Java 17 features

java 17 features

Java 17 (released on **September 14, 2021**) is a **Long-Term Support (LTS)** release, making it a major version for long-term stability and adoption. It includes several important **new features, enhancements, deprecations, and removals**.

✅ Major Features and Enhancements in Java 17

1. Sealed Classes (Standard) – JEP 409

Purpose: Restrict which other classes or interfaces may extend or implement them.

```java
public sealed class Animal permits Cat, Dog {}

final class Cat extends Animal {}
final class Dog extends Animal {}
```

✅ Benefits:

- Better control over class hierarchy.
- Great for pattern matching and ADTs (algebraic data types).

---

2. Pattern Matching for switch (Preview) – JEP 406

Purpose: Simplifies switch statements using pattern matching.

```java
static String formatter(Object obj) {
    return switch (obj) {
        case Integer i -> "int: " + i;
        case String s -> "string: " + s;
        default -> "unknown";
    };
}
```

✅ Benefits:

- Cleaner and type-safe code.
- Works with sealed types, enums, etc.

---

3. New macOS Rendering Pipeline – JEP 382
Purpose: Replace the old Quartz pipeline with a new **Metal-based** rendering pipeline on macOS.

✅ Benefits:

- Better performance and compatibility with new macOS versions.

---

4. Foreign Function & Memory API (Incubator) – JEP 412
Purpose: Allows Java programs to interoperate with native code and memory (like C) **without JNI**.

```
java

try (MemorySegment segment = MemorySegment.allocateNative(100,
ResourceScope.newConfinedScope())) {
    // Use native memory safely
}
```

✅ Benefits:

- Safer, faster, and more expressive than JNI.

---

5. Strong Encapsulation of JDK Internals – JEP 403
Purpose: Encapsulate internal elements of the JDK by default (`sun.misc.Unsafe`, etc.)

✅ Impact:

- Accessing internal APIs without `--add-opens` causes `IllegalAccessError`.

---

6. Enhanced Pseudo-Random Number Generators – JEP 356

Purpose: Provides new interfaces and implementations for random number generation.

```java
RandomGenerator gen = RandomGeneratorFactory.of("L64X256MixRandom").create();
int num = gen.nextInt();
```

✅ Benefits:

- Better statistical quality, reproducibility, pluggability.

---

7. Deprecate and Remove RMI Activation – JEP 407

- `java.rmi.activation` is **removed**.
- RMI itself still remains (for now).

---

8. Deprecate Applet API – JEP 398

- Applet is now **deprecated for removal**.
- Modern browsers no longer support Java Applets anyway.

---

9. Context-Specific Deserialization Filters – JEP 415

Purpose: Improve **security** by allowing fine-grained filtering during object deserialization.

```java
ObjectInputFilter filter = ObjectInputFilter.allowFilter(cl -> cl != null &&
cl.getPackageName().startsWith("my.safe.package"), ObjectInputFilter.Status.REJECTED);
```

✅ Benefits:

- Prevents deserialization-based attacks.

📦 Removed Features

Feature	Status
RMI Activation	Removed

Feature	Status
Experimental AOT and JIT Compiler (`jaotc`)	Removed (JEP 410)
Applet API	Deprecated
Security Manager	Deprecated for removal (JEP 411)

✨ Other Improvements

Enhancement	Description
JDK Flight Recorder (JFR) improvements	More events and stability
JVM internals performance	Code cache management, metaspace tuning
Support for Alpine Linux on AArch64	Better container compatibility

---

🧠 Summary Table

Feature	Type	JEP
Sealed Classes	Standard	409
Pattern Matching for `switch`	Preview	406
Foreign Function & Memory API	Incubator	412
Enhanced Pseudo-Random Generators	Standard	356
Metal rendering on macOS	Standard	382

Feature	Type	JEP
Strong encapsulation of JDK internals	Standard	403
Context-Specific Deserialization Filters	Standard	415
Removed: Experimental AOT, RMI Activation	Removal	410, 407
Deprecated: Applet API, Security Manager	Deprecation	398, 411

Would you like:

- Full working examples for each new feature?
- Difference between Java 11 vs 17?
- Java 17 vs 21 comparison?

Let me know what you'd like next.