Java 꽉 잡아!
- JVM부터 GC, 스레드 동기화까지!

JDK Description

김재녕 (2023-05-24)

[0] 참고

참고

Features 상태 별 차이 (Preview, Experimental, Incubating)

- Preview Features
  - 완전히 구현되었지만 영구적이지 않음, 개발자 피드백을 위한 용도이며 컴파일러의 `--enable-preview` 플래그를 통해 사용 가능
  - 베타 테스트 용도로 거의 완성된 상태이나 프로덕션 환경에서 사용은 권장되지 않음
  - 예시) JEP 325: Switch Expressions
- Experimental Features
  - 불안정하거나 불완전한 수준의 초기 버전, 전용 플래그를 통해 활성화 후 사용 가능
  - 만약 `Experimental` Features의 완성도가 약 25%라면 `Preview` Features의 완성도는 최소 95% 이상이어야 함
  - 예시) JEP-189: Shenandoah GC
- Incubating Features
  - 앞에 `jdk.incubator`가 붙은 `Experimental` API이며 별도의 모듈 형태로 배포됨
  - 접근하려면 해당 모듈을 명시적으로 추가해야 함
  - 예시) JEP-338: Vector API

[1] JDK

JDK

OpenJDK vs Oracle JDK

- 정의
  - Oracle JDK는 최초의 Java SE Platform (JDK)
  - OpenJDK는 Oracle JDK 기반 무료 버전 Java SE Platform (JDK)
- Release Schedule
  - Oracle JDK는 3년마다, OpenJDK는 6개월마다
  - Oracle JDK는 LTS 릴리스 지원, OpenJDK는 릴리스 변경 사항만 `다음 버전 릴리스까지` 지원
- Licenses
  - Oracle JDK는NFTC/OTN/BCL 3가지, OpenJDK는 GNU GPL
  - Oracle JDK는 8이후 버전은 비즈니스에 상용 라이센스 필요(NFTC 제외), OpenJDK는 비즈니스에 무료로 사용 가능
- Performance
  - OpenJDK가 Oracle JDK를 기반으로 하고 있기 때문에 성능 차이는 거의 없음
  - 하지만 OpenJDK가 일부적인 측면에서 조금 더 불안정할 수 있음
- Features
  - 11 버전 이후부터는 두 JDK는 거의 동일하지만 일부 기능, 옵션에 대한 차이가 존재
- Developments, Popularity
  - Oracle JDK는 Oracle 주도, OpenJDK는 Red Hat/Azul/IBM/Apple 등 다양한 기업의 개발 참여

JDK

Oracle JDK 17 License

- 현재 나온 가장 최신 LTS 버전
- Oracle JDK 17 버전부터 NFTC (No-Fee Terms and Conditions) 라이센스 적용
- NFTC (No-Fee Terms and Conditions) 라이센스
  - 앱 개발, 테스트, 프로토타이핑 등 개인적 사용 및 비즈니스 내부 운영 목적으로 사용 가능
  - 수정되지 않은 프로그램과 문서를 라이센스 규정에 따라 재배포 가능
  - 라이센스 사용자에게 사용, 배포 등 관련해 수수료 부과 금지
- NFTC 라이센스 지원 기간 (평생 무료가 아님)
  - LTS 버전 - 다음 LTS 버전 릴리스 이후 1년 (그 이후에는 OTN 라이센스 적용)
  - Non-LTS 버전 - 해당 버전 릴리스 이후 6개월
- OTN (Oracle Technology Network) 라이센스
  - Oracle JDK 8 버전부터 개인, 내부 운영 목적으로 무료로 사용 가능하나 상용 제한
  - 비독점적, 양도가 불가능한 제한된 라이센스
- BCL (Binary Code License)
  - 현재 Java SE Subscription이라는 구독형 유료 라이센스로 변경
  - 서버는 프로세서 단위 과금 (실행 중인 모든 프로세스에 적용), 데스크탑은 사용자 단위 과금 (설치된 경우)

JDK

JDK implementations by vendors

- OpenJDK (Oracle)
  - Oracle에서 제공하는 무료 JDK
- Oracle JDK (Oracle)
  - OpenJDK 기반의 상용 JDK
- Corretto (AWS)
  - OpenJDK 기반, GPL 라이센스
- Eclipse Temurin (Adoptium)
  - 기존 AdoptOpenJDK
- IcedTea (RedHat)
  - OpenJDK 기반, 재배포 가능, GPL 라이센스
- J9 (IBM)
  - 상용이었으나 오픈 소스로 변경된 JDK
- SapMachine (SAP)
  - OpenJDK 기반, GPL 라이센스
- Zulu (Azul System)
  - OpenJDK 기반, GPL 라이센스
- Zing (Azul System)
  - 고성능 상용 JDK, GPL 라이센스
  - 64비트 리눅스에서만 동작
  - 대용량 힙 메모리와 멀티 CPU 서버급 시스템을 위해 설계됨

[2] Java Tools & Components by version

Java Tools & Components by version

JDK 8

- Basic Tools
  - Javac, APT, jar 등 Java 애플리케이션을 생성, 빌드, 아카이브 등에 사용하는 도구
- Security Tools
  - KeyStore를 조작하기 위한 키, 인증서 관리 도구
    - KeyStore는 암호화, 인증, HTTP 통신 등을 위한 보안 컨테이너
- Internationalization Tool
  - 국제화를 위한 도구
  - 포함된 ~~native2ascii~~(JDK 9부터 제거)를 통해 문자가 포함된 파일을 ASCII 또는 Unicode로 인코딩된 파일로 변환
- ~~RMI (Remote Method Invocation) Tools~~ (JDK 9부터 제거)
  - Java 애플리케이션 간 원격 통신을 위한 도구 (서로 다른 JVM 간 다른 객체의 메서드 호출 가능)
- Java IDL(Interface Definition Language), RMI-IIOP(Internet InterORB Protocol) Tools
  - IDL은 CORBA(Common Object-Based Request Broker Architecture) 기능을 Java 플랫폼에 추가
    - IDL은 특정 언어에 종속되지 않아 소프트웨어 컴포넌트 간 상호 운용성을 보장
    - CORBA는 분산 컴퓨팅 환경에서 객체들이 서로 통신하는 방법을 정의한 규격
  - IIOP를 통한 RMI는 CORBA 서버와 앱 프로그래밍을 가능하게 함

Java Tools & Components by version

JDK 8

- Java Deployment Tools
  - pack200, unpack200 등 웹에서 Java 애플리케이션과 애플릿의 배포 등을 지원하는 도구
  - jar 파일은 zip 파일로 구성되어 있음
- ~~Java Plug-in Tool~~ (JDK 11부터 제외)
  - Java Applet, Java Web Start를 실행하는 브라우저 확장을 관리하는 도구
- ~~Java Web Start Tool (javaws)~~ (JDK 11부터 제거)
  - 브라우저에서 Java 애플리케이션의 다운로드/실행을 도와주는 도구
- Monitoring and Management Tools
  - jConsole, jps, jstat 등 JVM의 성능과 리소스의 사용 상태를 모니터링 하는 도구
- Troubleshooting Tools (실험용)
  - info, jmap, jstack 등 트러블슈팅을 위한 도구

Java Tools & Components by version

JDK 17

- jar
  - 클래스, 리소스에 대한 아카이브 생성
- jarsigner
  - jar 파일 서명 및 검증
- java
  - Java 앱 실행
- javac
  - Java 클래스, 인터페이스 등을 읽어 바이트코드(.class 파일)로 컴파일
- javadoc
  - Java 소스 파일로부터 API 문서의 HTML 페이지 생성
- javap
  - 하나 이상의 클래스 파일 분해하여 내부 구조 이해 및 바이트코드 분석 후 출력하는 도구
- jcmd
  - 실행중인 스레드 분석이나 GC 관리, 스레드 덤프 등을 수행하는 JVM 관리 도구
- jconsole
  - GUI를 통해 Java 앱 모니터링
- jdb
  - CLI 기반 디버깅 도구로서 실행 중단점 설정, 스텝 인/아웃 등 디버깅 작업 수행
- jdeprscan
  - 사용중인 deprecated API를 찾기 위해 jar 파일을 스캔하는 정적 분석 도구

Java Tools & Components by version

JDK 17

- jdeps
  - Java 클래스 파일 및 jar 파일의 종속성 분석
- jfr (Java Flight Recorder)
  - 이벤트 기반 프로파일링 도구
- jhsdb
  - Java 프로세스 연결 및 손상된 JVM 코어 덤프 내용 분석 (힙 덤프, 스레드 덤프 등을 분석)
- jinfo
  - 특정 Java 프로세스에 대한 Java(JVM) 구성 정보 확인, 수정
- jlink
  - 모듈 셋과 종속성을 커스텀 런타임 이미지로 어셈블, 최적화하여 생성
- jmap
  - 특정 Java 프로세스의 세부 정보(메모리 등)를 분석, 출력 (힙 덤프 생성, 메모리 맵 생성 등)
- jmod (모듈화된 jar 파일 포맷, 모듈 시스템에서 사용)
  - jmod 파일은 컴파일된 클래스, 리소스, 모듈 디스크립터 등과 같은 내용을 포함
- jpackage
  - 해당 Java 앱이 포함된 네이티브 패키지로 변환, OS에 종속적이지 않은 패키지를 생성
- jps
  - JVM이 실행중인 Java 프로세스의 정보를 출력

Java Tools & Components by version

JDK 17

- jrunscript
  - JS 등으로 작성된 CLI 스크립트 실행 (대화형, 배치 모드 등 지원)
- jshell
  - REPL(Read-Eval-Print Loop) 환경 제공, 코드를 바로 실행
- jstack
  - 지정된 Java 프로세스에 대한 Java 스레드의 스택을 추적하여 출력
- jstat
  - JVM의 성능 통계 및 모니터링 정보 수집
- jstatd
  - HotSpot VM의 생성, 종료를 모니터링하고 원격 모니터링 도구 사용을 위한 인터페이스를 제공하는 RMI 서버 (데몬)
- keytool
  - 키, 인증서 등과 KeyStore(데이터베이스) 관리
- rmid (JDK 19 버전부터 제거)
  - 객체의 등록, 검색 관리를 위해 활성화 시스템 데몬 실행
- rmiregistry
  - RMI(분산 시스템에서 객체 간 원격 호출)에서 사용되는 레지스트리를 생성 및 실행
- serialver
  - 직렬화된 클래스의 `serialVersionUID` 값 생성
- jwebserver (JDK 18 버전부터 추가)
  - 프로토타이핑, 테스트 등을 위한 간단한 HTTP 서버 제공

Java Tools & Components by version

Remove Tools & Components

- JDK 9
  - JVM TI hprof Agent (프로파일링 도구, hprof Agent 대신 jmap 사용)
  - VisualVM (JVM에서 실행되는 Java 앱)
  - native2ascii Tool (인코딩 도구, UTF-8 포맷이 기본이 되면서 필요 없어짐)
- JDK 10
  - javah (C 헤더 파일 및 소스 파일 생성 도구)
  - jhat Tool (힙 시각화 도구)
  - JavaDB (RDB, Aphache Derby)
- JDK 11
  - Java EE (Enterprise Edition, SE JDK에 기능이 추가적으로 포함된 기업용 버전)
  - CORBA 모듈 (Common Object-Based Request Broker Architecture, 분산 컴퓨팅 환경에서 앱을 실행하기 위한 표준)
  - JavaFX (GUI 도구)
- JDK 12
  - SecurityWarning 클래스 (Java Applet 관련 보안 경고 처리 클래스)
  - java.io 패키지의 FileIn(Out)putStream 클래스, java.util 패키지의 ZipFile/Inflator/Deflator 클래스 안에 finalize 메서드
  - Oracle 빌드 정보 스키마에서 `YY.M` 포맷의 문자열 벤더 버전
    - [Java 11] Java(TM) SE Runtime Environment 18.9 (build 11+28)
    - [Java 12] Java(TM) SE Runtime Environment (build 12+17)

Java Tools & Components by version

Remove Tools & Components

- JDK 13
  - Javadoc Tool의 오래된 기능 (HTML 4, 오래된 javadoc API 등)
- JDK 14
  - Pack200 Tool (앱 압축 패키지 생성과 인코딩-디코딩 도구)
  - CMS Garbage Collector (Concurrent Mark Sweep GC)
- JDK 15
  - Nashorn JavaScript Engine (JS를 실행하기 위한 엔진)
  - RMI Static Stub Compiler Tool (이하 rmic는 원격 객체에 대한 스켈레톤/스텁을 생성하는 도구)
- JDK 16
  - 실험용 Java Ahead-of-Time 컴파일러 jaotc, Java-based Graal JIT 컴파일러
  - KeyStore의 1024비트 RSA 공개키를 포함한 루트 인증서
  - 오래된 Elliptic Curves (암호화 기술 중 하나)
- JDK 17
  - RMI Activation 메커니즘 (나머지 RMI 요소는 보존)

Java Tools & Components by version

javah 부연 설명

- javah는 헤더 파일을 생성하는 도구
  일반적으로 Java는 자체 런타임 환경에서 라이브러리를 통해 시스템 리소스에 접근함
  - 따라서 Java 개발자가 직접 C 헤더 파일을 작성하는 경우가 거의 없음
- 그러면 Java에서 C 헤더 파일이 필요한 경우?
  - 특정 하드웨어를 직접 제어해야 하는 경우
  - 모종의 이유로 C/C++ 라이브러리를 재사용하는 경우
  - 성능적으로 최적화된 네이티브 코드를 호출해야 하는 경우
  - 그 외 직접 시스템 API에 접근해야 하는 경우
- JNI를 통해 네이티브 소스를 호출하는 경우?
  - 네이티브 코드에서 Java 네이티브 메서드를 참조하기 위해
    C 헤더 파일로 제공되는 글루코드(Glue Code)가 필요함
- 그럼 없앤 이유는 무엇일까?
  - Java 컴파일러(javac)의 성능 개선(JNI 헤더 생성 기능 추가)으로 javah의 C/C++ 컴파일러 종속성 제거

[3] Java Features by version

Java Features by version

JDK 8

- 인터페이스에 디폴트/스태틱 메서드 추가
- 람다, 펑셔널 인터페이스, 메서드 참조 기능 추가
- Optional 클래스 추가
- 같은 주석을 같은 곳에 중첩하여 사용 가능

Java Features by version
JDK 9

- 모듈러 시스템 추가
  - 모듈 단위의 관리 (종속성, 가시성 등), 이로 인해 모듈 기반 JVM 제공 가능
  - Java에서 모듈은 모듈 디스크립터(module-info.java)를 포함한 관련된 패키지 그룹
- 새로운 HTTP Client 추가 (Incubator)
- Process API 개선
- try-with-resource 추가
- 다이아몬드 연산자 개선 (익명 내부클래스에서 사용 허용)
- 인터페이스에 프라이빗 메서드 추가
- JDK 8에서 추가 되었던 G1 GC가 디폴트 GC로 설정됨

Java Features by version
JDK 10

- var 타입 추가
- 불변 컬렉션 추가
- Optional 클래스에 orElseThrow 메서드 추가
- 성능 향상
  - G1 GC (Mark-sweep-compact) 알고리즘이 싱글 처리에서 병렬 처리로 변경
  - CDS(Class-Data Sharing) 범위를 부트스트랩 클래스 로더에서 시스템 클래스 로더까지 확장
  - Graal 컴파일러를 실험용 JIT 컴파일러로 사용 가능
  - `for` 루프의 바이트코드 생성 개선
- 도커 컨테이너 실행여부 인식
  - 리눅스 베이스 한정으로 도커 컨테이너로 실행한 지를 인식해 컨테이너에 할당된 리소스를 쿼리
- Root Certificates 포함
- javah 삭제, Java 컴파일러에 `javac -h` 옵션 추가

Java Features by version
JDK 11 (1 page)

- 해당 버전부터 OpenJDK 제공
  - 11 버전부터 상업용으로 사용할 수 있는 완전 무료 LTS 버전이 없음
- String 클래스에 메서드 추가 (isBlank, lines, strip, repeat 등)
- File 클래스에 메서드 추가 (readString, writeString 등)
- Collection 인터페이스에 디폴트 메서드 추가 (toArray 등)
- Predicate 펑셔널 인터페이스에 메서드 추가 (not 등)
- 람다 파라미터에서 var 타입 변수 사용 가능
- HTTP Client (GA)
- Nest Based Access Control 개선
  - 기존에는 이너 클래스가 아우터 클래스의 프라이빗 멤버에 접근하기 위해서 브릿지 메서드가 필요했음
    각각 다른 클래스 파일로 컴파일 되기 때문에
  - 브릿지 메서드는 소스 코드 레벨에서는 보이지 않고 바이트코드 레벨에서만 노출되었으나 보안적인 이슈가 존재
  - 두 클래스를 중첩 위치로 배치하는 클래스 파일로 생성하도록 Java 컴파일러 개선
- Flight Recorder (JFR)
  - 실행 중인 앱을 진단하고 데이터를 수집하는 프로파일링 도구

Java Features by version
JDK 11 (2 page)

- Single-file Source-code program 추가
  - shebang이라는 기술을 통해 컴파일 없이 Java 단일 파일을 실행하는 기능
- `CONSTANT_Dynamic` 이라는 새로운 상수 풀 추가
  - 상수 값 계산을 부트스트랩 메서드에게 위임하여 유연한 동적 바인딩 제공
- AArch64(ARM 64) 내장 함수 향상
  - 해당 프로세서의 문자열과 배열 내장 함수 최적화 (`java.lang.Math`의 sin, cos, log 메서드를 위한 내장 함수 추가)
- Epsilon GC 추가 (experimental)
  - 메모리를 할당하지만 실제로는 GC 처리가 이뤄지지 않음 (실제 앱의 GC로는 적합하지 않음)
  - `out of memory` 테스트와 성능 테스트, 메모리 테스트, VM 인터페이스 등 다양한 테스트 가능
- 컴파일러 스레드의 동적 할당 기능 추가 (`UseDynamicNumberOfCompilerThreads` 옵션 추가, 디폴트 `on`)
  - 기존에는 요청 수에 상관없이 CPU가 많은 시스템 위에서 많은 수의 컴파일러 스레드를 실행
    스레드는 유휴 상태여도 메모리 낭비가 심한 리소스
- ZGC GC 추가 (experimental)
  - 동시 처리 GC로 Java 스레드가 실행되는 동안 계속해서 작업이 수행됨
    marking, compaction, reference processing, string table cleaning 등
- 오버 헤드가 낮은 방법으로 Heap 할당 샘플링(표본 생성)이 가능하고 JVMTI 통해 엑세스 가능

Java Features by version
JDK 12

- String 클래스에 메서드 추가 (indent 등)
- File 클래스에 메서드 추가 (mismatch 등)
- Collectors 클래스에 Teeing 컬렉터 추가
  - 두개의 컬렉터를 병렬로 처리후 병합하는 컬렉터를 생성
- CompactNumberFormat 추가 (`1_000_000` -> `1M`으로 표현)
- switch에 표현식 추가 (preview)
- instanceof 패턴 매칭 추가 (preview)
- 마이크로벤치마크 스위트 추가
- CDS(Class Data Sharing)의 디폴트가 활성화로 변경
- ZGC GC의 클래스 언로드 기능 추가
- G1 GC의 `Old generation` 영역을 다른 메모리 디바이스에 할당하는 기능 추가 (experimental)
- 특정 클래스들의 finalize 메서드 제거
- Shenandoah GC 추가 (OpenJDK - experimental)

Java Features by version
JDK 13

- switch에 yield 구문 추가 (Preview)
- String 클래스 text blocks 기능 추가 (Preview)
  여러 줄에 걸친 문자열 표현 기능
- 동적 CDS (Class-Data Sharing) 아카이빙 허용
- ZGC 개선
  - 사용되지 않은 힙 메모리를 OS로 반환하는 기능 추가
  - 최대 지원 힙 크기가 4TB > 16TB로 증가
- 레거시 Socket API 개편 (Reimplement)

Java Features by version
JDK 14

- instanceof 패턴 매칭 추가 (Preview)
- Record 클래스 추가 (Preview)
- Helpful NullPointerException 추가 (디폴트 값 `off`)
- ZGC가 Windows, MacOS도 지원 (Experimental)
- G1 GC도 NUMA(Non-uniform memory access)를 인식하도록 개선
  - NUMA(불균형 기억 장치 접근)는 프로세서별 독립적인 메모리 공간 소유
- CMS(Concurrent Mark Sweep) GC 제거
- Foreign Memory Access API (Incubator)
  - 힙 외부 메모리 액세스 API

Java Features by version
JDK 15

- Record 클래스 개선 (Preview)
- Sealed 클래스 추가 (Preview)
- Hidden 클래스 추가 (JVM 레벨)
  - 검색할 수 없는 클래스를 런타임에 생성 (리플렉션으로 찾을 수 없음)
- ZGC (GA)
- Shenandoah GC (GA)
- String 클래스의 text blocks (GA)
- Helpful NullPointerException의 디폴트 값을 `on`으로 변경
- TreeMap 클래스가 오버라이딩 가능한 메서드(putIfAbsent, computeIfAbsent) 제공

Java Features by version

JDK 16

- 인터페이스의 디폴트 메서드 내부 메커니즘 개선
  - `java.lang.reflect.invocationHandler`가 리플렉션을 통해 dynamic proxy가 디폴트 메서드를 호출하도록 개선
- Stream 클래스에 toList 메서드 추가
- Vector API 추가 (Incubator)
  - 벡터(행렬 or 배열) 연산
- Record 클래스 개선 (GA)
- Sealed 클래스 개선 (Preview)
- instanceof 패턴 매칭 (GA)
- Foreign Linker API (Incubator)
  - 네이티브 코드에 대한 정적 타입의 Pure Java 액세스를 제공하는 API
- ZGC가 스레드 스택을 동시에 처리
  - `stop-the-world` 퍼즈 대신 동시 처리 단계에서 JVM의 모든 루트를 처리할 수 있음
- G1 GC가 언커밋 메모리를 동시에 해제 처리
  - 비싼 작업을 Java 앱과 동시 실행되는 스레드로 오프로드
- Elastic Metaspace를 통해 메타스페이스 구현 점검
  - 클래스 언로드시 바로 해당 메모리가 OS로 반환됨
- jaotc(AOT 컴파일 도구)와 Graal(JIT 컴파일 도구) 제거
  - jaotc는 컴파일 시 내부적으로 Graal 컴파일러를 사용했음

Java Features by version
JDK 17

- 부동 소수점 연산 시맨틱을 항상 strict으로 실행하도록 복원 (strictfp 제거)
  - 기존에는 하드웨어 이슈로 strictfp, strict라는 2가지를 지원해서 오버헤드가 존재했음
- Pseudo-Random Number Generators 개선
  - 시드 데이터를 기반으로 한 랜덤 숫자 생성기
- MacOS를 위한 새로운 렌더링 파이프라인 추가
- JDK 내부를 강력하게 캡슐화
  - 기존에는 내부 API를 검색, 사용할 수 있었음
- switch에 패턴 매칭 추가 (Preview)
- RMI Activation 제거
- Sealed 클래스 (GA)
- 실험용이었던 AOT, JIT 컴파일러 (관련 코드) 제거
- JNI를 대체하기 위한 Foreign Function & Memory API 추가 (Incubator)
  - Foreign Function API : 네이티브 함수 호출을 위한 API (Foreign Linker API 포함)
  - Foreign Memory Acess API : 힙 외부 메모리 접근 API

참고 & 출처

참고 & 출처

JDK

- https://www.oracle.com/a/tech/docs/jdk17-lium.pdf
- https://blogs.oracle.com/javakr/post/jdk-17
- https://www.oracle.com/downloads/licenses/javase-license1.html
- https://www.oracle.com/downloads/licenses/binary-code-license.html
- https://www.oracle.com/kr/java/technologies/javase/jdk-faqs.html
- https://www.baeldung.com/oracle-jdk-vs-openjdk

참고 & 출처

Java Tools & Components

- https://docs.oracle.com/javase/8/docs/technotes/tools/
- https://docs.oracle.com/en/java/javase/17/migrate/removed-tools-and-components.html

참고 & 출처

Java 8

- https://www.baeldung.com/java-8-new-features
- https://www.oracle.com/java/technologies/javase/8-whats-new.html
- https://openjdk.org/projects/jdk8/features

참고 & 출처

Java 9

- https://www.baeldung.com/new-java-9
- https://www.baeldung.com/java-9-modularity
- https://www.oracle.com/java/technologies/javase/9-relnotes.html
- https://openjdk.org/projects/jdk9/

참고 & 출처

Java 10

- https://www.baeldung.com/java-10-overview
- https://www.baeldung.com/java-10-performance-improvements
- https://www.oracle.com/java/technologies/javase/10-relnote-issues.html
- https://openjdk.org/projects/jdk/10/

참고 & 출처

Java 11

- https://www.baeldung.com/java-11-new-features
- https://www.baeldung.com/java-single-file-source-code
- https://www.baeldung.com/java-nest-based-access-control
- https://www.oracle.com/java/technologies/javase/11-relnote-issues.html
- https://openjdk.org/projects/jdk/11/

참고 & 출처

Java 12

- https://www.baeldung.com/java-12-new-features
- https://www.baeldung.com/java-8-collectors
- https://www.baeldung.com/java-switch
- https://www.oracle.com/java/technologies/javase/12-relnote-issues.html
- https://openjdk.org/projects/jdk/12/

참고 & 출처

Java 13

- https://www.baeldung.com/java-13-new-features
- https://www.oracle.com/java/technologies/javase/13-relnote-issues.html
- https://openjdk.org/projects/jdk/13/

참고 & 출처

Java 14

- https://www.baeldung.com/java-14-new-features
- https://www.oracle.com/java/technologies/javase/14-relnote-issues.html
- https://openjdk.org/projects/jdk/14/

참고 & 출처

Java 15

- https://www.baeldung.com/java-15-new
- https://www.oracle.com/java/technologies/javase/15-relnote-issues.html
- https://openjdk.org/projects/jdk/15/

참고 & 출처

Java 16

- https://www.baeldung.com/java-16-new-features
- https://www.baeldung.com/java-dynamic-proxies
- https://www.oracle.com/java/technologies/javase/16-relnote-issues.html
- https://openjdk.org/projects/jdk/16/

참고 & 출처

Java 17

- https://www.baeldung.com/java-17-new-features
- https://www.oracle.com/java/technologies/javase/17-relnote-issues.html
- https://openjdk.org/projects/jdk/17/