The latest generally available version of Java is JDK 24, released in March 2025. The upcoming Long-Term Support (LTS) release, JDK 25, is scheduled for September 2025. Recent versions of Java introduce language and API improvements that enhance performance, simplify concurrent programming, and improve developer productivity.

New features in JDK 24 (March 2025)
1) Stream Gatherers (Standard): This feature standardizes custom intermediate operations in the Stream API, offering more flexibility for data transformations.

2) Ahead-of-Time (AOT) Class Loading & Linking (Standard): As part of Project Leyden, this can improve application startup performance by pre-loading and linking classes from a previous "training run".

3) Class-File API (Standard): JDK 24 finalized a standard API for parsing, generating, and transforming Java class files, reducing the reliance on third-party libraries for bytecode manipulation.

4) Synchronize Virtual Threads without Pinning (Standard): This resolves a performance issue where blocking synchronization operations would pin a virtual thread to its carrier, improving the scalability of concurrent applications.

5) Quantum-Resistant Cryptography (Standard): Adds quantum-resistant key encapsulation and digital signature algorithms to future-proof Java's cryptography.

Key features coming in JDK 25 (September 2025)
JDK 25 is the next LTS release, and many of its features are moving from preview to final status.
1) Flexible Constructor Bodies (Final): Developers can place statements before an explicit super() or this() constructor call. This enables more expressive and fail-fast validation logic.

2) Primitive Types in Patterns (Preview): This is a refinement of pattern matching that allows the use of primitive types like int and double in instanceof and switch statements, bringing more consistency to the feature.

3) Structured Concurrency (Fifth Preview): Continued refinement of the API that simplifies concurrent programming by treating groups of related tasks as a single unit. This helps streamline error handling and cancellation.

4) Scoped Values (Final): Offers an efficient, immutable, and thread-safe way to share data within a thread and its child threads. This is an improved alternative to ThreadLocal.

5) Compact Source Files and Instance Main Methods (Standard): This simplifies the Java language for beginners and for writing small scripts by removing the need for top-level class declarations and static main methods.

6) Ahead-of-Time Method Profiling (Final): Improves application warmup times by using profiles recorded from a prior run to inform the Just-in-Time (JIT) compiler.