

## Java

java, programming, jvm, language

### Java 9부터 변경점

#### 9

- Java Platform **Module System(JPMS)** → 라이브러리 만들때 제외하고는 사용이 드물다
- 확장된 try-with-resources → try 밖 지역변수도 사용 가능
- @SafeVarargs를 private 메소드에서도 사용 가능
- 익명 Inner Class에도 diamond syntax(Generic 타입 명시 생략) 가능
- Interface안에 private 메소드 사용 가능
- '\_' 하나만으로 변수, 함수 이름 사용 불가능
- Collection에서 정적(불변) 팩토리 메소드 'of' 추가
- Optional에서 ifPresentOrElse, or, stream 메소드 추가
- Stream
  - takewhile(predicate)<sup>1)</sup>, dropWhile(predicate)<sup>2)</sup>
  - ofNullable(t)
  - iterate가 for문 인자와 비슷하게 개선 (limit 메서드 대체) → iterate(0, i -> i < 10, i -> i + 1);
- CompletableFuture에 타임아웃(orTimeout(long)), 지연(CompletableFuture.delayedExecutor()) 기능 추가
- Process API에서 native 프로세스를 제어할 수 있는 ProcessHandle 인터페이스 추가
- StackWalker → Stack을 훑는 기능. Debug할 때 좋을 듯.
- 내부 문자열 처리 방식 개선으로 메모리 절약. 기존에는 UTF-16 char 배열이었지만 문자에 따라 1byte 혹은 2byte가 필요한지에 따라 다르게 저장 → isLatin()
- **G1GC가 default GC. 이전 Parallel GC가 메모리를 물리적으로 나누었다면, G1GC는 메모리를 바둑판처럼 쪼갠 region이라는 논리적 개념으로 관리한다.**
- OOM 발생시 사용할 수 있는 옵션 두 가지 추가
  - ExitOnOutOfMemoryError
  - CrashOnOutOfMemoryError
- Flow → Java에서 공식적으로 Reactive Stream API 추가. 직접 구현을 해보면 Observer 패턴

#### 10

- 타입 추론이 가능한, 가변 변수 예약어 'var'
- Collection에서 깊은 복사 'copyOf' 팩토리 메소드 추가
- G1GC 성능 개선

#### 11(LTS)

- var에 랴다식 매개변수에 적용 가능
- String, Predicate 기능 추가
- Files 클래스에 파일 전체↔문자열 변환(readString/WriteString) 기능 추가
- 새로운 Http Client 모듈
- ZGC(experimental) → 25에서 정식 GC로 승격

## 12 ~ 17(LTS)

## 15

### Text Block

여러 줄에 걸친 문자열을 만들기 위한 새로운 문법

12에서 preview였다가 15에서 정식 기능으로 승격

```
// "" 다음으로 문자가 올 수 없고, 한줄로 작성할 수 없다!
// \를 사용하면 개행 제거
// 들여쓰기도 가능
String str = ""
    A
    BC
    DEF""
```

### Switch Expression 개선

12에서 preview였다가 14에서 정식 기능으로 승격

```
private String calculateTestGrade1(int score) {
    return switch (score) {
        case 5:
            yield "A";
        case 4, 3:
            yield "B";
        case 2:
            yield "C";
        default:
            yield "F";
    };
}

private String calculateTestGrade2(int score) {
    return switch (score) {
        case 5 -> "A";
        case 4, 3 -> "B";
        case 2 -> {
            System.out.println("C!");
            yield "C";
        }
        default -> "F";
    };
}
```

```
}
```

## 'Instance of' Pattern Matching

14에서 preview였다가 16에서 정식 기능으로 승격

```
public String say(Animal animal) throws IllegalAccessException {
    if (animal instanceof Dog dog) {
        return dog.bark();
    } else if (animal instanceof Cat cat) {
        return cat.purr();
    }

    throw new IllegalAccessException();
}

// 아래처럼 부정인 경우 Scope가 괄호 밖까지 확장 가능
public String sayIfDog(Animal animal) throws IllegalAccessException {
    if (!(animal instanceof Dog dog)) {
        throw new IllegalAccessException();
    }

    return dog.bark();
}

public interface Animal {
}

public static class Dog implements Animal {
    public String bark() {
        return "Dog barking...";
    }
}

public static class Cat implements Animal {
    public String purr() {
        return "Cat purring...";
    }
}
```

## Record Class

데이터 전달을 위한 클래스 (DTO) → Lombok 기능을 대체

14에서 preview였다가 16에서 정식 기능으로 승격

```
public record PersonDtoV1 (
    String name,
    int age
) {

    // Compact Constructor. 매개변수를 전혀 받지 않는다. this를 사용하지 않는다. (매개변수가 있다고 가정)
    /*public PersonDtoV1 {
    }*/
}
```

## Sealed Class/Interface

하위 클래스를 지정된 클래스로만 상속을 제한(봉인)

자바 15에서 preview였다가 17에서 정식 기능으로 승격

```
// 한 파일에 있다면 permits 생략 가능
public sealed interface Animal permits Dog, Cat {
}

// final : 재상속 불가능
// sealed : 한 번더 sealed class로 동작
// non-sealed : 상속 가능하지만 하위 타입 추적 불가능
public final static class Dog implements Animal {
    public String bark() {
        return "Dog barking...";
    }
}

public final static class Cat implements Animal {
    public String purr() {
        return "Cat purring...";
    }
}
```

<sup>1)</sup> 참인 경우 뒤에 버림

<sup>2)</sup> 참인 경우 앞에 버림

From:

<http://ledyx.xyz/> - Ledyx WIKI

Permanent link:

<http://ledyx.xyz/doku.php?id=language:java&rev=1788245404>

Last update: 2026/09/01 06:50



