Waratek Q & A: What's new in Java 15?

New Java version includes hi-performance GCs, TLS support; says bye to Solaris support

Oracle has released a new, powerful version of the Java platform that Waratek Founder & CTO John Matthew Holt says has exciting new features which positions Java above other current application platforms. CEO John Adams and Holt discuss the new features found in Java 15 and how Waratek's *Upgrade* agent helps organizations stuck on older Java versions.

*Click to **listen to the full Q & A** or read the edited transcript below.*

John Adams: What's new in Java 15?

John Matthew Holt: *There are a number of features in Java 15 that have been experimental are fully supported production features for the very first time. The most significant new feature that programmers and developers may see is sealed classes. And, there are about 50 new classes and methods in Java 15, about 25 of them allow Java 15 to support a key part of the TLS 1.3 standard.*

What is really exciting to me is the great step in the Java platform with Java 15 debuting two new low-pause garbage collectors. ZGC and Shenandoah GC can support enormous heaps sizes; hundreds of gigabytes with constant low-pause time. It's a really big change for the way Java programmers and Java developers have thought about heap size in the past. These new GC really do put the Java platform head and shoulders above any other abstract machine runtime in the industry and any other application engines in the industry.

JA: Java 11 was a big leap forward and ended a lot of long-time Java features. Are there features ending in this version of Java?

JMH: *The aggressive major release cadence has kind of made it possible to roll back old features and impediments to development and improvement that hindered the evolution of the platform with the older version management style. For example, in Java 15, the Nashorn JavaScript engine - something that's been around for many Java releases - has been removed entirely. The Remote Method Invocation (RMI) activation feature set has now been deprecated for removal in a subsequent release.*

Similarly, all of the code for supporting Solaris and the Sparc platforms and the porting of the JDK for those platforms have also been removed now in Java 15. Backing out the Solaris and Sparc support entirely frees up the JDK and these new feature initiatives to kind of sprint ahead without carrying these older architectures with them.

JA: What are the advantages to the new garbage collectors?

JMH: *The sophistication of these two new GCs is just way beyond what any other JIT compiler runtime has at its disposal today. Customers using the Waratek Upgrade agent are now able to bring the benefit of these low-pause collectors to bear on their existing workloads without having to make any source code changes or worry about incompatibilities with the application. They can essentially just press a button, restart their application, and suddenly their existing core business workloads and business critical workloads can take advantage of the high performance that can come from these new collectors. In the case of Waratek customers, that benefit now goes all the way back to Java 4.*

JA: As of earlier this year, 2/3rds of developers where still writing apps in Java 8. How difficult is it to move from Java 8 to Java 15?

JMH: *For most applications, and certainly for most significant applications in terms of code size or sophistication, it's not feasible to move to the newer versions of Java and certainly not something like Java 15. That's the dark side of the rapid change in Java that began with Java 9. So, while Java's new version management process from Java 9 onwards has facilitated and enabled an explosion of new features and improvements, it has come at the very conscious trade off of sacrificing backwards compatibility. For new apps that are under very active development with code bases that are relatively young in calendar terms, that's fine.*

But, the majority of established, trusted business critical apps are powered by Java 8, which has become something of a glass ceiling that these established apps have no hope of passing through. That's what's holding back the large number of developers still working in Java. The code they're working on today began life on Java 8 or earlier and it's just not feasible to refactor that code for Java 15 today and then in five months-time refactor that code again for Java 16, and then six months later do it for Java 17, and so on and so on into the future.

JA: With so many versions of Java available, how do Waratek's solutions help companies manage their Java estates?

JMH: *There's a really big need for tools to bridge the chasm that has emerged between - on the one hand, the universe of established Java workloads that power most of the world's enterprises - and on the other hand, the rapidly evolving Java platform embodied in Java 15 today.*

We produce tiny agents for a range of languages and platforms to solve security and compliance and modernization needs for core business workloads. But one of those agents, aptly named the Waratek Upgrade agent, instantly migrates any Java app on any Java version immediately to the newest JDK with no source code changes and no incompatibilities. The consequence of this product set and distribution set is that after an application is restarted with the Waratek Upgrade agent it immediately begins running on that latest JDK (Java 15) with all of the features and benefits that that new JDK brings with it. The new low-pause garbage collectors, the new and faster JIT compilers, newer crypto, and TLS standards compliance, and, of course, all of the security updates and fixes that are that are so important for applications as well.