JAVA SE 24

Java SE 24 was released on **March 18, 2025** as a **non-LTS (short-term) release**. Unlike Java 21 (LTS), this version will only be supported for six months, but it's packed with experimental, preview, and finalized features that pave the way for future LTS releases.

Key Features of Java SE 24

1. Memory & Performance

- **Generational Shenandoah GC (Experimental)**
  Improves performance of the Shenandoah garbage collector by separating young and old generations.

- **Compact Object Headers (Experimental)**
  Shrinks object header size, reducing memory footprint and improving cache usage (part of Project Lilliput).

2. Language Enhancements

- **Scoped Values (Fourth Preview)**
  A safer alternative to thread-local variables for passing immutable data across threads.

- **Primitive Types in Patterns & `switch` (Second Preview)**
  Allows primitive values in pattern matching and `switch`, making code more expressive.

- **Flexible Constructor Bodies (Third Preview)**
  Lets you write initialization code *before* calling `super()` or `this()` in constructors.

- **Module Import Declarations (Second Preview)**
  Adds more flexible ways of importing modules.

- **Simple Source Files & Instance `main` (Fourth Preview)**
  Enables short, script-like Java programs without boilerplate.

3. APIs & Libraries

- **Stream Gatherers (Final)**
  Adds a new `Stream.gather(Gatherer)` method for advanced data processing

like sliding windows, folding, or grouping.

- **Class-File API (Final)**
  New API for reading and writing Java class files programmatically.

- **Key Derivation Function API (Preview)**
  Introduces standard APIs for deriving secure cryptographic keys.

4. Virtual Threads & Project Loom

- **Synchronize Virtual Threads Without Pinning (Final)**
  Removes the issue where synchronized blocks could "pin" a virtual thread to an OS thread, improving scalability.

5. Security & Cryptography

- **Quantum-Resistant Algorithms (Final)**
  Adds support for ML-KEM (key encapsulation) and ML-DSA (digital signatures) to prepare Java for a post-quantum future.

- **Warnings on `sun.misc.Unsafe` (Final)**
  Now warns when using old, unsafe memory access methods.

6. Garbage Collection & JVM

- **ZGC: Remove Non-Generational Mode (Final)**
  Simplifies ZGC by keeping only the generational mode.

- **G1 GC Improvements** with late barrier expansion for efficiency.

7. Removals & Deprecations

- **Removed Security Manager (Final)**
  Fully disabled, as it's considered outdated.

- **Deprecated/Removed 32-bit x86 Port**
  Focus shifts completely to modern 64-bit platforms.

- **Preparing to restrict JNI (Java Native Interface)** for better security.