Java and AI

<Insert your name here if using>

JUG Presentation

Prepared by Zoran Sevarac & the JCP Executive Committee (EC) Java and AI Working Group

Version 1.0

About

- These slides are created to promote and share information about AI related work within Open JDK projects and Java ecosystem.
- Use them to disseminate information in your Development teams, communities, Java User Groups (JUG) and Java related events.
- Some of the Java features mentioned here are still incubating or under development.

About

- These slides are created to promote and share information about AI related work within Open JDK projects and Java ecosystem.
- Use them to disseminate information in your Development teams, communities, Java User Groups (JUG) and Java related events.
- Some of the Java features mentioned here are still incubating or under development.

Java and AI – The Big Picture

Java Platform

- New features are being developed within **OpenJDK** projects
- They enable high performance, safe, scalable and maintainable usage of native libraries, foreign platforms and specialized hardware from Java (GPUs and other hardware acceleration).
- They will enable Java to be the platform of choice for running AI workloads, offering powerful AI development platform

Ecosystem Evolution

- These features will enhance development of Java based AI frameworks, libraries and tools, that provide specific AI related features.

We continuously evolve Java to meet your future app development needs

Java has 30 years of experience evolving with the latest tech trends

Data-centric World

Amber

Continuously improve developer productivity through evolution of the Java language.

ZGC

Create a scalable low-latency garbage collector capable of handling terabyte heaps.

Cloud-powered World

Loom

Massively scale lightweight threads, making concurrency simple again.

Leyden

Improve the start-up time and time to peak performance of cloud applications.

AI-driven World

Babylon

Extend the reach of Java including to machine learning models and GPUs

Panama

Safe and efficient interoperation with native libraries and native Java.

Valhalla

Unify primitives and classes to improve productivity and performance.

Open JDK Projects Important for AI

- Panama - Interconnecting JVM and native code (C libraries) using **Foreign Function and Memory API**, and simplifying calls to native AI and math libraries.
  - Vector API - Using CPU vector operations to accelerate mathematical operations used for AI/machine learning (part of Panama)
- Babylon - Extend the reach of Java to foreign programming models such as GPUs (write specialized functions for parallel hardware in Java)
  - HAT (Heterogenious Acceleration Toolkit) How Babylon and Panama Enable Java GPGPU
- Valhalla Combining abstractions of the object-oriented programming with the performance characteristics of primitive types.

All together, they make Java platform faster, more suitable for numerical computing required for machine learning, and easier to use hardware acceleration.

Project Panama

Foreign Function and Memory API

API that simplifies calls to native functions written in C from Java

jextract - open source tool to automatically create initial Java binding to C native libraries from .h files.

How it will influence AI development:
Simplify development and usage of native libraries used in machine learning (mostly highly efficient numeric computation), and AI/ML libraries directly

It completely removes need for writing and compiling C code (as it was the case with JNI) and manually writing binding to native libraries and removes huge maintenance overhead.

https://openjdk.org/projects/panama/

Vector API

The **Java Vector API** is a feature in Java that helps your programs **run faster** by doing **many operations at the same time** — especially for **numbers and math**.

This is done using a technique called **SIMD** (Single Instruction, Multiple Data). Instead of adding numbers one by one, SIMD can add many numbers at once — like doing multiple calculator operations in a single step.

Why is it useful?

It’s especially helpful in anything that deals with big arrays of numbers like machine learning, image processing, physics simulations and scientific data processing.

https://openjdk.org/jeps/469

Valhalla

Aims to unify primitive types and classes, to improve performance and preserve productivity.
Enhance the Java language and JVM with data layout and type system features, so Java can better handle performance-sensitive workloads in areas like numerical computing and high-performance collections.

Its main goals are to make Java's data model more flexible without breaking existing code, and to reduce the overhead of Java’s traditional object model when you don’t need it.

How will it help AI?

It will unlock high performance numerical computing for Java, while preserving key object-oriented benefits.

Babylon

What is Project Babylon?

Project Babylon is effort to **make it easier for Java developers to write code that runs efficiently on GPUs and AI hardware**, which are often used for training and running large AI models.

How is it used?

Java developers will be able to:

- Write specialized function that will be executed on parallel hardware (called kernel) in Java code
- **offload compute-heavy tasks (execute kernel) to the GPU**
- Enjoy Java's safety, portability, and tooling—but with **AI-grade performance**.

Project Babylon = Java + GPUs + AI acceleration

It helps Java developers build high-performance AI apps **without switching to other languages** like C or CUDA.

HAT - Heterogenious Acceleration Toolkit

In Simple Terms: Normally, Java runs on the CPU. But today's computers also have powerful GPUs and other chips (like NPUs or FPGAs) that can handle certain tasks much faster than the CPU — especially for AI, machine learning, and scientific computing.
JHAT helps Java code "offload" some of that heavy work to those accelerators, without needing to write low-level accelerator code in C or CUDA.

What JHAT Does: Analyzes your Java code to find parts that can benefit from acceleration (like matrix operations). Compiles those parts to run on accelerators (GPU, etc.). Manages communication between your Java code and the hardware behind the scenes.

Why It's Important:
Brings Java closer to the performance of native AI/ML frameworks.
Let's Java developers use high-performance hardware without leaving the Java ecosystem.
Helps modernize Java for the AI era.

https://cr.openjdk.org/~psandoz/conferences/2024-JVMLS/JAVA_BABYLON_HAT-JVMLS-24-08-05.pdf

Approaches for Using Java for AI

Serve or use model as a web service	• You can build a model in any language, using a model is language agnostic. • Possible issues with latency and scalability.
Use Java wrappers or connect to native libraries from Java	• High performance and low latency. • Portability issues with native dependencies, memory/stability issues, limited scalability, distribution and maintenance overhead.
Use Java native AI libraries	• Highly scalable, low latency, easy to use, integrate with existing development and production environment, and distribute on large scale. • Java environment to build and run models in production. • Out of the box models mostly not available, but this approach is good for smaller and custom models

Java AI Ecosystem – examples

Serve or use model as a web service

- Langchain4j https://github.com/langchain4j/langchain4j
- Spring AI https://spring.io/projects/spring-ai
- Helidon https://helidon.io/

Use Java wrappers or connect to native libraries from Java

- DJL https://djl.ai/
- DL4J https://deeplearning4j.konduit.ai/

Use Java native AI libraries

- Jlama https://github.com/tjake/Jlama
- Deep Netts https://www.deepnetts.com

Want to help?

- You don’t have to become Open JDK contributor, just join the mailing list and share feedback
- Let us know the new Java features are relevant to your favorite library
- Have you tried them? Let us know your experience with the Early Access builds of the project(s)
- Be very specific about your experience and use case - what and how, with examples and all info required to reproduce.

Java and AI JCP EC Working Group

Official page:

https://github.com/jcp-org/Java-and-AI-Working-Group/

People:

- Heather Vancura, JCP Chair, Working Group Lead
- Zoran Sevarac, Deep Netts Technologies, Working Group Lead
- JCP Executive Committee Members