22. What is the current Java version ? tell me some features

Since the current date is **January 2026**, the latest long-term support (LTS) version is **Java 25 (LTS)**, released in September 2025. The latest non-LTS version is likely **Java 26** (due March 2026).

Here is how to answer this:

Answer

"The current LTS version is **Java 25**, released in September 2025. However, many enterprise applications (including mine) are running on **Java 21 LTS** or **Java 17 LTS** for stability."

Key Features of Recent Versions (Java 21 to 25)

Focus on the features that actually matter for backend development:

1. Virtual Threads (Project Loom) - *The Big One*

- **What it is:** Lightweight threads that are managed by the JVM, not the OS. You can create millions of them.
- **Why it matters:** It revolutionizes concurrency. We no longer need complex reactive code (WebFlux) for high throughput. We can write simple "thread-per-request" code that scales massively.

2. Pattern Matching for Switch

- **What it is:** You can use `switch` on objects and handle types directly, removing ugly `instanceof` casts.
- **Example:**

```
return switch (obj) {
    case Integer i -> String.format("int %d", i);
    case String s  -> String.format("string %s", s);
    default        -> obj.toString();
};
```

3. Record Patterns

- **What it is:** Deconstruct records directly in an `instanceof` or `switch` check.
- **Example:**

```
if (obj instanceof Point(int x, int y)) {
    System.out.println(x + y); // No need to call .x() or .y()
}
```

4. Sequenced Collections

- **What it is:** Uniform methods (`addFirst`, `addLast`, `getFirst`, `getLast`) for all ordered collections (List, Deque, SortedSet).
- **Why it matters:** No more checking "is this a List or a Deque?" to access the first element.

5. String Templates (Preview/Standard)

- **What it is:** String interpolation like in Python/JavaScript.
- **Example:** `STR."Hello \{name}"` instead of `String.format("Hello %s", name)`.

6. Scoped Values (Project Loom)

- **What it is:** A modern, lightweight alternative to `ThreadLocal`. It allows passing data (like Transaction IDs or User Context) safely to a specific scope of code and its child threads without memory leaks.

Which one is your favorite?

Be ready for this follow-up.
"My favorite is **Virtual Threads**. It solves the 'throughput vs. complexity' dilemma. We can keep our simple synchronous code style but get the scalability benefits of asynchronous I/O."

⁂

1. https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/images/81815274/429036f4-3bc3-4a7a-8c6f-90246e73b9b9/image.jpg
2. https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/images/81815274/24f18c0a-b56e-4862-a619-e2959036a5c2/image.jpg
3. https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/images/81815274/97e9ed86-ed11-4035-a18b-1eb05c1bab4e/image.jpg
4. https://ppl-ai-file-upload.s3.amazonaws.com/web/direct-files/attachments/images/81815274/87f62423-96f2-4071-9802-8f6699e0ecd8/image.jpg