. Red Hat is also a contributor for the Java standards, OpenJDK, MicroProfile, Jakarta EE, and Quarkus.

We partnered with Red Hat to deliver Red Hat JBoss Enterprise Application Platform (EAP) on Azure App Service—enabling Java developers to deploy their Jakarta EE applications into App Service without requiring a separate Red Hat subscription or license with integrated support from both companies. We've since launched similar joint offerings for JBoss EAP on Azure VMs, on Azure VM Scale Sets, and on Azure RedHat OpenShift (ARO)—the latter also jointly operated by Microsoft and Red Hat.

> *"By offering JBoss EAP on Azure, we are combining the best of our areas of expertise and enabling customers to successfully choose how they want to manage applications on the cloud."*
>
> *— Rich Sharples, Senior Director, Product Management, Red Hat*

WebLogic Server (Oracle)

We partnered with Oracle to deliver Oracle WebLogic Server (WLS) on Azure VMs and Oracle WebLogic Server on Azure Kubernetes Service. These solutions facilitate easy migrations to Azure by automating boilerplate operations such as provisioning virtual networks/storage, installing Linux/Java resources, setting up WebLogic Server, and configuring security with a network security group.

"Oracle is delighted to partner with Microsoft to enable deployment of Oracle WebLogic Server applications on Azure Virtual Machines and the Azure Kubernetes Service. Many Oracle WebLogic Server customers want to take their enterprise Java workloads to the cloud, and by collaborating with the Azure team we can give them more choices and better solutions, including the ability to run workloads across Azure and Oracle Cloud."

— Will Lyons, Senior Director of Product Management - Enterprise Cloud Native Java, Oracle

---

WebSphere/Liberty/Open Liberty (IBM)

We partnered with IBM, jointly developing solutions for WebSphere Application Server (WAS) on Azure VMs, WebSphere Liberty and Open Liberty on Azure Kubernetes Service, and WebSphere Liberty and Open Liberty on Azure Red Hat OpenShift. These solutions enable easy migration of WebSphere workloads to Azure, automating most of the resource provisioning tasks needed to set up a highly available WebSphere cluster. The partnership covers a range of use cases—from existing mission-critical workloads to cloud-native applications.

"Together, we're helping eliminate obstacles so [developers] can focus on core tasks: whether it's through the option to modernize your existing Enterprise Java applications and move them to Microsoft's Azure Cloud or deciding on the approach to develop and deploy your next-generation cloud-native application on Azure…. The collaboration between IBM and Microsoft aims to cover a range of use cases, from mission-critical existing traditional workloads to cloud-native applications."

— Willie Tejada, GM ISV/Build Partners & Chief Developer Advocate, IBM

Apache Kafka on Confluent Cloud (Confluent)

In the past, Azure customers who wanted to use Confluent's Kafka service had to create and manage resources and users separately in Azure and Confluent Cloud. To ease this pain, Confluent and Microsoft partnered to deliver Apache Kafka for Confluent Cloud, an Azure Marketplace offering that provides Apache Kafka as a fully managed service—including the ability to create and manage Confluent Cloud resources through the Azure portal, Azure CLI, or Azure Management SDKs.

Today, the customer experience is simpler, safer, and more seamless. Customers can provision and manage Confluent Cloud resources along with their Azure resources, as part of a unified workflow—and take advantage of fully managed connectors built for Azure Functions, Azure Blob Storage, Azure Event Hubs, Azure Data Lake Storage Gen2, and Microsoft SQL Server. Developers can continue to code using Apache Kafka client libraries.

"... I am thrilled to announce a new strategic alliance with Microsoft to enable a seamless, integrated experience between Confluent Cloud and the Azure platform. This represents a significant milestone in our ongoing collaboration with the Microsoft engineering and product teams that deepens our strategic and technical alignment with the Microsoft ecosystem. We have made it even easier for Azure customers to accelerate their journey to the cloud with event streaming, Apache Kafka®, and Confluent as the central nervous system of their business."

— Jay Kreps, Co-founder and CEO, Confluent

Joint development with partners for many of the above offerings is a continual, ongoing effort. As our partners continue to innovate on their offerings, we're working closely with them to quickly bring those same innovations to Azure—so that customers can deploy and scale their Java applications with confidence and ease.

In summary, Azure supports your workload with an abundance of choice regardless of what you're doing with Java. You can build any Java app, use any framework, run any application server, and support any application architecture—from monolithic applications running on VMs or in containers to cloud-native, microservices-based applications running on fully managed services.

Scale with end-to-end security, monitoring, and automation

While designing applications, we need to determine how to adapt to changes in workload, recover from unexpected failures, minimize security risks, and so on. While one could start with a trial-and-error approach, that takes time away from other organizational objectives, and could adversely impact our reputation. Azure provides architectural guidance needed to get things right from the start. You also have everything you need to build a scalable application—from state-of-the-art security and auto-scaling to supporting services for data, messaging, caching, performance monitoring, and automation. Many of these supporting services are based on popular open-source software as well—such as PostgreSQL, Redis, JMS, and Kafka—so you won't get locked into proprietary solutions.

Now let's take a look at some key Azure services and features—and how you can put them to use to build scalable Java applications.
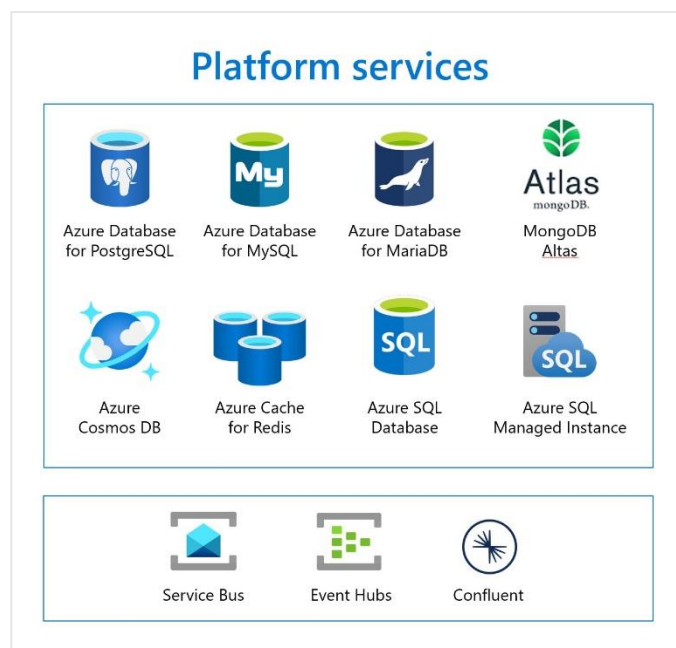
## Extend the capabilities for Java applications - databases and messaging

In addition to providing several options for running your Java code, Azure offers a broad range of fully managed services to support your database needs—including Azure Database for PostgreSQL, Azure Database for MySQL, MongoDB Atlas, Azure Cosmos DB, Azure SQL Database, and Azure SQL Managed Instance. The same holds true for messaging, with options that include Azure Service Bus, Azure Event Hubs, and Apache Kafka for Confluent Cloud. Azure Service Bus Premium tier supports JMS, the Java Messaging Service programming model.

Regardless of whether your applications are running on VMs, in Kubernetes, or on fully managed PaaS services, you can quickly provision and leverage these fully managed data and messaging services using open-source clients, Azure Java SDKs, Spring starters, and application server integrations. They all provide the compliance, availability, and reliability guarantees that you would expect from Microsoft and Azure. Many Java and Spring developers want to use idiomatic libraries to simplify connections to their preferred cloud services. Microsoft maintains a comprehensive list of libraries, drivers, and modules that let you easily interact with Azure services across data, messaging, cache, storage, eventing, directory, and secrets management.
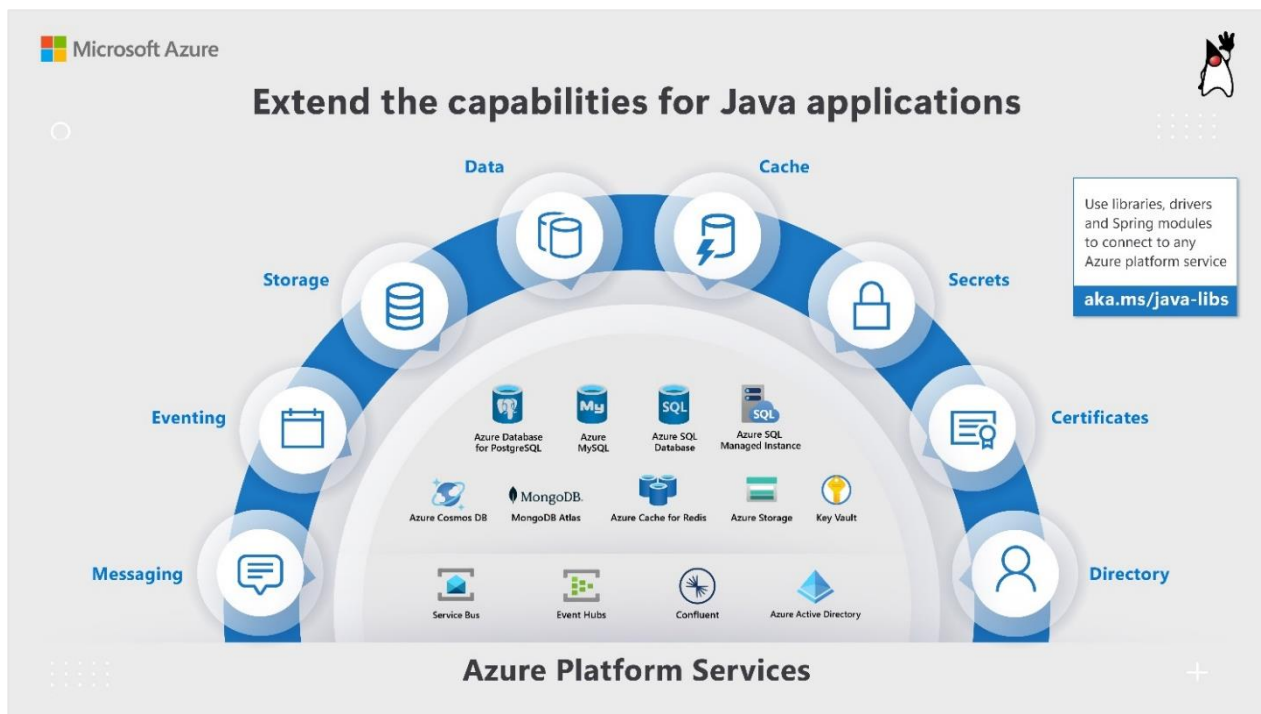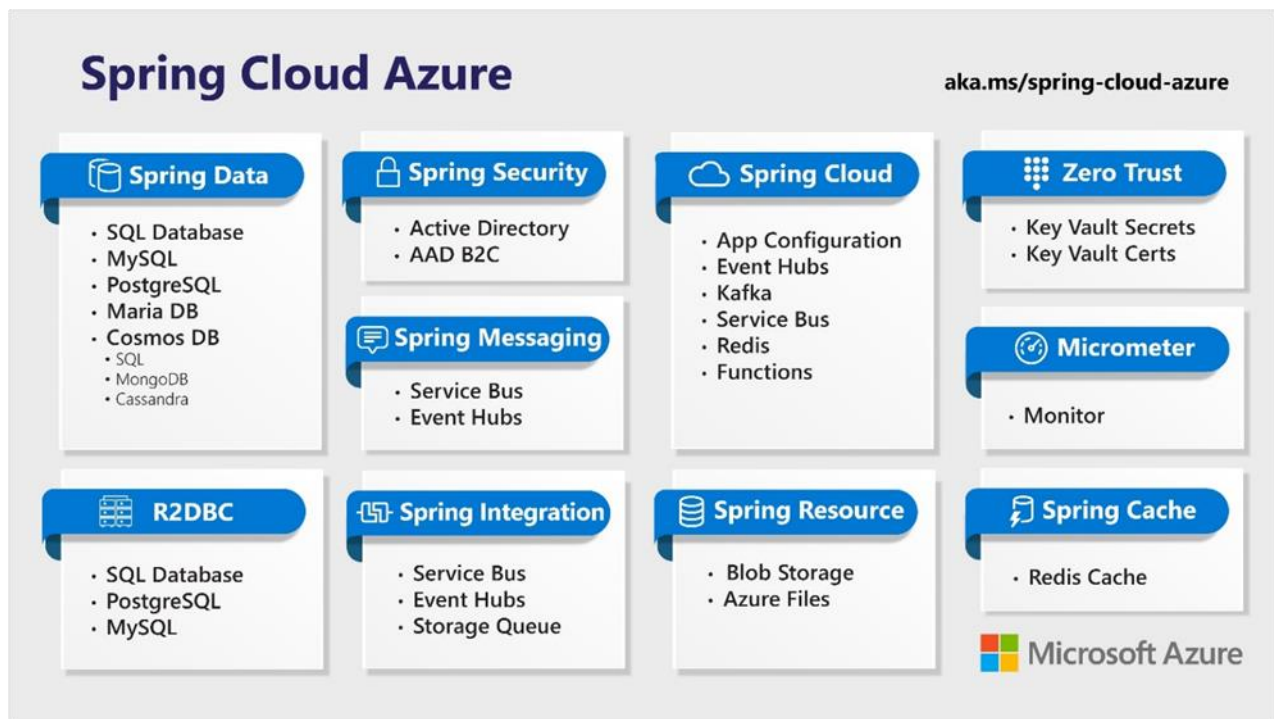
Zero-Trust – Secure network

You can secure your Java applications by deploying them in an Azure Virtual Network (VNET)—the fundamental building block for your own private networks in Azure. VNET enables many types of Azure resources to securely communicate with each other, with the internet, and with your on-premises networks and systems. You can use a VNET to isolate your applications and supporting backend services from the Internet and place them on your private networks. You can assume full control of ingress and egress for your applications and backend systems.
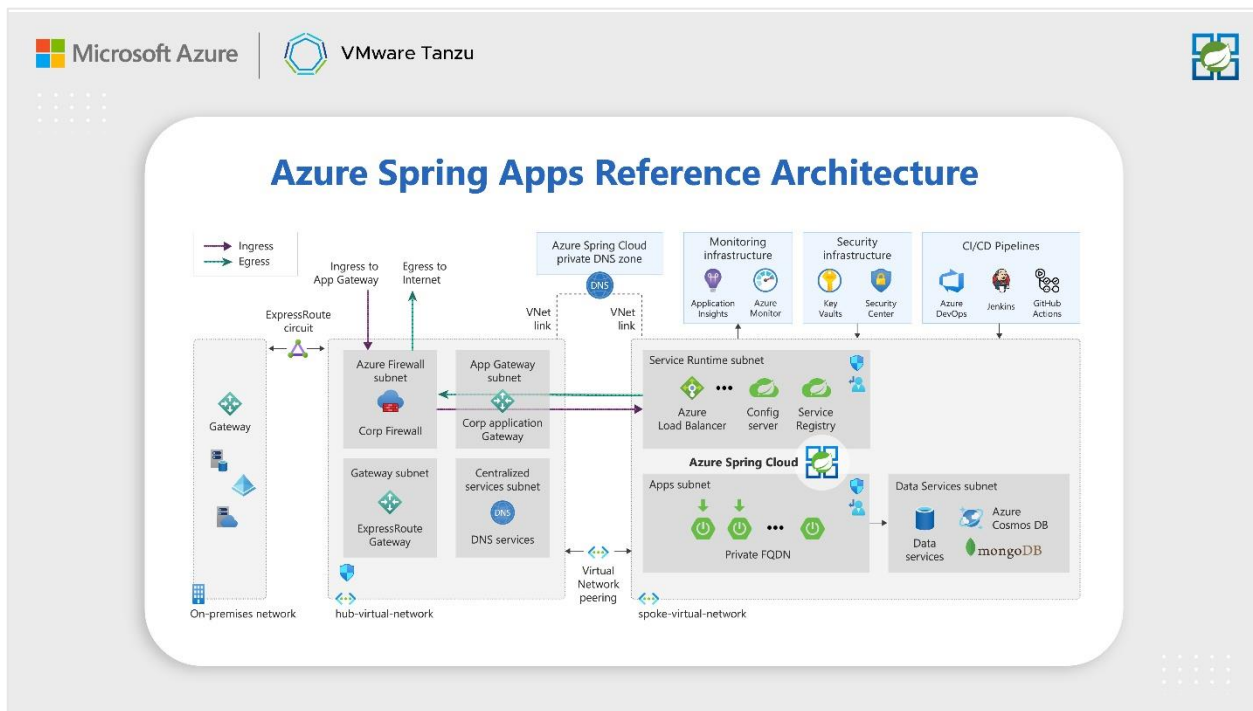
Zero-Trust – Secure communications end-to-end

Implementing secure communications as part of a solution architecture can be challenging. Many companies manually rotate their certificates or build their own solutions to automate provisioning and configuration. Even then, there are still data exfiltration risks, such as unauthorized copying or data transfer.

With Azure, you can secure communications end-to-end or terminate transport-level security at any communication point. You can also automate the provisioning and configuration for all the Azure resources needed for securing communications. This article shows how it works for Azure Spring Apps; it's similar for the other Azure compute services you can use to run your Java applications.

Based on the principle of "never trust, always verify, and credential-free", Zero Trust helps to secure all communications by eliminating unknown and unmanaged certificates, and by only trusting certificates that are shared by verifying identity prior to granting access to those certificates. You can use any type of SSL certificate,

including certificates issued by a certificate authority, extended validation certificates, wildcard certificates with support for any number of sub-domains, or self-signed certificates for development and test environments.

Java or Spring Boot apps can securely load certificates from Azure Key Vault (discussed next). With Azure Key Vault, you control the storage and distribution of certificates to reduce accidental leakage. Applications and services can securely access certificates using managed identities, role-based access control, and the principle of least privilege. This secure loading is powered using the Azure Key Vault JCA (Java Cryptography Architecture) Provider.

"Implementing end-to-end encryption and Zero Trust has been at the top of the list of security requirements for our new API platform. Neither requirement was ever achievable on our old platform. Azure Spring Apps, and its built-in integrations with services like Azure Key Vault and Managed Identities, will finally help us to meet those requirements in an easily automated and manageable way."

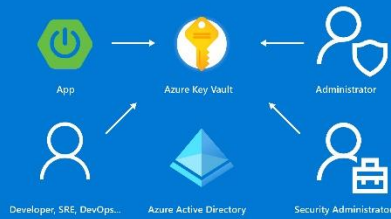*– Claus Lund, Senior Infrastructure Engineering Lead, National Life Group*
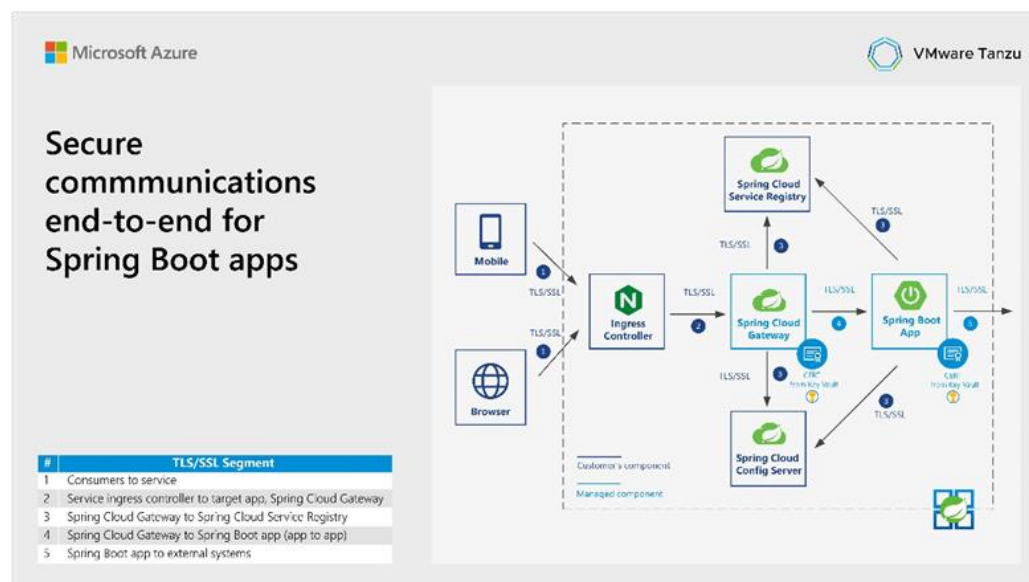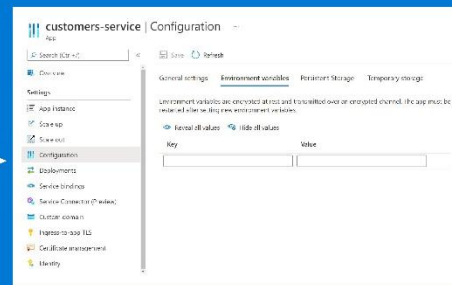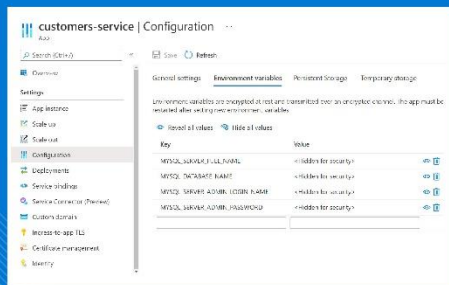

Zero-Trust – Manage secrets

Many Java applications connect to supporting services using URLs and credentials—information that, if exposed, could be used to gain unauthorized access to sensitive data. Embedding such information in an app itself presents a huge security risk for many reasons, including discovery via a code repository. Many developers externalize such credentials using environment variables, so that multiple applications can load them, but this only shifts the risk from the code itself to the execution environment.

Azure Key Vault provides a better, safer, and more secure way to safeguard secrets. It gives you full control over the storage and distribution of application secrets, using Role Based Access Control (RBAC) and the principle of least privilege to limit access. You keep control over your application secrets —simply grant permission for your applications to use them as needed. Upon application startup, prior to granting access to secrets, the application authenticates with Azure Active Directory and Azure Key Vault authorizes using Azure RBAC. Azure Key Vault includes full audit capabilities and has two service tiers: Standard, which encrypts with a software key, and a Premium tier, which includes hardware security module (HSM)-protected keys.

End-user authentication and authorization

Most enterprise Java applications require user authentication and authorization, which you can implement using Azure Active Directory—a complete identity and access management solution with integrated security.

End-user accounts can be organizational identities or social identities from Facebook, Twitter, or Gmail using Azure Active Directory and AAD B2C. You can implement Azure Active Directory based solutions using the Microsoft Authentication Library for Java or Spring Boot Starter for Azure Active Directory. You can also use any identity provider of your choice-- such as ForgeRock, Auth0, Ping, or Okta.

Monitor end-to-end

With Azure, you can monitor your Java applications end-to-end, using any tool and platform. Alternately, you can implement fully-managed, native monitoring—including application performance monitoring (APM)—by using Application Insights, a feature of Azure Monitor. It provides strong support for Java, Spring, and frameworks like Micrometer and Spring Boot, enabling you to quickly identify and troubleshoot issues. Features include live metrics streaming, request rate and response time tracking, event tracing, and external dependency rates—everything you need to monitor the availability, performance, reliability, and usage of your Java applications running on Azure or on-premises.

You can monitor end-to-end by aggregating logs and metrics in Log Analytics, a tool in the Azure portal, which can be used to edit and run queries on logs and metrics data in Azure Monitor. You can write a simple query that returns a set of records and then use Log Analytics to sort, filter, and analyze them. Or you can write a more advanced query to perform statistical analysis and visualize the results in a chart, as may be needed to identify a particular trend. Whether you work with the results of your queries interactively or use them with other Azure Monitor features such as log query alerts or workbooks, Log Analytics is a good tool to use for writing and testing your queries.

That said, we realize that customers who are bringing their Java applications to Azure may want to continue using the same APM tools they're using to monitor their on-premises applications. To support this, we partnered with New Relic, AppDynamics, Dynatrace, and Elastic to integrate their monitoring solutions with Azure App Service and Azure Spring Apps. Monitoring agents run side-by-side with your code, and we'll install and keep the agents updated for you. When you deploy to Azure Container Apps, Azure Kubernetes Service, or Virtual Machines, you can run any of these agents (including New Relic, AppDynamics, Dynatrace, Elastic and Datadog) alongside your applications, but you'll need to install and manage them on your own. Likewise, you can monitor end-to-end by aggregating logs and metrics in Elastic and Splunk.
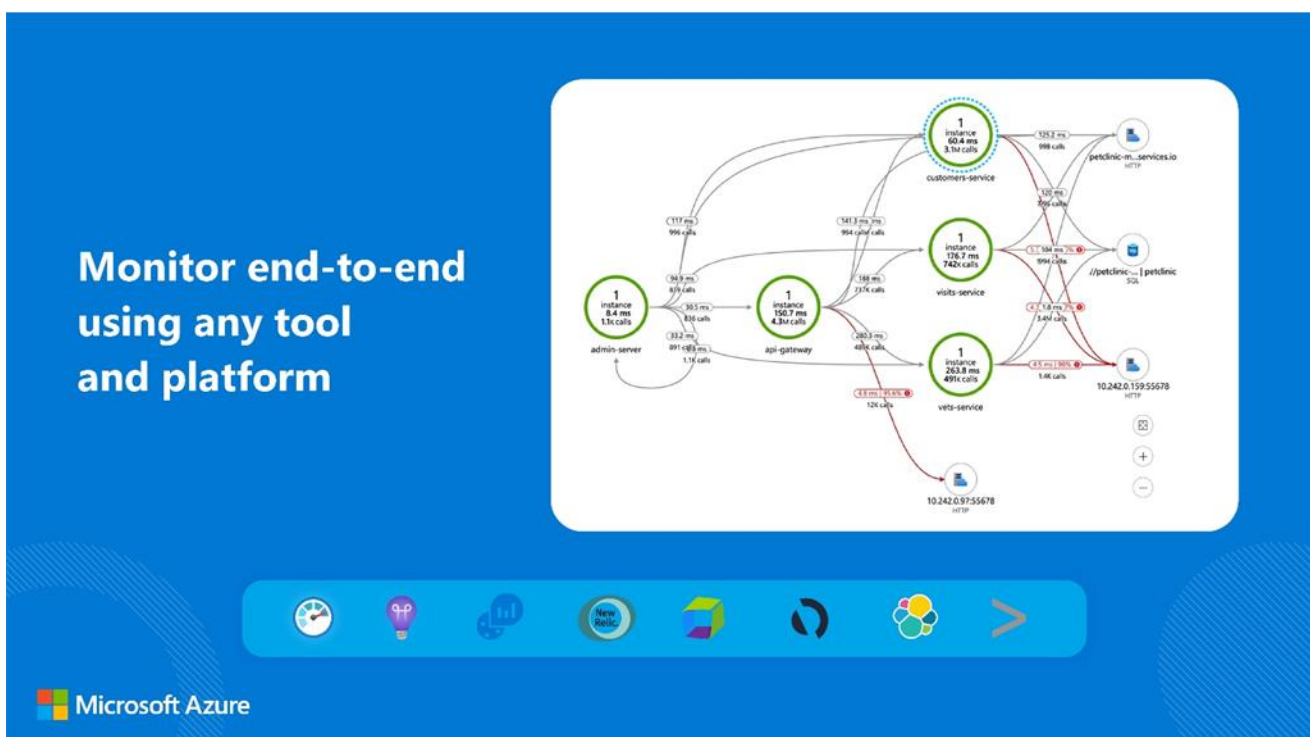
"We are proud of our strategic collaboration with Microsoft. This is reflected in the deep integrations between Dynatrace and Azure, which enable our customers to drive innovation faster, more efficiently, and with greater impact. Our newest integration for Azure Spring Apps is a great example. This will help our joint customers accelerate their digital transformation journeys, with unparalleled observability, application security, and AIOps."

— Eric Horsman, Global Director, Strategic Alliances, Dynatrace

"With Azure's integration, you can use New Relic One to visualize your data and troubleshoot critical issues all on one, unified platform. You can easily combine your Spring Boot app data with events, traces, and log data from other entities across your software stack. This fully connected view makes troubleshooting easier and provides shared context across for teams so that your applications stay available, reliable, and fast."

— Rachel Siemens, Principal Product Marketing Manager, New Relic

"I'm excited by our continued partnership with Microsoft. AppDynamics will provide developers working within Azure Spring Apps with real-time visibility and correlated insights that enable them to isolate the root cause of any performance issues and optimize microservices with context to the business impact."

— Gregg Ostrowski, Executive CTO, AppDynamics

We also realize that many customers want to continue using Grafana to query, visualize, alert on, and understand their metrics. That's why we partnered with Grafana Labs to deliver [Azure Managed Grafana](#), a fully managed service that lets customers run Grafana natively on Azure. The service makes it easy to deploy secure and scalable Grafana instances and connect them to open-source, cloud, and third-party data sources for visualization and analysis. It's optimized for Azure-native data sources like Azure Monitor and Azure Data Explorer, and it includes application performance monitoring (APM) integrations with Azure compute services like Azure App Service, Azure Spring Apps, Azure Kubernetes Service, Splunk, Datadog, and Azure Virtual Machines.

"At Grafana Labs, we don't believe in a 'one size fits all' approach to observability deployment - we want our customers to be able to deploy Grafana where it makes the most sense for their infrastructure, whether that's on a local server or in a public cloud platform like Microsoft Azure. Through our strategic partnership with Microsoft to bring Grafana directly to the Azure cloud platform, we're giving millions of users instant access to the gold standard for monitoring and visualizing their cloud data, and the ability to upgrade at any point to integrate with even more data sources."

— Raj Dutt, CEO and co-founder, Grafana Labs

Accelerate Java applications using caching

As the workloads for your Java applications grow, you can increase performance by using [Azure Cache for Redis](#) to implement an in-memory caching layer for query results, session states, and static content. It's a great way to improve application throughput and reduce latency without

Accelerate and Scale Java apps with Redis Cache

Web App (servlet) — Web App (servlet) — Spring Boot App — Spring Boot App — Spring Cloud Gateway

Fully managed Azure Cache for Redis

Accelerate apps using distributed app cache | Externalize HTTP sessions | Search

Microsoft Azure

having to rearchitect your underlying database. Azure Cache for Redis Enterprise tiers, developed in partnership with Redis and fully managed by Microsoft, is the most highly available and scalable deployment option for running Redis on Azure—including features such as active geo-replication, externalized session management, and high-speed search and indexing.
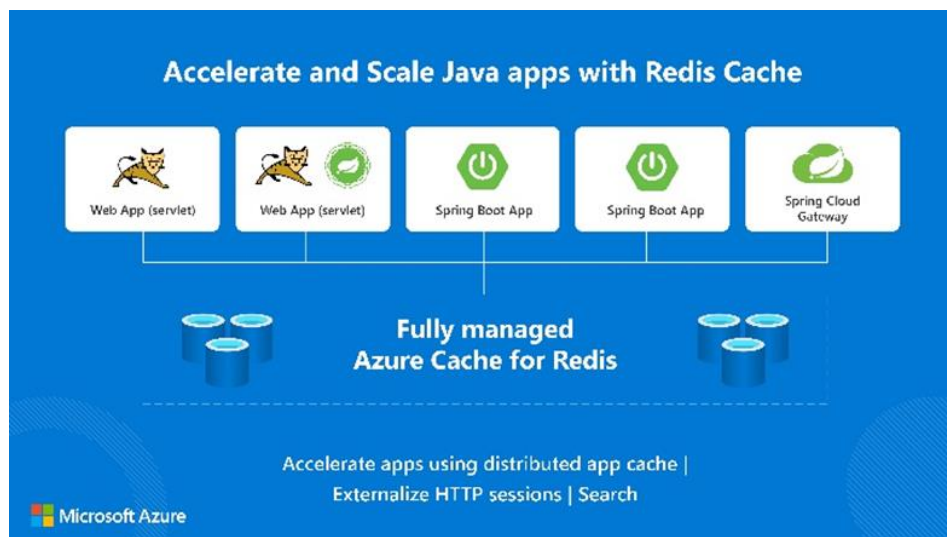
"Companies can now effortlessly incorporate the performance and reliability of Redis Enterprise with the breadth and simplicity of consumption Azure offers to serve a variety of low-latency use cases. This unique service enables customers confidently operate Redis at scale with five-nines availability and active geo-redundancy, expanded use cases with Redis modules, and operate at a very attractive cost on Azure."

— Ofer Bengal, CEO and Co-Founder, Redis Labs

Automatic scaling

All Azure "compute" services for running Java applications support automatic scaling (auto-scaling), which can help you maximize cost-efficiency and adapt to changing workloads without paying for more capacity than needed. Once enabled, you can rest assured that auto-scale will take care of your underlying infrastructure and your application workloads.

You can automatically scale in or out based on load or schedule. In load-based (or metric-based) mode, your applications are horizontally scaled-out to the resources needed to handle the load, up to the limits that you set. Similarly, when load decreases, resources are horizontally scaled-in, never falling below the minimums that you set.

In schedule-based mode, your applications are scaled-in and scaled-out based on a defined schedule and limits. Schedule-based mode is useful for workloads that follow a predictable pattern and can be used to establish a baseline for additional load-based scaling.

Automation from idea to production

As you move your applications to the cloud, you'll want to automate everything—as needed for Java development at enterprise scale. Of course, you'll need to consider auto-scaling to address application workloads, as covered above. But you'll also need to scale and

automate your cloud environment as a whole—ideally from idea to production—including how to rapidly provision of new environments for test, QA, production, blue/green deployments, geographic expansion, and so on.

Azure lets you automate from idea to production using a broad range of tools and platforms. At a high level, such automation pipelines can be broken down into three categories:

- **Provisioning pipelines** – You can provision Azure resources using Terraform, Azure Resource Manager (ARM) templates, Bicep templates, or the Azure CLI, as needed to create repeatable scripts for consistently spinning-up and spinning-down environments.

- **Build pipelines** – Based on tools such as Maven or Gradle, as discussed earlier in this eBook.

- **Deployment pipelines** – You can use GitHub Actions, Azure Pipelines, Jenkins Pipelines, GitLab Pipelines, or the Azure CLI to automate code deployments, including blue/green deployments that keep critical systems in production as you deploy code updates.

Continue to use existing practices and systems

As you migrate or build and then scale your Java applications on Azure, you can use your existing investments in networking, monitoring, automation, identity providers, on-premises systems, development and build tool, and app libraries. The following table provides some examples.

| Category	Java ecosystem products and services
Networking	F5, Palo Alto, Cloudflare, Checkpoint, Infoblox
Monitoring	New Relic, Dynatrace, AppDynamics, Elastic, Splunk
Automation	GitHub Actions, Azure Pipelines, Jenkins, GitLab
Identity providers	Azure Active Directory, ForgeRock, Auth0, Ping, Okta
On-premises systems	Databases (such as Oracle DB or IBM DB2), messaging (such as IBM MQ or TIBCO EMS), eventing (such as Kafka), directories (such as Active Directory, OpenLDAP, or IBM ID)
Development tools	IntelliJ, VS Code, Eclipse, Spring Tool Suite, Maven, Gradle

Reference architectures

The Azure Architecture Center provides guidance for building solutions on Azure using established patterns and practices, including how to put the above capabilities to use. These reference architectures are based on what we've learned from customer engagements, taking into consideration cost optimization, operational excellence, performance efficiency, reliability, scalability, security, monitoring, smoke-testing, and more. They also address solution design components such as Azure landing zones—environments for hosting your workloads that are pre-provisioned through infrastructure-as-code, as needed to enable Java application migrations and greenfield development at enterprise scale.

For example, here's a reference architecture for Azure Spring Apps, showing how to implement a hub-and-spoke design in which Azure Spring Apps is deployed in a single spoke that's dependent on shared services hosted in the hub. It's built with components to achieve the tenets in the Microsoft Azure Well-Architected Framework. To explore an implementation of this architecture, see the Azure Spring Apps Reference Architecture repository on GitHub. You can apply the same

approach to any Java applications deployed to any Azure "compute" destination—such as Azure App Service, Azure Container Apps, or Azure Kubernetes Service. In addition, if you're looking at migrating existing Java applications to Azure, we've got a comprehensive set of migration guides and recommended strategies.
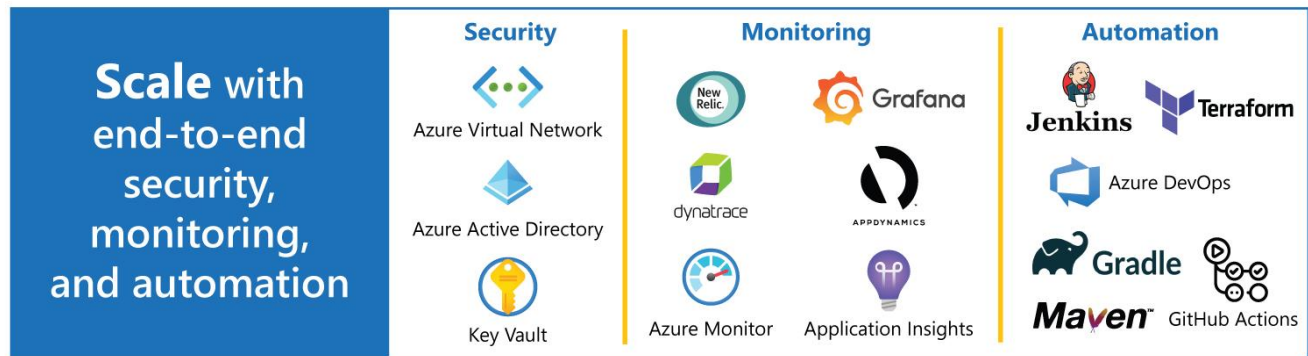
Conclusion

Moving your Java applications to Azure is simple and intuitive, allowing you to benefit from all the cloud has to offer without having to learn new skills or adopt new tools or frameworks. You can continue using familiar tools like IntelliJ, Eclipse, VS Code, GitHub, Maven, and Gradle, and you'll have more time for coding since you won't have to deploy or manage infrastructure.

You can also continue using the same Java application servers and other open-source software you already know and trust. These aren't proprietary Microsoft implementations; rather, they're the "real thing" from trusted names in the open-source ecosystem, like Spring Boot, JBoss EAP, OpenShift, WebLogic, WebSphere, Kafka, Grafana, and Redis.

Azure also provides everything you'll need to scale your applications with confidence and ease, starting with proven reference architectures that are designed for cost control, scalability, high availability, security, and more. You'll also have access to state-of-the art security features, built-in auto-scaling, tools for end-to-end monitoring and automation, and supporting services for data, messaging, and caching.

With Azure, you have everything needed to code, deploy, and scale your Java applications in the cloud—and can start benefiting from all it has to offer without having to change how you work.

Get started with Java on Azure today

If you'd like to learn more about Java on Azure, here are some curated learning paths:

- Get started with Java on Azure.
- Expand the capabilities for Java applications on Azure.
- Best practices for Java applications on Azure.
- Developer.Microsoft.com/Java for Java development.