Java On Azure

Christian Köberl

Global Azure Austria, May 2022

Christian Köberl

🧑‍💻 All-end developer and software gardener.

Chief Technical Architect @ Porsche Informatik.

Develops webapps with Java since 1999.

- You'll find my code at github.com/derkoe
- I tweet at @derkoe
- More talks and information at derkoe.dev

Deploy Apps to Azure

Possible Services

- Virtual Machines
- App Service (Web-Apps)
- Container Instances
- Container Apps
- Azure Functions
- Azure Spring Cloud
- Service Fabric
- AKS (Azure Kubernetes Service)
- ARO (Azure Red Hat OpenShift)

App Service (Web-Apps)

- PaaS solution for different platforms: JVM, .NET, Node.js, PHP, Python, Ruby
- Deploy JAR files to be run with `` `java -jar` `` or WAR files to Tomcat
- Supports Linux container images (Docker)
- Automatic Ingress with SSL certificates and custom domains
- Blue/green deployment via deployment slots
- Integrated authentication via social auth and OpenID Connect
- Storage account integration
- Integration with API Management, Front Door and Azure CDN

App Service (Web-Apps) - Drawbacks

- Issues with App Service Java runtime stack
  - Java did not start after patch-update (application hangs on startup)
  - Very slow with updates (old Java, App Insights agent, etc.)
- Recommendation: deploy your own image
  - Add SSH support to your image: Enable SSH
  - Add Application Insights agent: Java Agent

Container Apps

- Serverless Kubernetes offering including Ingress, KEDA and Dapr
- Kubernetes Event-driven Autoscaling (KEDA) - scales your app on load. Scale to 0 possible.
- Dapr (Distributed Apps Runtime) is a runtime integration for different services
- Provides own secret store

- Drawbacks
  - Currently no custom domain support
  - Little visibility
  - No Kubernetes tool support (kubectl logs/exec/debug)

Azure Functions

- Function as a Service platform for .NET, Node.js, Python, Java, PowerShell
- Supports custom handlers (any binary) and Linux containers (Docker)
- Serverless and App Service Plan (VM) based deployments
- Custom handlers + GraalVM native-image 💕

- Drawbacks
  - No sub-paths - `` `/api/articles/123` `` not possible
  - Static file handling not possible/easy → use Static Webapps or Blob Storage

Kubernetes Solutions (OpenShift and AKS)

- Standard solutions - runs on any cloud / datacenter
- Excellent Kubernetes tools (kubectl logs, exec, debug, port-forward, ...)
- Azure RedHat OpenShift (ARO) and Azure Kubernets Service (AKS) are easy to setup/maintain
- Drawbacks
  - Ingress, Cert-Manager have to be deployed
  - You have to manage/upgrade the Kubernetes cluster and it's components
  - OpenShift has big overhead for small clusters - master nodes have to be run/paid

Comparison

Serivce	Price	Dev Experience	Operation/ Maintenance
VMs	$$	-	--
AppSVC	$$$$	○	+
Container-Apps	$$$	○	+
Spring Cloud	$$$$$*	○	+
Functions	$	○	+
AKS	$$	++	-
ARO	$$$$*	++	-

* depends on cluster size

Pricing Comparison

Compare monthly price of 2 vCPUs / 8GB RAM in West Europe, running the whole month

Service	Pay as you go	3 year reserved	3 year / 8GB RAM
AKS single node cluster	€ 78.90	€ 34.60	€ 34.60
Azure Webapp Linux	€ 117.04	€ 52.61	€ 52.61
Azure Container Apps	€ 98.86	€ 98.86	€ 98.86
Only availabe in bigger sizes:			
Azure Spring Cloud (8 vCPUs, 16 GB Memory)	€ 637.78	€ 637.78	€ 318,89
ARO three node cluster (3 x 4 vCPUs, 16 GB Memory)	€ 2,217.85	€ 875.70 + € 12,818.91 one-time	€ 205,30

Tips and Gotchas

App Service and X-Forwarded-For

- App Service sends proxy headers different than any other reverse proxy
- `` `X-Forwarded-For` `` includes the client port number (e.g. `` `165.225.200.218:2342` ``)
- Tomcat filter cannot cope with that - `` `HttpServletRequest#getRemoteAddr()` `` also contains port
- `` `IP:Port` `` in remote address break some frameworks/libraries

The solution for Spring Boot:

```
server.tomcat.remoteip.remote-ip-header=X-Client-Ip
```

Managed Identity with Hikari and PostgreSQL

```java
public class AzureHikariDataSource extends HikariDataSource {
  public static final TokenRequestContext TOKEN_CONTEXT =
    new TokenRequestContext().addScopes("https://ossrdbms-aad.database.windows.net");

  private TokenCredential managedIdentityCredential = new ManagedIdentityCredentialBuilder()
      .clientId(System.getenv("IDENTITY_CLIENT_ID")).build();
  private AccessToken token;

  // constructors calling super ...

  @Override
  public String getPassword() {
    if (token == null || token.isExpired()) {
      token = managedIdentityCredential.getToken(TOKEN_CONTEXT).block();
    }
    return token.getToken();
  }

  @Override
  public void setPassword(String password) {
    // ignore password
  }
}
```

Application Insights

APM Service with excellent Java support

- Agent is open source https://github.com/Microsoft/ApplicationInsights-Java (uses OpenTelemtry for Java)
- Autocollects metrics from a lot of frameworks / libraries
- Micrometer integration
- If you do not have an APM in your company yet use App Insights!
- Drawback: it is not cheap €2.843 per GB ingested data

Conclusion

- Lots of options to run Java on Azure, check your use case and then choose
- My recomendations:
  - Existing single app, simple scaling → App Service (Webapp) with Docker
  - Multiple apps / "Microservices" → AKS (Azure Kubernetes Service)
  - New greenfield app → check if Azure Functions works for you
  - Use Application Insights - it is awesome
  - Watch Contaier Apps - when ready they could replace App Service (and simple AKS scenarios)

🙏 Thank you! 🙏

Code: https://github.com/derkoe/java-on-azure-global-azure-2022

Slides: https://derkoe.dev/talks/

🐦 @derkoe

Sources

- Get started with Java on Azure
- Java at Microsoft
- Application Insights for Java
- Azure Java SDK