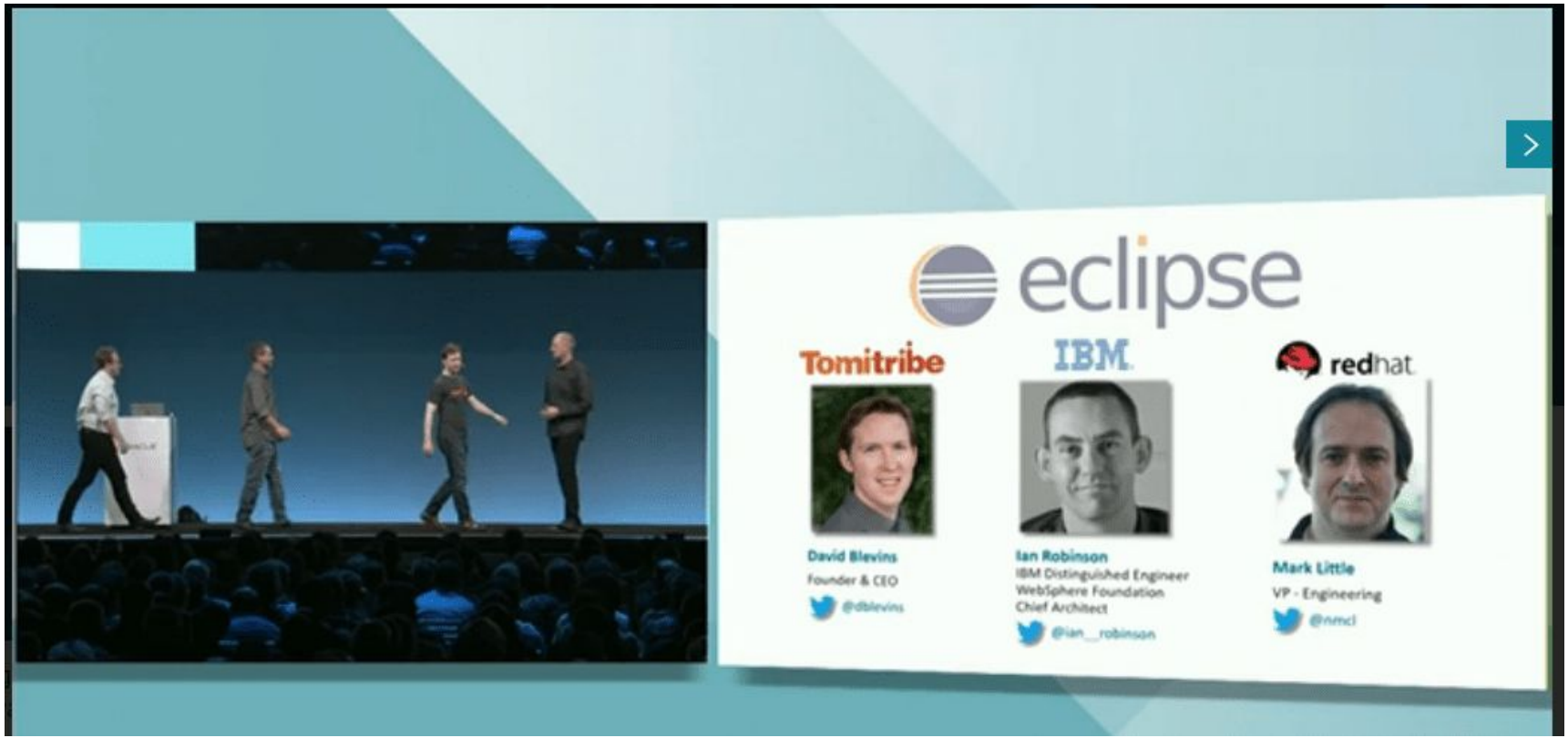
Jakarta EE: What is it and What Does it Mean for Enterprise Java?

JavaOne 2017



COMMUNITY, JAVAEE | Thursday, August 17, 2017

Opening Up Java EE

By: David Delabassee | Software Evangelist

We continue to make great progress on Java EE 8. Specifications are nearly complete, and we expect to deliver the reference implementation this summer. As we approach the delivery of Java EE 8 and the JavaOne 2017 conference, we believe there is an opportunity to rethink how Java EE is developed in order to make it more agile and responsive to changing industry and technology demands.

Java EE is enormously successful, with a competitive market of compatible implementations, broad adoption of individual technologies, a huge ecosystem of frameworks and tools, and countless applications delivering value to enterprises and end users. But although Java EE is developed in open source with the participation of the Java EE community, often the process is not seen as being agile, flexible or open enough, particularly when compared to other open source communities. We'd like to do better.

We are discussing how we can improve the Java EE development process following the delivery of Java EE 8. We believe that moving Java EE technologies including reference implementations and test compatibility kit to an open source foundation may be the right next step, in order to adopt more agile processes, implement more flexible licensing, and change

The Eclipse Enterprise for Java Project Top Level Project Charter

Please see the Frequently Asked Questions.

Overview

Eclipse Enterprise for Java (EE4J) is an open source initiative to create standard APIs, implementations of those APIs, and technology compatibility kits for Java runtimes that enable development, deployment, and management of server-side and cloud-native applications. EE4J is based on the Java™ Platform, Enterprise Edition (Java EE) standards, and uses Java EE 8 as the baseline for creating new standards.

Although Java EE was developed in open source, often the Java EE development process was not perceived as nimble, flexible or open enough when compared to other open source projects and processes. EE4J enables the use of more nimble processes, more flexible licensing, and a more open governance process for evolution of the platform.

This charter was developed in accordance with the **Eclipse Development Process** and will outline the mission of the EE4J Project. This document extends the Eclipse **Standard Top-Level Charter v1.2**, and includes the required content and overrides that follow. It is anticipated that as the standard charter is updated, this charter will incorporate the changes and make adjustments as seen fit by the PMC, and with approval from the EMO and board of directors.

Strategic Members

FUJITSU IBM ORACLE payara redhat Tomitribe

Participating Members

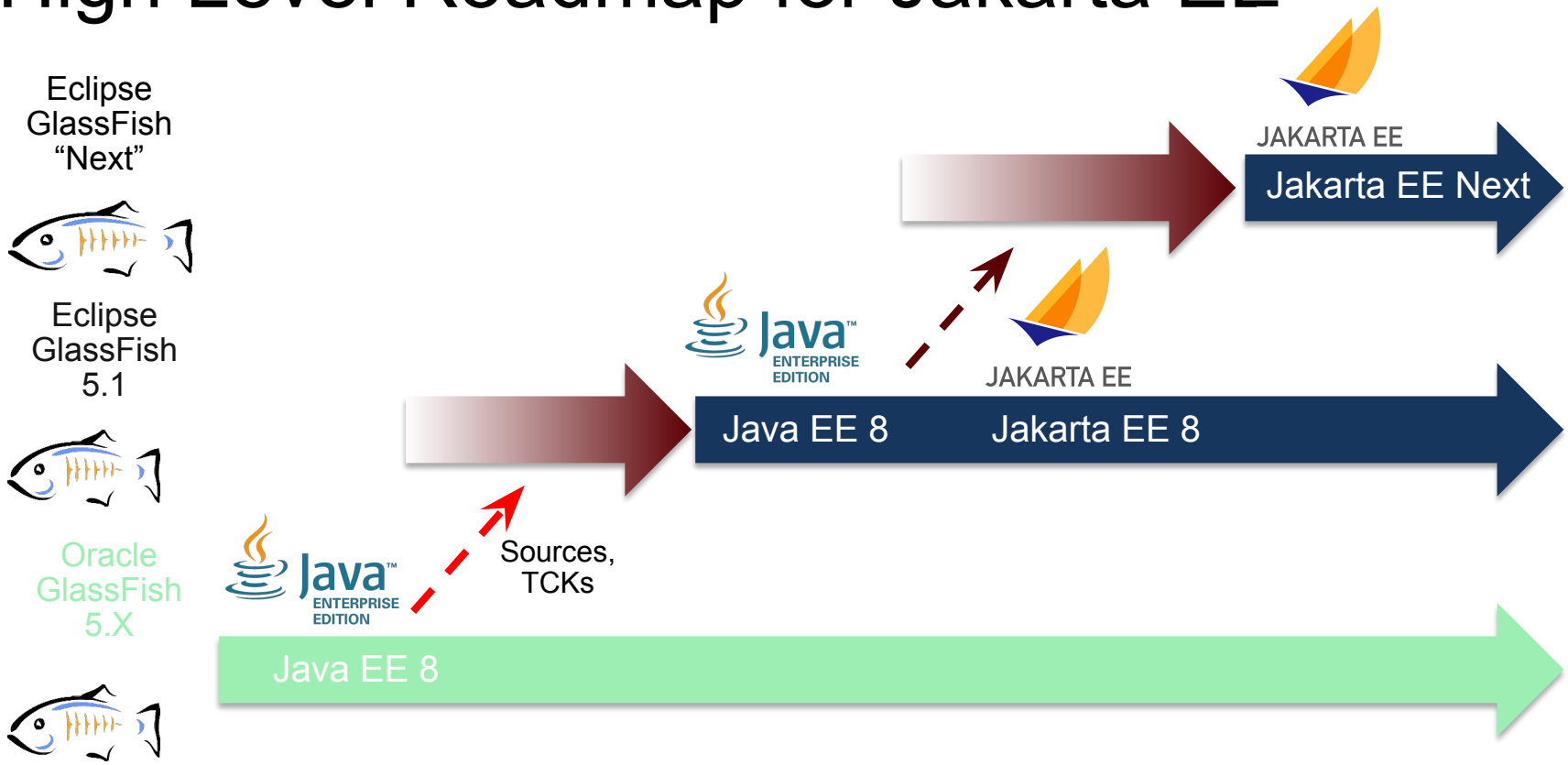
cloudbees DOCDOKU Genuitec IncQueryLabs Liferay Lightbend LJC

Microsoft MIZUHO Pivotal RCPVision SAP USEOPEN vaadin

webtide

https://jakarta.ee/

High Level Roadmap for Jakarta EE



Eclipse GlassFish Contributions Complete, RC1 on 10/22

https://www.eclipse.org/ee4j/status.php

GlassFish Project

Jersey (JAX-RS)

JSONB & JSONP

HK2

JavaServer Faces (Mojarra)

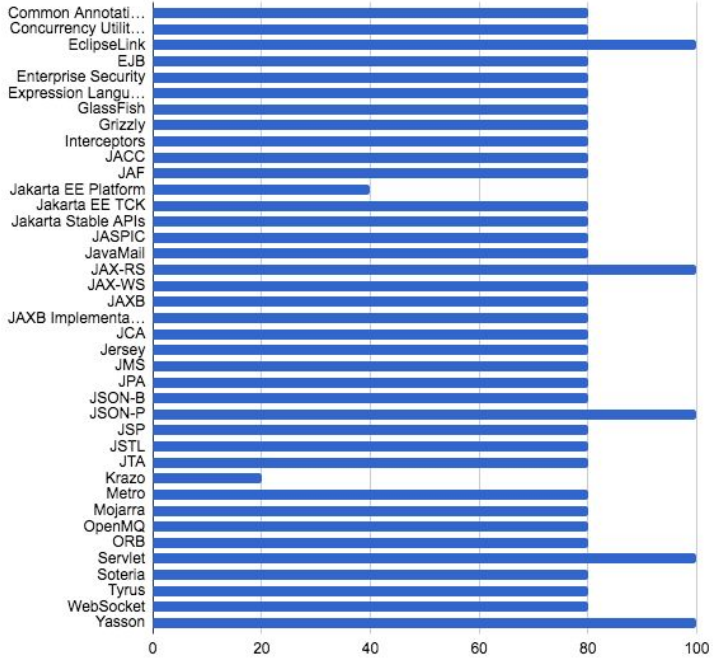
Open MQ (JMS)

Metro (JAX*)

JavaMail

… and much, much more

7.7 M Lines of code
Over 60K files
38 Projects



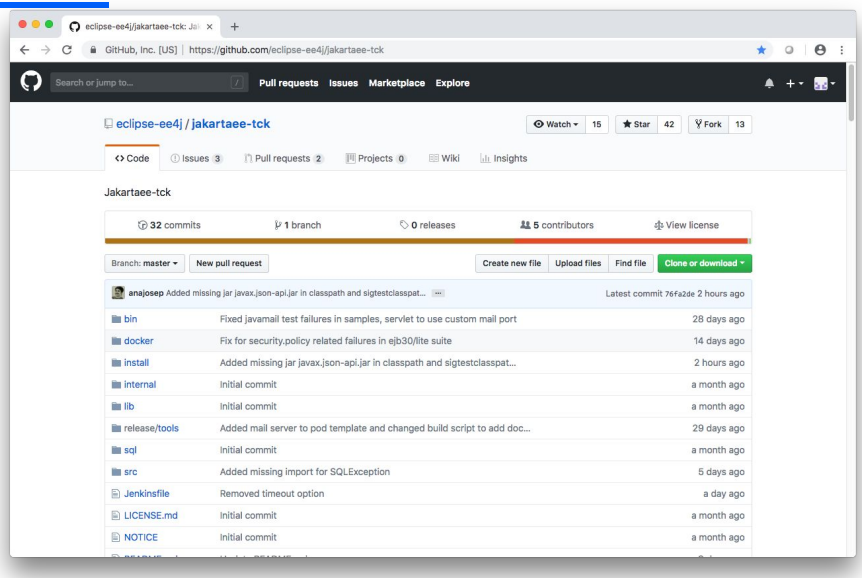
Java EE 8 TCKs Are Now Open Source in Jakarta EE!

https://github.com/eclipse-ee4j/jakartaee-tck

All the TCK source-code is available

All the necessary porting kits

Intended as foundation for Jakarta EE 8 TCKS



5.7 M Lines of code
Over 30K files

New Specification Process

https://tinyurl.com/ybh8sx8j

Eclipse Foundation Specification Process

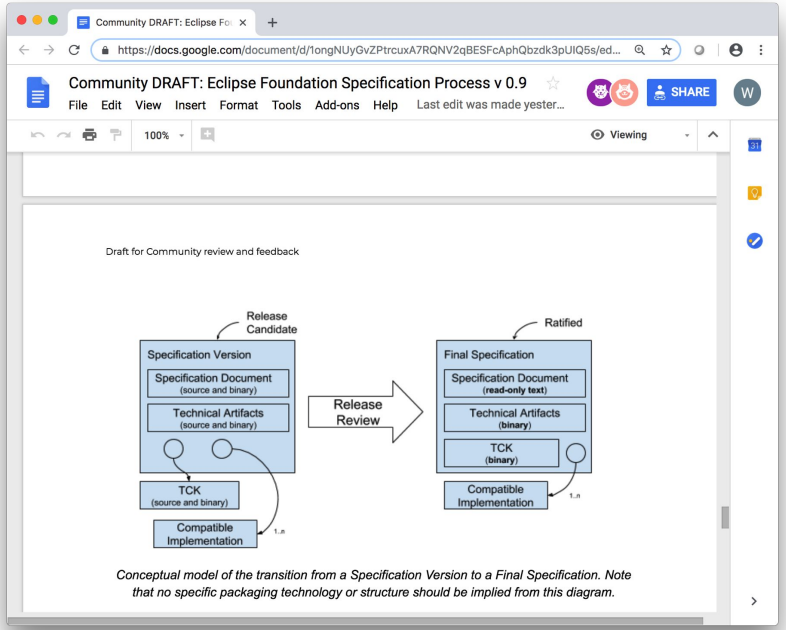
Developed by Jakarta EE Spec Committee

Feedback mechanisms

– Jakarta EE community mailing list (preferred)
– Document comments

Approach

– Based on Eclipse Development Process
– Allow code-first development
– Produce high quality specifications



Jakarta EE Technical Directions

Top 3

Critical areas cited for improvement:

1. Better support for microservices
2. Native integration with Kubernetes
3. A faster pace of innovation

Top

Frameworks for building microservices include: Jersey, Spring, Eclipse MicroProfile, Node.js & Kubernetes

67%

Currently building microservices or planning to <1 yr

40%

Say large memory requirements most challenging aspect of working with Java EE

#1

Reason Java EE chosen for Java applications is stability

60%

Say Foundation should prioritize better support for microservices

Key Updates

- Announcing Eclipse GlassFish
- Schedule for Eclipse GlassFish Java EE 8 Certification
- Java EE TCKs are open sourced
- New Specification Process
- Working Group Member Commitments
- Technical Direction

Working Group Member Commitments

To evolve Jakarta EE technologies

Certify offerings as Jakarta EE compatible

Leverage technologies in offerings

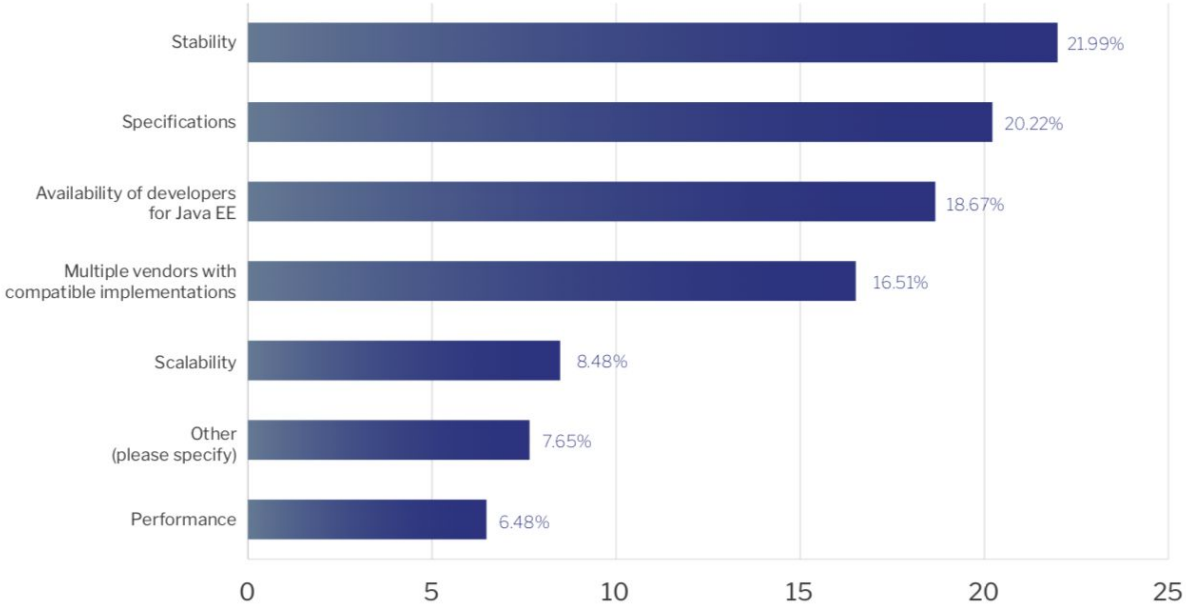
Committed to three years of funding

- Marketing activities
- Project management
- Infrastructure

Shape the Future of Cloud Native Java

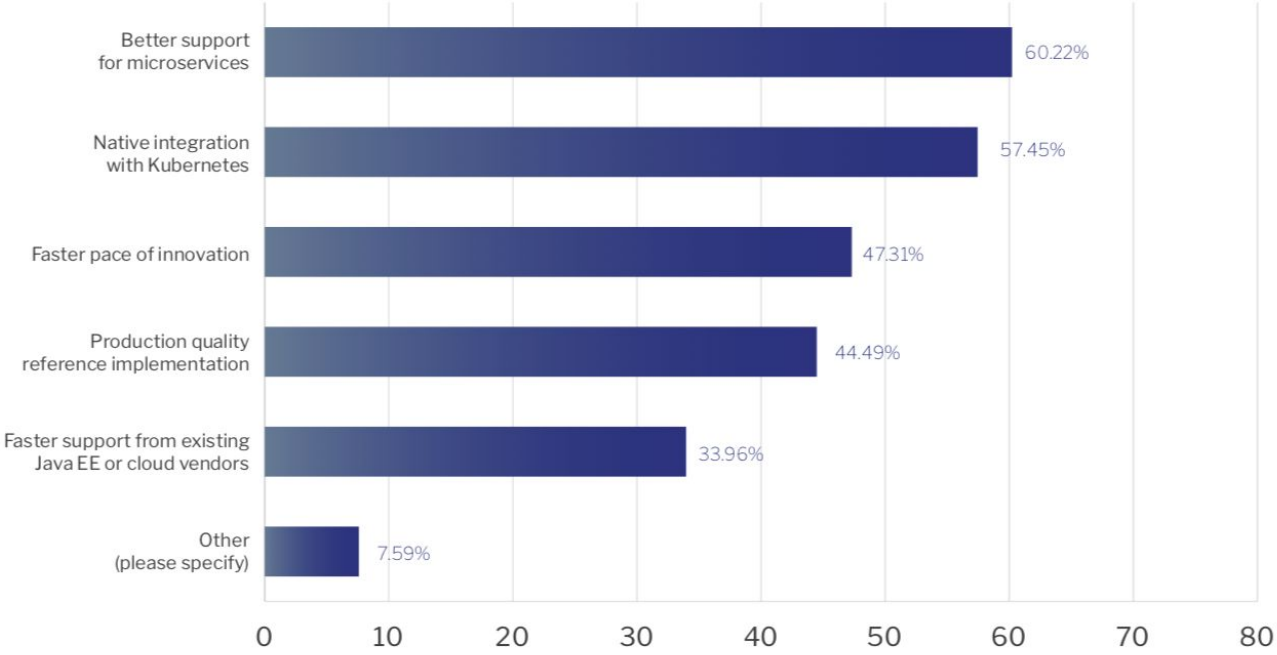
- Join the Jakarta EE community
  - https://accounts.eclipse.org/mailing-list/jakarta.ee-community
- Join the Jakarta EE Working Group
  - https://accounts.eclipse.org/mailing-list/jakarta.ee-wg
- Join the Jakarta EE specifications list
  - https://accounts.eclipse.org/mailing-list/jakarta.ee-spec

WHAT ASPECT OF JAVA EE HAS MOST MADE IT THE PLATFORM OF CHOICE FOR YOUR JAVA APPLICATIONS?

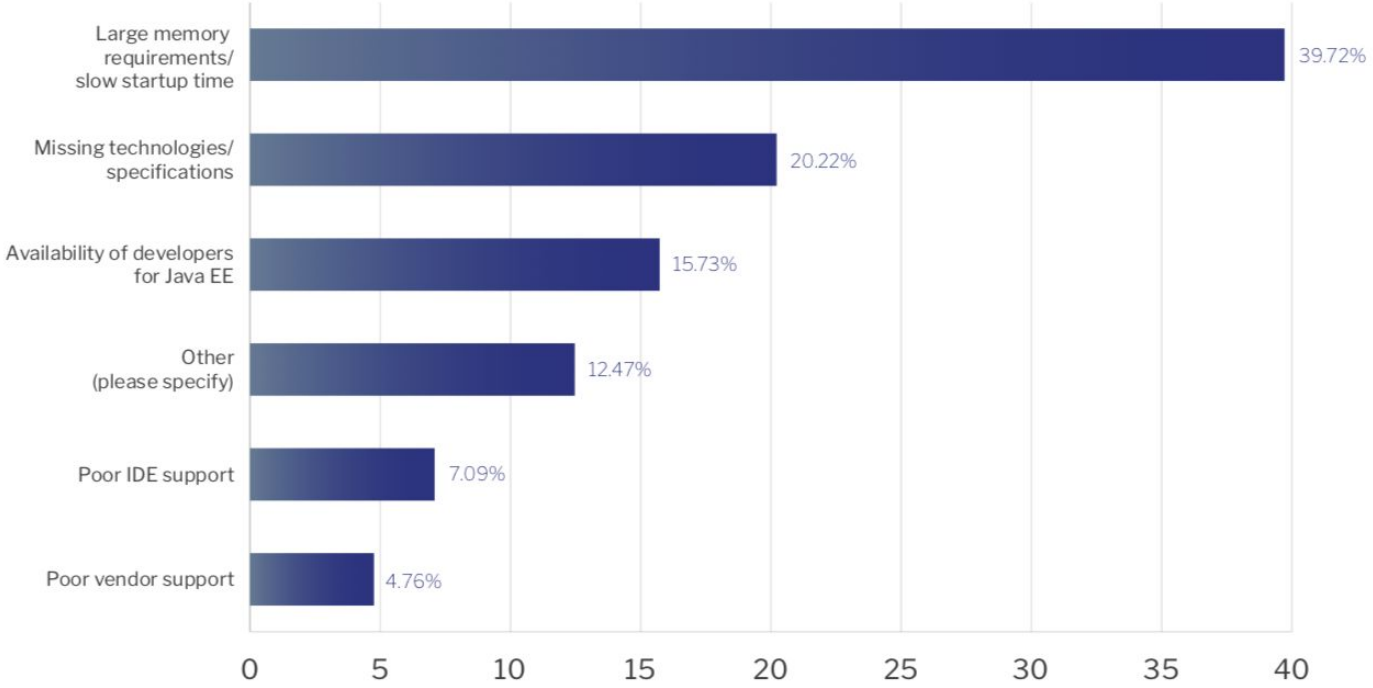


*https://jakarta.ee/news/2018/04/24/jakarta-ee-community-survey/

HOW CAN THE ECLIPSE FOUNDATION BEST EVOLVE JAKARTA EE TO MEET YOUR CLOUD NEEDS? SELECT ALL THAT APPLY.



WHAT IS THE MOST CHALLENGING ASPECT OF WORKING WITH JAVA EE?



WHAT PERCENTAGE OF YOUR JAVA SYSTEMS WILL BE RUNNING IN THE CLOUD IN TWO YEARS?