

런타임 데드 코드 분석 도구 Scavenger

-당신의 코드는 생각보다 많이 죽어있다.

김태연 권오준

NAVER Platform Labs

NAVER DEVVIEW 2023

CONTENTS

1. What is Dead Code?

1.1 Dead Code의 정의 / 발생 원인

1.2 Dead Code의 문제점

1.3 Dead Code 검출 방법

2. What is Scavenger?

2.1 Scavenger 데모

2.2 Scavenger

2.3 Scavenger 활용

2.4 Scavenger 아키텍처

3. Behind the Scenes

3.1 코드베이스 탐색

3.2 메서드 호출 추적

3.3 데이터의 전송, 기록, 열람

4. 정리 및 마무리

1. What is Dead Code?

1.1 Dead Code의 정의

Dead Code란? <https://devopedia.org/dead-code>

- 실행되지 않는(Unreachable) 코드
- 실행되더라도 애플리케이션의 동작에 영향을 미치지 않는 (Executed But Not Effect) 코드

Unreachable Code

```
public class SampleClass {  
    private void methodA() {  
    }  
    private void methodB() {  
        methodA();  
    }  
}
```

Executed But Not Effect

```
private int methodA() {  
    int value = methodB1();  
    value = methodB2();  
    return value;  
}  
private int methodB1() { return 1; }  
private int methodB2() { return 2; }
```

1.1 Dead Code의 발생 원인

- 협업팀이 API 개발 요청만 하고, 미사용 통보를 안 할 때... ☹
- 새로운 기획 반영 후 기존 코드를 삭제 안 할 때
- 이거 진짜 지워도 돼?? 불안하고... 로그 찍기 귀찮고... 잘 돌아가니.. 그냥 놔두자...

```
public class SampleController {  
    @Get("/new") 미사용 API  
    public void methodA() {  
        methodB();  
    }  
  
    private void methodB() { }  
}
```

```
public void methodC(Data data) {  
    if (data.type == TYPE.A) { methodA(); }  
    else if (data.type == TYPE.B) { methodB(); }  
    else if (data.type == TYPE.C) { methodC(); }  
    .....  
    스펙 아웃 된 타입  
}  
  
private void methodC() { }
```

1.1 Dead Code의 발생 원인

- 협업팀이 API 개발 요청만 하고, 미사용 통보를 안 할 때... ☹
- 새로운 기획 반영 후 기존 코드를 삭제 안 할 때
- 이거 진짜 지워도 돼?? 불안하고... 로그 찍기 귀찮고... 잘 돌아가니.. 그냥 놔두자...

```
public class SampleController {
    @Get("/new")
    public void methodA() {
        methodB();
    }
    private void methodB() { }
}
```

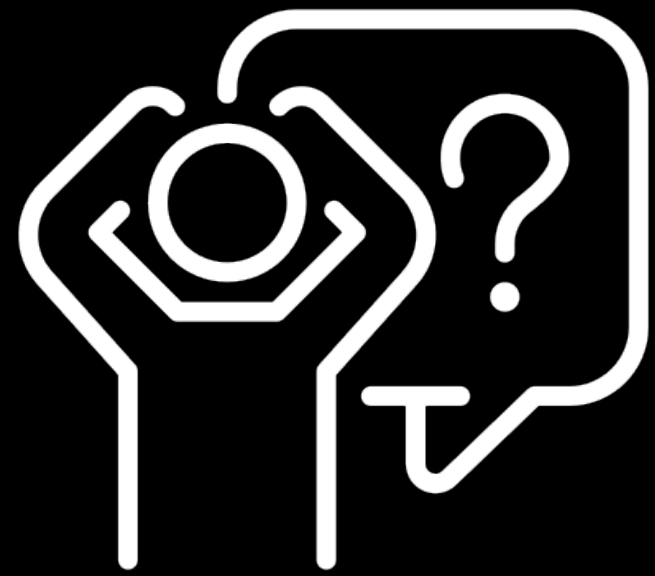
Dead Code

```
public void methodC(Data data) {
    if (data.type == TYPE.A) { methodA(); }
    else if (data.type == TYPE.B) { methodB(); }
    else if (data.type == TYPE.C) { methodC(); }
    .....
}
private void methodC() { }
```

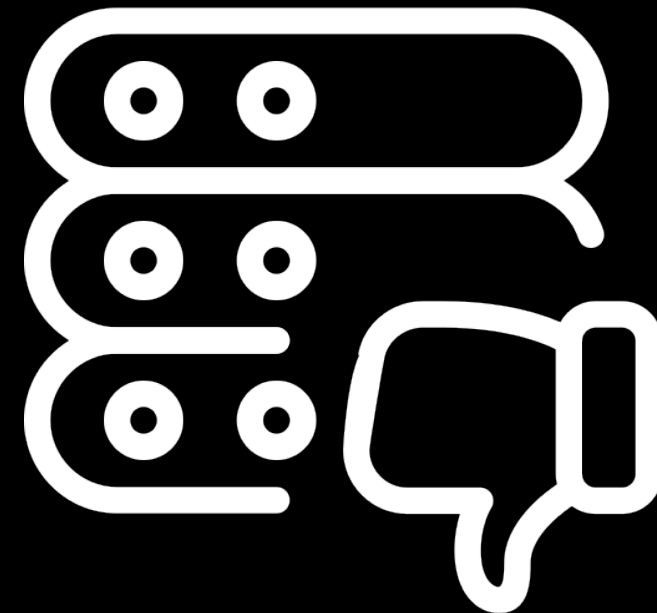
Dead Code

Dead Code

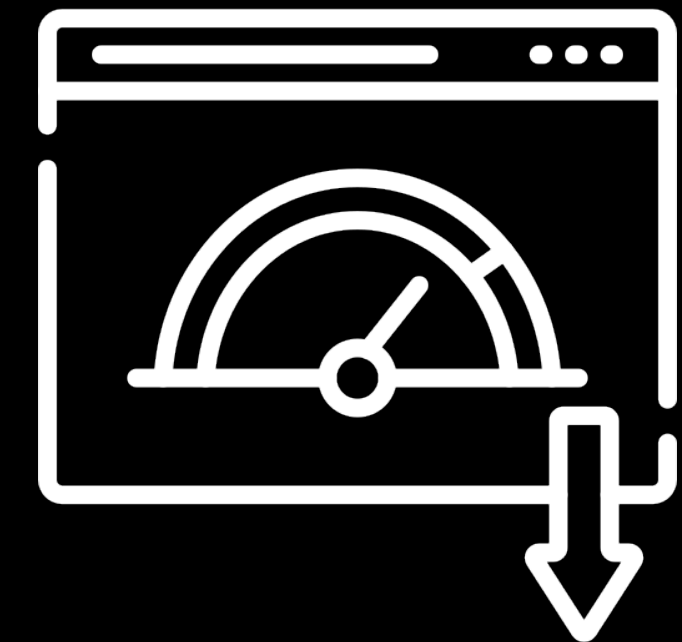
1.2 Dead Code의 문제점



시스템을 이해 / 유지보수
하기 어렵게 만듭니다.



성능 / 보안에
악영향을 줍니다.



컴파일 / 테스트 속도를
지연시켜 전체 개발 속도
저하시킵니다.

1.3 Dead Code 검출 방법

정적 분석 도구(static analysis)로 검출

```
public class SampleClass {
    private void methodA() {
    }
    private void methodB() {
        methodA();
    }
}
```

```
private int methodA() {
    int value = methodB1();
    value = methodB2();
    return value;
}
private int methodB1() { return 1; }
private int methodB2() { return 2; }
```

진짜 실행하여(Runtime Analysis) 디버깅 또는 로그를 찍어 확인

```
public class SampleController {
    @Get("/new")
    public void methodA() {
        methodB();
    }
    private void methodB() { }
}
```

● 로그 추가

```
public void methodC(Data data) {
    if (data.type == TYPE.A) { methodA(); }
    else if (data.type == TYPE.B) { methodB(); }
    else if (data.type == TYPE.C) {
        methodC();
    }
}
private void methodC() { }
```

● 로그 추가

2. What is Scavenger?

2. What is Scavenger?



- Scavenger는 런타임 Dead Code 분석 도구입니다.
- 디버깅 또는 로그를 추가하지 않고 메서드 호출이 확인 가능합니다.
- 메서드 호출 기록을 수집하여 이를 시각화해 유저에게 보여줍니다.
- Java agent 방식으로 손쉽게 사용 가능합니다.

2.1 Scavenger 데모

```
package com.example.scavengerdemo;

....

@SpringBootApplication
@RestController
public class DemoApplication {

    public static void main(String[] args) { SpringApplication.run(DemoApplication.class, args); }

    @GetMapping("/invoke")
    public String invoke() { success(); }

    public String success() { return "success"; }

    @GetMapping("/not-invoke")
    public String notInvoke() { return "notInvoke"; }

}
```

2.1 Scavenger 데모

1. scavenger agent, 설정 파일과 함께 demo.jar 실행

```
taeyeon ~/scavenger-demo ls -l
total 54624
-rw-r--r--@ 1 taeyeon  staff  17645868  1 16 01:40 demo.jar
-rw-r--r--@ 1 taeyeon  staff  10313476  1 16 00:15 scavenger-agent.jar
-rw-r--r--@ 1 taeyeon  staff      202  2 23 14:49 scavenger.conf

taeyeon ~/scavenger-demo java -Dscavenger.configuration=scavenger.conf -javaagent:scavenger-agent.jar -jar demo.jar
2월 23, 2023 6:24:59 오후 com.navercorp.scavenger.javaagent.ScavengerAgent premain
정보: [scavenger] scavenger agent version 1.0.2 is starting...
2월 23, 2023 6:24:59 오후 com.navercorp.scavenger.javaagent.util.ConfigUtils locateConfigFile
정보: [scavenger] config file found explicitly: scavenger.conf
```

2.1 Scavenger 데모

2. Scavenger 배너와 함께 agent 설정 로그 확인



SCAVENGER

```
agent version :: 1.0.2
config location :: /Users/taeyeon/scavenger-demo/scavenger.conf
  api key :: f0c7c708-388e-4538-a45e-93c1abe9e76e
  server url :: http://localhost:9090
  app name :: demo
  app version :: unspecified
  environment :: <default>
  package :: com.example.scavengerdemo
  method visibility :: protected
  exclude constructors :: false
  exclude setters, getters :: false
  hostname :: AL01978856.local
async code base scan mode :: false
legacy compatibility mode :: false
  force interval seconds :: 10
```

2.1 Scavenger 데모

3. agent가 스캔한 메서드 및 호출 내역에 대한 로그 확인

```

2월 23, 2023 6:25:05 오후 com.navercorp.scavenger.javaagent.collecting.CodeBaseScanner scanMethods
정보: [scavenger] com.example.scavengerdemo.DemoApplication() is scanned
2월 23, 2023 6:25:05 오후 com.navercorp.scavenger.javaagent.collecting.CodeBaseScanner scanMethods
정보: [scavenger] com.example.scavengerdemo.DemoApplication.main(java.lang.String[]) is scanned
2월 23, 2023 6:25:05 오후 com.navercorp.scavenger.javaagent.collecting.CodeBaseScanner scanMethods
정보: [scavenger] com.example.scavengerdemo.DemoApplication.invoke() is scanned
2월 23, 2023 6:25:05 오후 com.navercorp.scavenger.javaagent.collecting.CodeBaseScanner scanMethods
정보: [scavenger] com.example.scavengerdemo.DemoApplication.notInvoke() is scanned
2월 23, 2023 6:25:05 오후 com.navercorp.scavenger.javaagent.collecting.CodeBaseScanner scanMethods
정보: [scavenger] com.example.scavengerdemo.DemoApplication.success() is scanned
2월 23, 2023 6:25:05 오후 com.navercorp.scavenger.javaagent.collecting.CodeBaseScanner scan
정보: [scavenger] codebase(3a4bea3073be189a21decabc4809df793062089403e5d7fe5ba53e239b55f957) scanned in 353 ms: 5 methods

```

```

2023-02-23 18:49:00.593 INFO 22996 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port(s): 8080 (http) with context path ''
2023-02-23 18:49:00.599 INFO 22996 --- [main] c.example.scavengerdemo.DemoApplication : Started DemoApplication in 0.93 seconds (JVM running for 6.848)
2023-02-23 18:49:09.508 INFO 22996 --- [enger scheduler] c.n.s.javaagent.publishing.Publisher : [scavenger] polling dynamic config
2023-02-23 18:49:09.514 INFO 22996 --- [enger scheduler] c.n.s.javaagent.publishing.Publisher : [scavenger] trying to resolve collector url by accessing http://localhost:9090/javaagent/v5/initConfig
2023-02-23 18:49:09.965 INFO 22996 --- [enger scheduler] c.n.s.javaagent.publishing.Publisher : [scavenger] resolved collector url is localhost:9090
2023-02-23 18:49:09.966 INFO 22996 --- [enger scheduler] c.n.s.javaagent.publishing.GrpcClient : [scavenger] creating new grpc client. host is localhost:9090
2023-02-23 18:49:10.419 INFO 22996 --- [enger scheduler] c.n.s.javaagent.publishing.Publisher : [scavenger] publishing codebase: 5 methods
2023-02-23 18:49:10.433 INFO 22996 --- [enger scheduler] c.n.s.javaagent.publishing.Publisher : [scavenger] codebase published
2023-02-23 18:49:10.447 INFO 22996 --- [enger scheduler] c.n.s.javaagent.publishing.Publisher : [scavenger] publishing invocation data: 2 invocations
2023-02-23 18:49:10.460 INFO 22996 --- [enger scheduler] c.n.s.javaagent.publishing.Publisher : [scavenger] invocation data published

```

2.1 Scavenger 데모

4. Dead Code 분석을 위해 Scavenger UI 접근

Scavenger
Dead Code Analysis Tool

Learn more

워크스페이스 1

demo	
ID	1

삭제

+ 워크스페이스 생성

Total 1 < 1 >

2.1 Scavenger 데모

5. 설정 파일은 UI에서 쉽게 생성 가능

The screenshot shows the Scavenger web interface. A modal dialog titled "Scavenger 설정 파일 생성" (Generate Scavenger Settings File) is open, allowing users to configure settings for a specific agent. The background shows a dashboard with a table of agents and a sidebar with navigation options like "대시보드", "스냅샷 (0/20)", and "관리".

Scavenger 설정 파일 생성

- serverUrl: http://localhost:9090
- apiKey: f0c7c708-388e-4538-a45e-93c1abe9e76e
- appName: demo
- 수집 대상 packages: com.navercorp.foo, com.navercorp.bar
- codeBase 위치: /home1/irteam/service/tomcat/webapps/ROOT/WEB-INF, /home1/irteam/apps/tomcat/lib
- 수집 대상 클래스 annotation: @org.springframework.stereotype.Controller, @org.springframework.stereotype.Service
- 수집 대상 method visibility: public protected package-private private
- 수집 대상 생성자 제외여부: 생성자를 수집 제외시 체크
- 수집 대상 Getter / Setter 제외 여부: Getter / Setter를 수집 제외시 체크
- environment 구분자 (권장): dev
- version 구분자: 0.0.1
- 수집 제외 package: com.navercorp.baz, com.navercorp.foo
- codeBase 수집 async 모드 활성화: Scavenger가 정상 동작한 것을 확인한 뒤에, scavenger에 의한 부팅 지연을 해소하고자 한다면 체크

Buttons: 다운로드, 취소

2.1 Scavenger 데모

6. 특점 시점 이후 호출 내역을 확인하기 위한 스냅샷 생성

The screenshot shows the Scavenger application interface. On the left sidebar, the '스냅샷 (0/20)' menu item is highlighted with a red box and a circled '1'. In the main area, the '스냅샷 목록' section has a '+ 생성하기' button highlighted with a red box and a circled '2'. Below this is a table with columns: ID, 이름, 애플리케이션, 환경, 포함 패키지, and 처리 기간. The table currently shows 'No Data'. A modal dialog titled '스냅샷 생성' is open, highlighted with a red box and a circled '3'. The dialog contains the following fields:

- 이름 *: demo
- 애플리케이션 *: demo
- 환경 *: <default>
- 포함 패키지: com.navercorp.** , **.Controller.**
- 처리 기간: 이후 호출 건만 처리

At the bottom right of the modal, there are two buttons: '생성' (Generate) and '취소' (Cancel).

2.1 Scavenger 데모

7. 생성한 스냅샷에서 메서드 호출 내역 확인

프로젝트 파일 트리

메서드 시그니처와 호출 여부

시그니처 (5)	마지막 호출 시간	메서드 수	사용률
<code>c.e.s.DemoApplication.DemoApplication()</code>	2023.02.23 18:48	1	100%
<code>c.e.s.DemoApplication.invoke()</code>	-	1	0%
<code>c.e.s.DemoApplication.main(String[])</code>	2023.02.23 18:48	1	100%
<code>c.e.s.DemoApplication.notInvoke()</code>	-	1	0%
<code>c.e.s.DemoApplication.success()</code>	-	1	0%

2.1 Scavenger 데모

8. API 호출 후 invoke(), success() 호출 확인을 위해 스냅샷 재생성

```
taeyeon ~ curl http://localhost:8080/invoke  
success%
```

```
2023-02-23 19:11:46.823 INFO 22996 --- [nio-8080-exec-1] c.n.s.j.collecting.InvocationTracker : [scavenger] method public java.lang.String com.example.scavengerdemo.DemoApp  
lication.invoke() is invoked - 8038a7a50305c764f075340b8b558041  
2023-02-23 19:11:46.824 INFO 22996 --- [nio-8080-exec-1] c.n.s.j.collecting.InvocationTracker : [scavenger] method public java.lang.String com.example.scavengerdemo.DemoApp  
lication.success() is invoked - 97573a6059fd3c39774207bd8e65bdc3  
2023-02-23 19:11:49.503 INFO 22996 --- [enger scheduler] c.n.s.javaagent.publishing.Publisher : [scavenger] polling dynamic config  
2023-02-23 19:11:49.543 INFO 22996 --- [enger scheduler] c.n.s.javaagent.publishing.Publisher : [scavenger] publishing invocation data: 2 invocations  
2023-02-23 19:11:49.547 INFO 22996 --- [enger scheduler] c.n.s.javaagent.publishing.Publisher : [scavenger] invocation data published
```

com.example.scavengerdemo.DemoApplication

com.example.scavengerdemo 5, 40%

• DemoApplication 5, 40%

• DemoApplication()

• invoke()

• main(String[])

• notInvoke()

• success()

2.1 Scavenger 데모

9. 재생성한 스냅샷에서 invoke(), success() 호출 확인

com.example.scavengerdemo.DemoApplication

시그니처 (5)

시그니처 (5)	마지막 호출 시간	메서드 수	사용률
c.e.s.DemoApplication.DemoApplication()	2023.02.23 18:48	1	100%
c.e.s.DemoApplication.invoke()	2023.02.23 19:11	1	100%
c.e.s.DemoApplication.main(String[])	2023.02.23 18:48	1	100%
c.e.s.DemoApplication.notInvoke()	-	1	0%
c.e.s.DemoApplication.success()	2023.02.23 19:11	1	100%

2.1 Scavenger 데모

10. Dead Code인 notInvoke() 삭제

```
@SpringBootApplication
@RestController
public class DemoApplication {

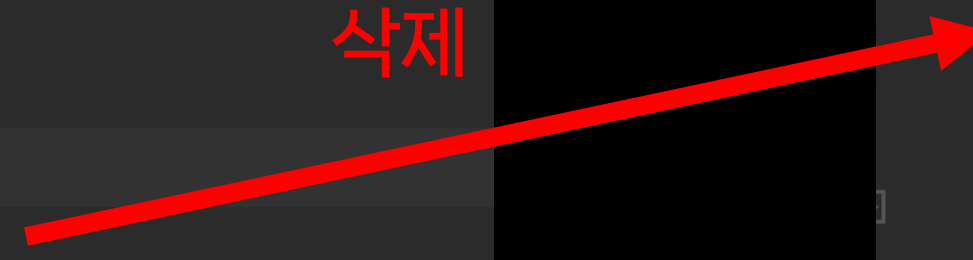
    no usages
    public static void main(String[] args) { SpringApplication

    no usages
    @GetMapping("/invoke")
    public String invoke() {
        return success();
    }

    no usages
    @GetMapping("/not-invoke")
    public String notInvoke() {
        return "notInvoke";
    }

    1 usage
    public String success() { return "success"; }
}
```

삭제



```
@SpringBootApplication
@RestController
public class DemoApplication {

    no usages
    public static void main(String[] args) { SpringAp

    no usages
    @GetMapping("/invoke")
    public String invoke() {
        return success();
    }

    1 usage
    public String success() { return "success"; }
}
```

2.2 Scavenger

이 거대한 시스템을
새로운 아키텍처로
구현해야 한다고요?

사용되는 부분만
확인할 수 있는 툴
하나 만들죠.



네이버페이 Next 프로젝트 회의

2.2 Scavenger



- Codekvast 라는 오픈소스 도구를 기반으로 개발 시작
- 하이에나 같은 생태계 청소 동물을 총칭하는 Scavenger의 탄생
- 지식 iN, 블로그, 네이버 페이, 시리즈온, 네이버 뮤직... 약 80여개의 서비스에서 사용 중

미사용 테이블 코드 삭제 #14075 #14478

Merged merged 17 commits into from on 14 Jul 2022

Conversation 5 Commits 17 Checks 0 Files changed 81 **+28 -3,611**

2.2 Scavenger



scavenger

- 성능 최적화, 설정 단순화를 위해 전체 재개발
 - 압축을 풀지 않고 클래스 로딩을 하지 않고 사용자 코드를 읽음
 - H2, MySQL, Cloud Native 환경의 분산 DB 시스템인 Vitess를 Database로 지원
 - 어노테이션 기준 추적 범위 설정, 생성자 제외 기능 등 여러 편의 기능 추가
- 성능 오버헤드가 높지 않음(약 31만 2천개의 메서드가 있는 대형 네이버 서비스를 대상으로 측정)
 - Java agent 실행 시 초기 1회 전체 메서드를 스캔하는데 8.7s (설정에 따라 개선 가능)
 - 메서드 한번 호출 시 30ns

2.2 Scavenger



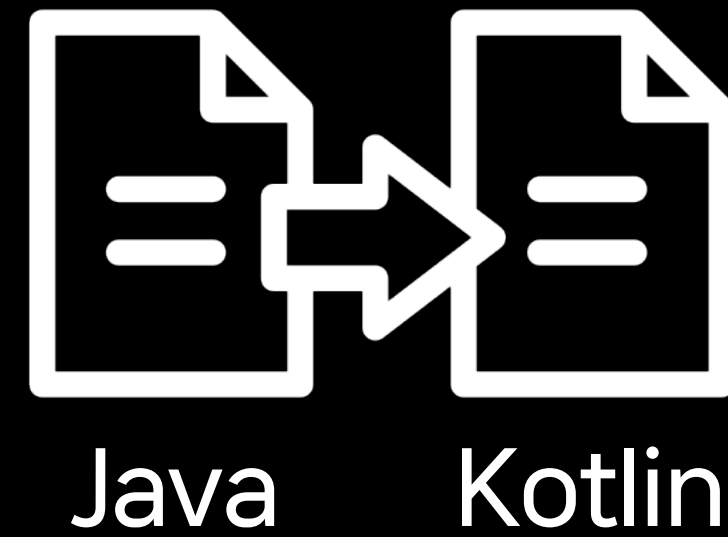
scavenger

- JVM 기반 언어(Java, Kotlin)만 지원
 - Python은 현재 베타 버전, 그 외 다양한 언어 지원 예정
- Java agent는 Maven Central에서 다운로드 가능
 - <https://repo1.maven.org/maven2/com/navercorp/scavenger/scavenger-agent-java/{VERSION}/scavenger-agent-java-{VERSION}.jar>
- Python agent는 pip로 설치 가능
 - `pip install scavenger-agent-python`
- GitHub Repository release를 통해 Collector, API 빌드 없이 사용 가능
 - <https://github.com/naver/scavenger/releases>

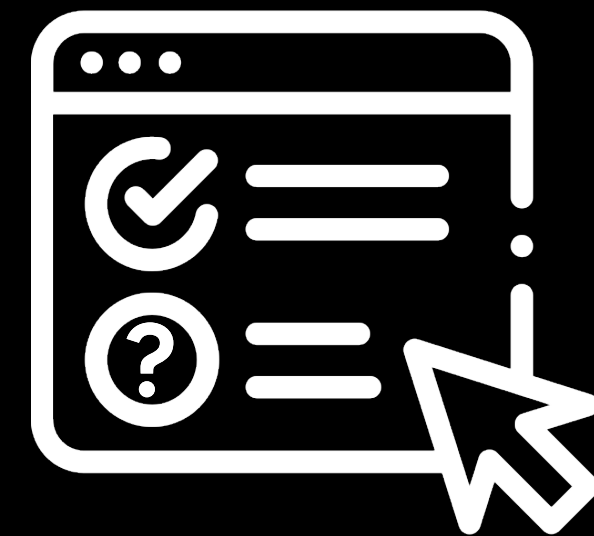
2.3 Scavenger 활용 예시



Dead Code 검출

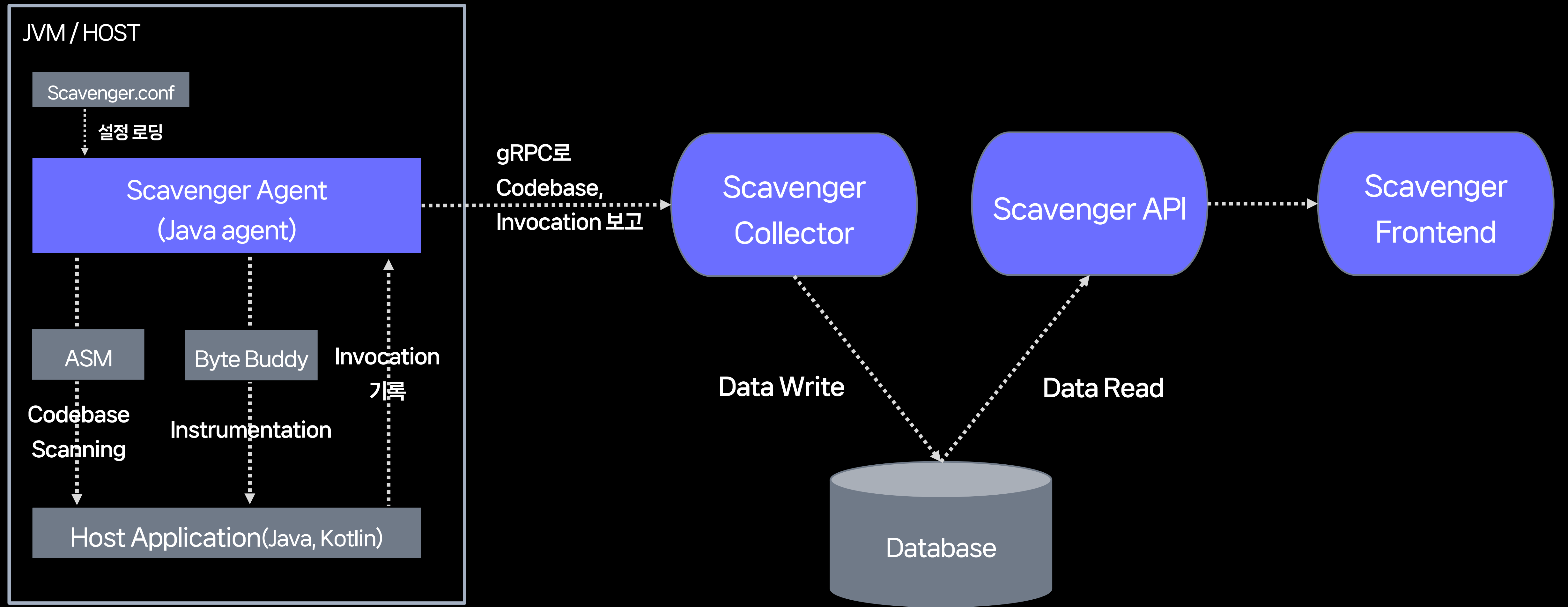


새로운 언어로 포팅 시
포팅 범위 확인



QA 진행 시, 기능성 테스트
부족 인지

2.4 Scavenger 아키텍처



3. Behind the Scenes

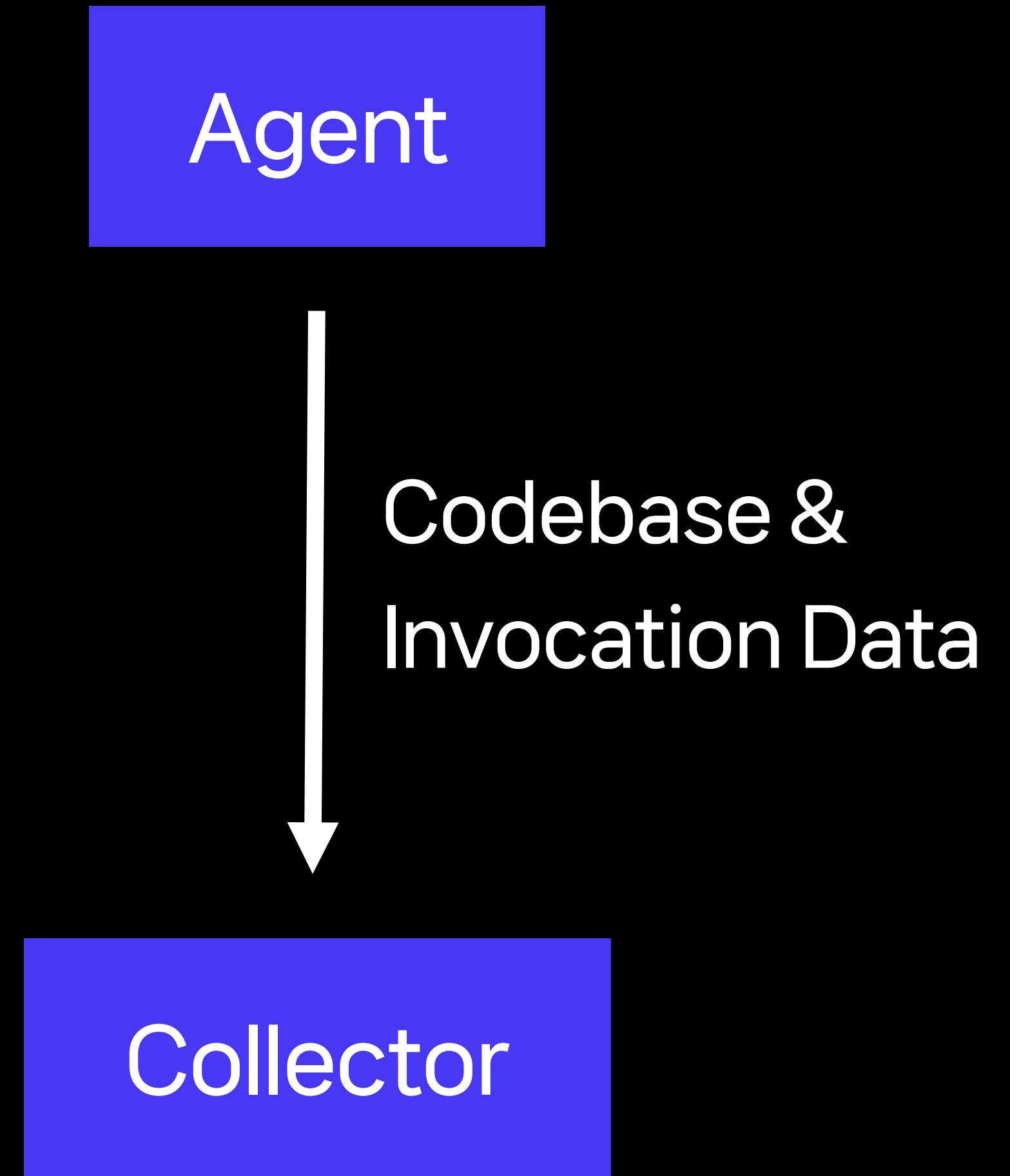
3.1 수집하는 데이터

Codebase

- 추적할 메서드 목록과 관련 정보
- 메서드 이름, parameter/return 타입 등

Invocation Data

- 호출된 메서드 목록과 호출 시각



코드베이스 탐색

3.2 Scavenger 설정 예시

- 아래와 같은 다양한 필터링 기준을 두어 추적 대상을 지정할 수 있다
- 추적할 메서드 목록과 그 정보를 어떻게 추출할 수 있을까

```
packages=com.example.demo
annotations=@org.springframework.web.bind.annotation.Controller
additionalPackages=com.example.demo.additional
methodVisibility=protected
excludeConstructors=true
excludeGetterSetter=true
```

3.3 자바 실행 예시

- 자바 프로그램은 아래와 같이 컴파일 과정을 거친 후 실행

```
> javac Hello.java
> ls
Hello.class Hello.java
> java Hello
Hello, world!
```


3.3 자바 실행 예시

- 실제 생성되는 파일을 살펴보면 이렇게 생겼다

```
public class Test {  
    public int test(int a) {  
        return a + a;  
    }  
}
```

Test.java

javac



```
> xxd Test.class  
00000000: cafe babe 0000 003b 000f 0a00 0200 0307 .....;  
00000010: 0004 0c00 0500 0601 0010 6a61 7661 2f6c .....java/l  
00000020: 616e 672f 4f62 6a65 6374 0100 063c 696e ang/Object...<in  
00000030: 6974 3e01 0003 2829 5607 0008 0100 0454 it>...()V.....T  
00000040: 6573 7401 0004 436f 6465 0100 0f4c 696e est...Code...Lin  
00000050: 654e 756d 6265 7254 6162 6c65 0100 0474 eNumberTable...t  
00000060: 6573 7401 0004 2849 2949 0100 0a53 6f75 est...(I)I...Sou  
00000070: 7263 6546 696c 6501 0009 5465 7374 2e6a rceFile...Test.j  
00000080: 6176 6100 2100 0700 0200 0000 0000 0200 ava.!.....  
00000090: 0100 0500 0600 0100 0900 0000 1d00 0100 .....  
000000a0: 0100 0000 052a b700 01b1 0000 0001 000a .....*.....  
000000b0: 0000 0006 0001 0000 0001 0001 000b 000c .....  
000000c0: 0001 0009 0000 001c 0002 0002 0000 0004 .....  
000000d0: 1b1b 60ac 0000 0001 000a 0000 0006 0001 ..`.....  
000000e0: 0000 0003 0001 000d 0000 0002 000e .....
```

Test.class

3.4 클래스 파일 구조

- 이 클래스 파일의 구조는 자바 스펙에 명시되어 있음!

4.1. The ClassFile Structure

A class file consists of a single ClassFile structure:

```
ClassFile {  
    u4          magic;  
    u2          minor_version;  
    u2          major_version;  
    u2          constant_pool_count;  
    cp_info     constant_pool[constant_pool_count-1];  
    u2          access_flags;  
    u2          this_class;  
    u2          super_class;  
    u2          interfaces_count;  
    u2          interfaces[interfaces_count];  
    u2          fields_count;  
    field_info  fields[fields_count];  
    u2          methods_count;  
    method_info methods[methods_count];  
    u2          attributes_count;  
    attribute_info attributes[attributes_count];  
}
```

3.4 클래스 파일 구조

- 실제 클래스 파일에서 메서드 정보를 직접 추출해 보자

4.1. The ClassFile Structure

A class file consists of a single ClassFile structure:

```
ClassFile {
    u4          magic;
    u2          minor_version;
    u2          major_version;
    u2          constant_pool_count;
    cp_info     constant_pool[constant_pool_count-1];
    u2          access_flags;
    u2          this_class;
    u2          super_class;
    u2          interfaces_count;
    u2          interfaces[interfaces_count];
    u2          fields_count;
    field_info  fields[fields_count];
    u2          methods_count;
    method_info methods[methods_count];
    u2          attributes_count;
    attribute_info attributes[attributes_count];
}
```

3.4 클래스 파일 구조

```

> xxd Test.class
00000000: cafe babe 0000 003b 000f 0a00 0200 0307 .....;.....
00000010: 0004 0c00 0500 0601 0010 6a61 7661 2f6c .....java/l
00000020: 616e 672f 4f62 6a65 6374 0100 063c 696e ang/Object...<in
00000030: 6974 3e01 0003 2829 5607 0008 0100 0454 it>...()V.....T
00000040: 6573 7401 0004 436f 6465 0100 0f4c 696e est...Code...Lin
00000050: 654e 756d 6265 7254 6162 6c65 0100 0474 eNumberTable...t
00000060: 6573 7401 0004 2849 2949 0100 0a53 6f75 est...(I)I...Sou
00000070: 7263 6546 696c 6501 0009 5465 7374 2e6a rceFile...Test.j
00000080: 6176 6100 2100 0700 0200 0000 0000 0200 ava!.....
00000090: 0100 0500 0600 0100 0900 0000 1d00 0100 .....
000000a0: 0100 0000 052a b700 01b1 0000 0001 000a .....*.....
000000b0: 0000 0006 0001 0000 0001 0001 000b 000c .....
000000c0: 0001 0009 0000 001c 0002 0002 0000 0004 .....
000000d0: 1b1b 60ac 0000 0001 000a 0000 0006 0001 ..`.....
000000e0: 0000 0003 0001 000d 0000 0002 000e .....

```

3.4 클래스 파일 구조

```
> xxd Test.class
00000000: cafe babe 0000 003b 000f 0a00 0200 0307
00000010: 0004 0c00 0500 0601 0010 6a61 7661 2f6c
00000020: 616e 672f 4f62 6a65 6374 0100 063c 696e
00000030: 6974 3e01 0003 2829 5607 0008 0100 0454
00000040: 6573 7401 0004 436f 6465 0100 0f4c 696e
00000050: 654e 756d 6265 7254 6162 6c65 0100 0474
00000060: 6573 7401 0004 2849 2949 0100 0a53 6f75
00000070: 7263 6546 696c 6501 0009 5465 7374 2e6a
00000080: 6176 6100 2100 0700 0200 0000 0000 0200
00000090: 0100 0500 0600 0100 0900 0000 1d00 0100
000000a0: 0100 0000 052a b700 01b1 0000 0001 000a
000000b0: 0000 0006 0001 0000 0001 0001 000b 000c
000000c0: 0001 0009 0000 001c 0002 0002 0000 0004
000000d0: 1b1b 60ac 0000 0001 000a 0000 0006 0001
000000e0: 0000 0003 0001 000d 0000 0002 000e
```

```
.....;.....
.....java/l
ang/Object...<in
it>...()V....T
est...Code...Lin
eNumberTable...t
est...(I)I...Sou
rceFile...Test.j
ava!.....
.....
.....*.....
.....
.....
.....`.....
.....
```

3.4 클래스 파일 구조

```
> xxd Test.class
00000000: cafe babe 0000 003b 000f 0a00 0200 0307 .....;.....
00000010: 0004 0c00 0500 0601 0010 6a61 7661 2f6c .....java/l
00000020: 616e 672f 4f62 6a65 6374 0100 063c 696e ang/Object...<in
00000030: 6974 3e01 0003 2829 5607 0008 0100 0454 it>...()V.....T
00000040: 6573 7401 0004 436f 6465 0100 0f4c 696e est...Code...Lin
00000050: 654e 756d 6265 7254 6162 6c65 0100 0474 eNumberTable...t
00000060: 6573 7401 0004 2849 2949 0100 0a53 6f75 est...(I)I...Sou
00000070: 7263 6546 696c 6501 0009 5465 7374 2e6a rceFile...Test.j
00000080: 6176 6100 2100 0700 0200 0000 0000 0200 ava!.....
00000090: 0100 0500 0600 0100 0900 0000 1d00 0100 .....
000000a0: 0100 0000 052a b700 01b1 0000 0001 000a .....*.....
000000b0: 0000 0006 0001 0000 0001 000b 000c .....
000000c0: 0001 0009 0000 001c 0002 0002 0000 0004 .....
000000d0: 1b1b 60ac 0000 0001 000a 0000 0006 0001 ..`.....
000000e0: 0000 0003 0001 000d 0000 0002 000e .....
```

Methods

3.4 클래스 파일 구조

```
> xxd Test.class
00000000: cafe babe 0000 003b 000f 0a00 0200 0307 .....;.....
00000010: 0004 0c00 0500 0601 0010 6a61 7661 2f6c .....java/l
00000020: 616e 672f 4f62 6a65 6374 0100 063c 696e ang/Object...<in
00000030: 6974 3e01 0003 2829 5607 0008 0100 0454 it>...()V.....T
00000040: 6573 7401 0004 436f 6465 0100 0f4c 696e est...Code...Lin
00000050: 654e 756d 6265 7254 6162 6c65 0100 0474 eNumberTable...t
00000060: 6573 7401 0004 2849 2949 0100 0a53 6f75 est...(I)I...Sou
00000070: 7263 6546 696c 6501 0009 5465 7374 2e6a rceFile...Test.j
00000080: 6176 6100 2100 0700 0200 0000 0000 0200 ava.!.....
00000090: 0100 0500 0600 0100 0900 0000 1d00 0100 .....
000000a0: 0100 0000 <init>() 01b1 0000 0001 000a .....*.....
000000b0: 0000 0006 0001 0000 0001 0001 000b 000c .....
000000c0: 0001 0009 0000 001c 0002 0002 0000 0004 .....test()
000000d0: 1b1b 60ac 0000 0001 000a 0000 0006 0001 ..`.....
000000e0: 0000 0003 0001 000d 0000 0002 000e .....
```

3.4 클래스 파일 구조

```

> xxd Test.class
00000000: cafe babe 0000 003b 000f 0a00 0200 0307 .....;.....
00000010: 0004 0c00 0500 0601 0010 6a61 7661 2f6c .....java/l
00000020: 616e 672f 4f62 6a65 6374 0100 063c 696e ang/Object...<in
00000030: 6974 3e01 0003 2829 5607 0008 0100 0454 it>...()V.....T
00000040: 6573 7401 0004 436f 6465 0100 0f4c 696e est...Code...Lin
00000050: 654e 756d 6265 7254 6162 6c65 0100 0474 eNumberTable...t
00000060: 6573 7401 0004 2849 2949 0100 0a53 6f75 est...(I)I...Sou
00000070: 7263 6546 696c 6501 0009 5465 7374 2e6a rceFile...Test.j
00000080: 6176 6100 2100 0700 0200 0000 0000 0200 ava!.....
00000090: 0100 0500 0600 0100 0900 0000 1d00 0100 .....
000000a0: 0100 0000 052a b700 01b1 0000 0001 000a .....*.....
000000b0: 0000 0006 0001 0000 0001 0001 000b 000c .....
000000c0: 0001 0009 0000 001c 0002 0002 0000 0004 .....
000000d0: 1b1b 60ac 0000 0001 000a 0000 0006 0001 .. \ .....
000000e0: 0000 0003 0001 000d 0000 0002 000e .....

```


3.4 클래스 파일 구조

```
> xxd Test.class
00000000: cafe babe 0000 003b 000f 0a00 0200 0307 .....;.....
00000010: 0004 0c00 0500 0601 0010 6a61 7661 2f6c .....java/l
00000020: 616e 672f 4f62 6a65 6374 0100 063c 696e ang/Object...<in
00000030: 6974 3e01 0003 2829 5607 0008 0100 0454 it>...()V.....T
00000040: 6573 7401 0004 436f 6465 0100 0f4c 696e est...Code...Lin
00000050: 654e 756d 6265 7254 6162 6c65 0100 0474 eNumberTable...t
00000060: 6573 7401 0004 2849 2949 0100 0a53 6f75 est...(I)I...Sou
00000070: 7263 6546 696c 6501 0009 5465 7374 2e6a rceFile...Test.i
00000080: 6176 6100 2100 0700 0200 0000 0200 0000 ava...!.....
00000090: 0100 0500 0600 0100 0900 0000 1d00 0100 .....
000000a0: 0100 0000 052a b700 01b1 0000 0001 000a .....*.....
000000b0: 0000 0006 0001 0000 0001 0001 000b 000c .....
000000c0: 0001 0009 0000 001c 0002 0002 0000 0004 .....
000000d0: 1b1b 60ac 0000 0001 000a 0000 0006 0001 .. .....
000000e0: 0000 0003 0001 000d 0000 0002 000e .....
```

name_index

descriptor_index



3.4 클래스 파일 구조

```

> xxd Test.class
00000000: cafe babe 0000 003b 000f 0a00 0200 0307 .....;.....
00000010: 0004 0c00 0500 0601 0010 6a61 7661 2f6c .....java/l
00000020: 616e 672f 4f62 6a65 6374 0100 063c 696e ang/Object...<in
00000030: 6974 3e01 0003 2829 5607 0008 0100 0454 it>...()V.....T
00000040: 6573 7401 0004 636f 6465 0100 0f4c 696e est...Code...Lin
00000050: 654e 756d 6265 7254 6162 6c65 0100 0474 eNumberTable...t
00000060: 6573 7401 0004 2849 2949 0100 0a53 6f75 est...(I)I...Sou
00000070: 7263 6546 696c 6501 0009 5465 7374 2e6a rceFile...Test.j
00000080: 6176 6100 2100 0700 0200 0000 0000 0200 ava.!.....
00000090: 0100 0500 0600 0100 0900 0000 1d00 0100 .....
000000a0: 0100 0000 052a b700 01b1 0000 0001 000a .....*.....
000000b0: 0000 0006 0001 0000 0001 0001 000b 000c .....
000000c0: 0001 0009 0000 001c 0002 0002 0000 0004 .....
000000d0: 1b1b 60ac 0000 0001 000a 0000 0006 0001 ..`.....
000000e0: 0000 0003 0001 000d 0000 0002 000e .....

```

Constant Pool

3.4 클래스 파일 구조

```

> xxd Test.class
00000000: cafe babe 0000 003b 000f 0a00 0200 0307 .....;.....
00000010: 0004 0c00 0500 0601 0010 6a61 7661 2f6c .....java/l
00000020: 616e 672f 4f62 6a65 6374 0100 063c 696e ang/Object...<in
00000030: 6974 3e01 0003 2829 5607 0008 0100 0454 it>...()V.....T
00000040: 6573 7401 0004 436f 6465 0100 0f4c 696e est...Code...Lin
00000050: 654e 756d 6265 7254 6162 6c65 0100 0474 eNumberTable...t
00000060: 6573 7401 0004 2849 2949 0100 0a53 6f75 est...(I)I...Sou
00000070: 7263 6546 696c 6501 0009 5465 7374 2e6a rceFile...Test.j
00000080: 6176 6100 2100 0700 0200 0000 0000 0200 ava!.....
00000090: 0100 0500 0600 0100 0900 0000 1d00 0100 .....
000000a0: 0100 0000 052a b700 01b1 0000 0001 000a .....*.....
000000b0: 0000 0006 0001 0000 0001 0001 000b 000c .....
000000c0: 0001 0009 0000 001c 0002 0002 0000 0004 .....
000000d0: 1b1b 60ac 0000 0001 000a 0000 0006 0001 ..`.....
000000e0: 0000 0003 0001 000d 0000 0002 000e .....
  
```

3.4 클래스 파일 구조

```
> xxd Test.class
00000000: cafe babe 0000 003b 000f 0a00 0200 0307
00000010: 0004 0c00 0500 0601 0010 6a61 7661 2f6c
00000020: 616e 672f 4f62 6a65 6374 0100 063c 696e
00000030: 6974 3e01 0003 2829 5607 0008 0100 0454
00000040: 6572 7401 0004 4248 4445 0100 0f4c 696e
00000050: 6974 3e01 0003 2829 5607 0008 0100 0454
00000060: 6572 7401 0004 4248 4445 0100 0f4c 696e
00000070: 6974 3e01 0003 2829 5607 0008 0100 0454
00000080: 6572 7401 0004 4248 4445 0100 0f4c 696e
00000090: 6974 3e01 0003 2829 5607 0008 0100 0454
000000a0: 0100 0000 052a b700 01b1 0000 0001 000a
000000b0: 0000 0006 0001 0000 0001 0001 000b 000c
000000c0: 0001 0009 0000 001c 0002 0002 0000 0004
000000d0: 1b1b 60ac 0000 0001 000a 0000 0006 0001
000000e0: 0000 0003 0001 000d 0000 0002 000e
```

```
public class Test {
    public int test(int a) {
        return a + a;
    }
}
```

```
.....;.....
.....java/l
ang/Object...<in
it>...()V.....T
est...Code...Lin
eNumberTable...t
est...(I)I...Sou
rceFile...Test.j
ava.!.....
.....
.....*.....
.....
.....
.....
.....
.....
.....
```

3.5 메서드 정보 추출

- 이러한 방식으로 클래스 파일에서 메서드 정보를 추출할 수 있음
- Scavenger에서는 ASM 라이브러리를 이용
- 해당 정보를 이용하여 메서드 목록 필터링

3.6 정적으로 파싱하는 이유

- 클래스 로딩 등을 통해 동적으로 분석하는 경우 피분석 프로그램의 행위를 변경할 수도 있으므로 바이너리를 직접 파싱

```
public class Hello {  
    static {  
        System.out.println("Hello, world!");  
    }  
}
```

클래스를 로드하는 순간
static 구문이 실행됨!

3.7 클래스 파일 탐색

- 지정된 디렉터리에 있는 클래스 파일을 파싱하여 메서드 목록을 추출
- JAR / WAR의 경우에는 재귀적으로 압축 파일 내부를 탐색

```
if (file.getName().endsWith(".class") && containsPackageNameInPath(file.getName(), File.separator)) {  
    byte[] bytes = readAllBytes(stream);  
    if (isMagicValid(bytes)) {  
        ClassNode node = getNode(bytes);  
        if (filterClass(node)) {  
            classes.add(node);  
        }  
    }  
} else if (file.getName().endsWith(".jar") || file.getName().endsWith(".war")) {  
    JarInputStream innerStream = new JarInputStream(stream);  
    classes.addAll(scanClasses(innerStream));  
}
```

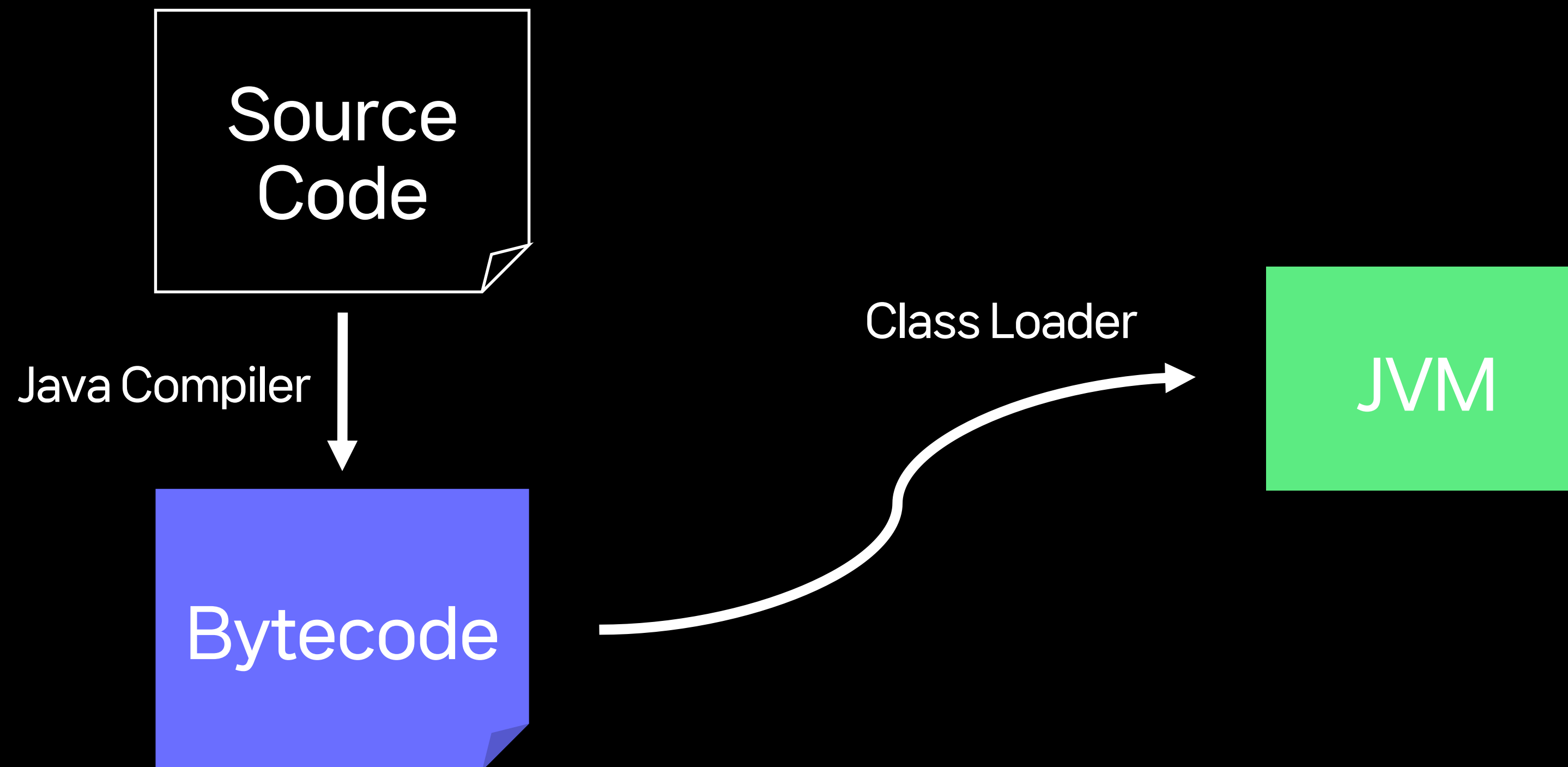
메서드 호출 추적

3.8 메서드 호출 추적

- 메서드 호출을 추적하기 위해서는 메서드 실행 전후에 임의로 코드 삽입이 가능해야 한다

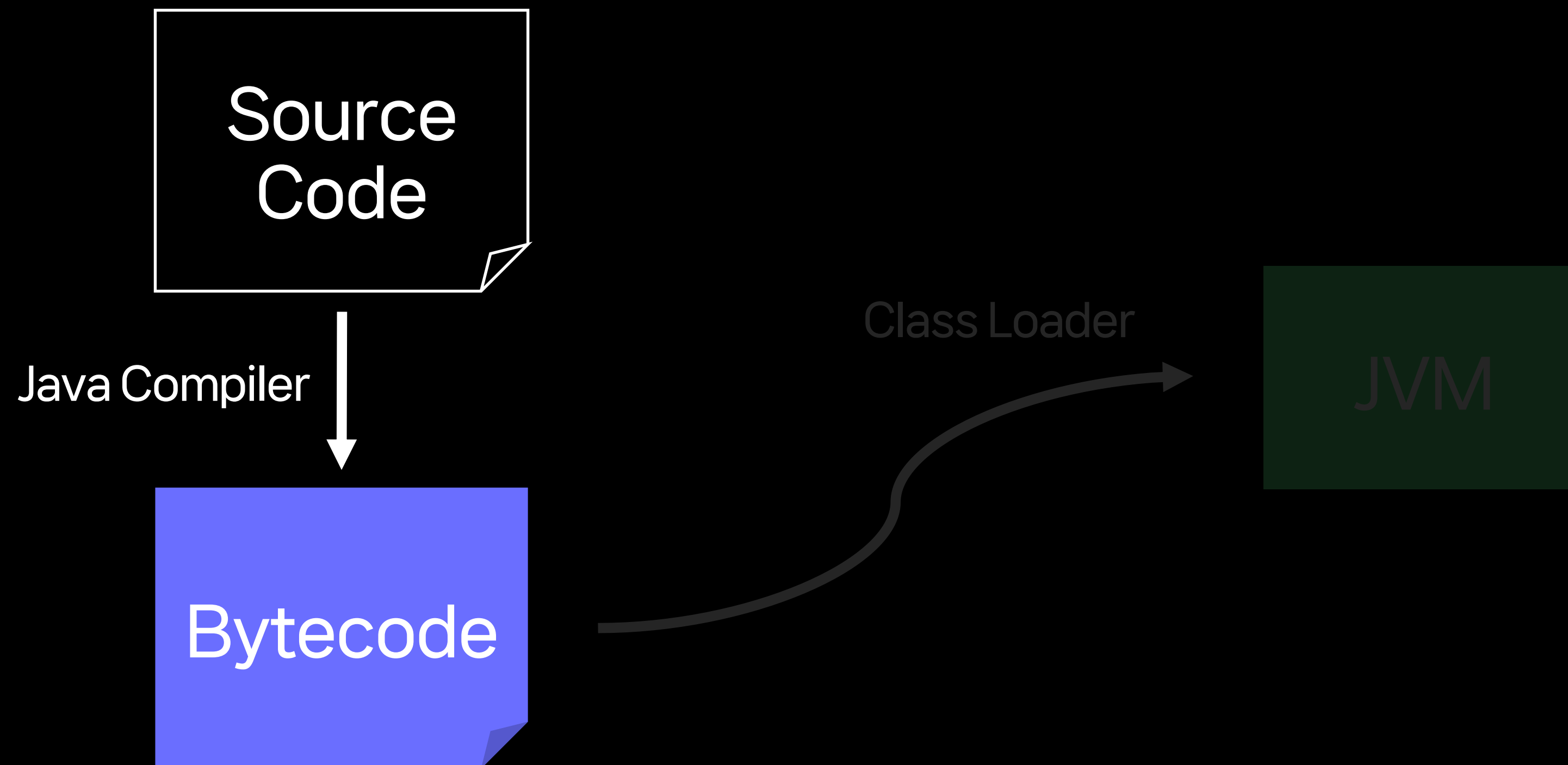
3.9 자바 실행 구조

- 전체적인 자바 실행 구조를 다시 한번 살펴보자



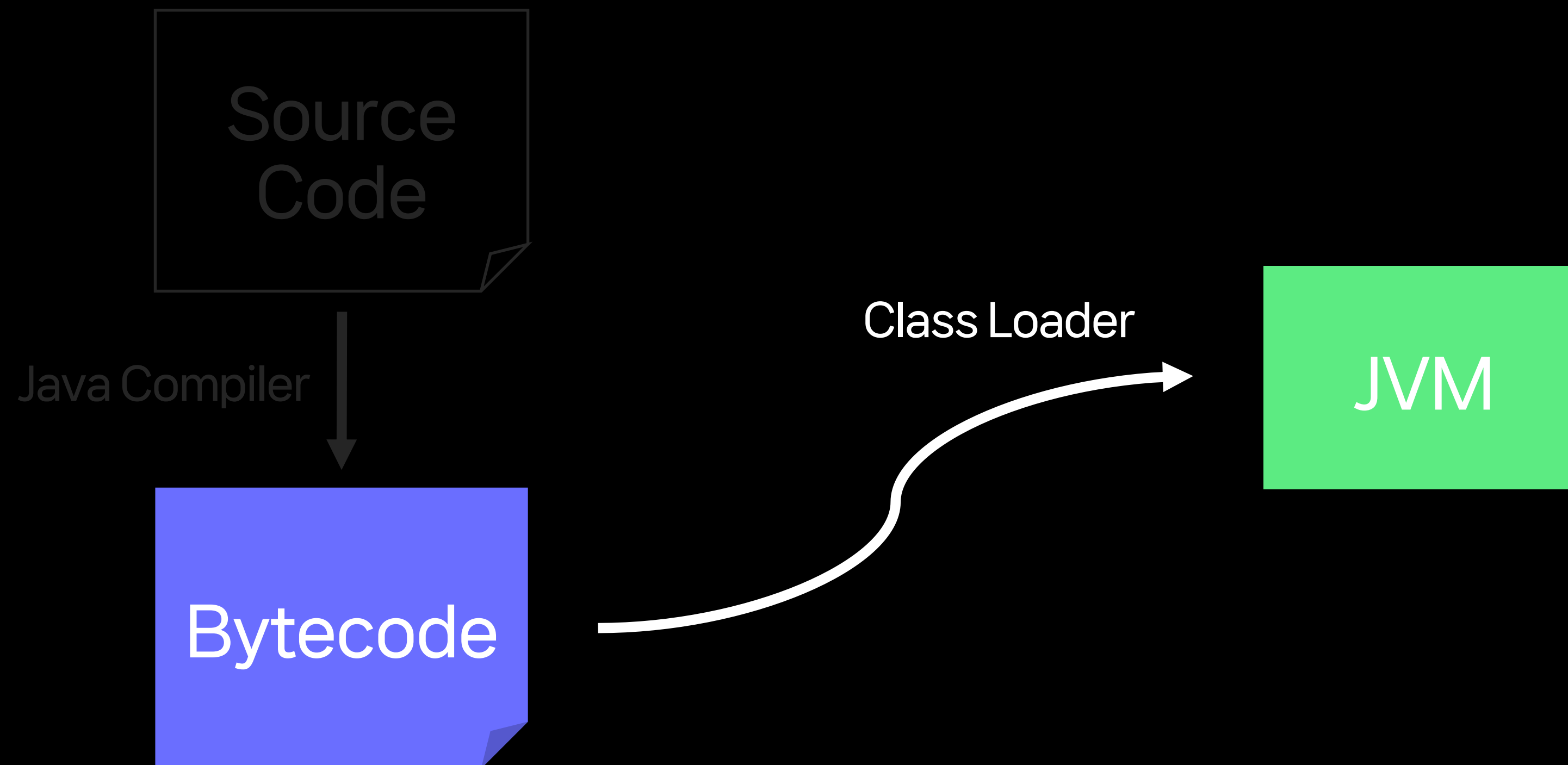
3.9 자바 실행 구조

- 소스 코드를 바이트 코드로 컴파일하는 과정과



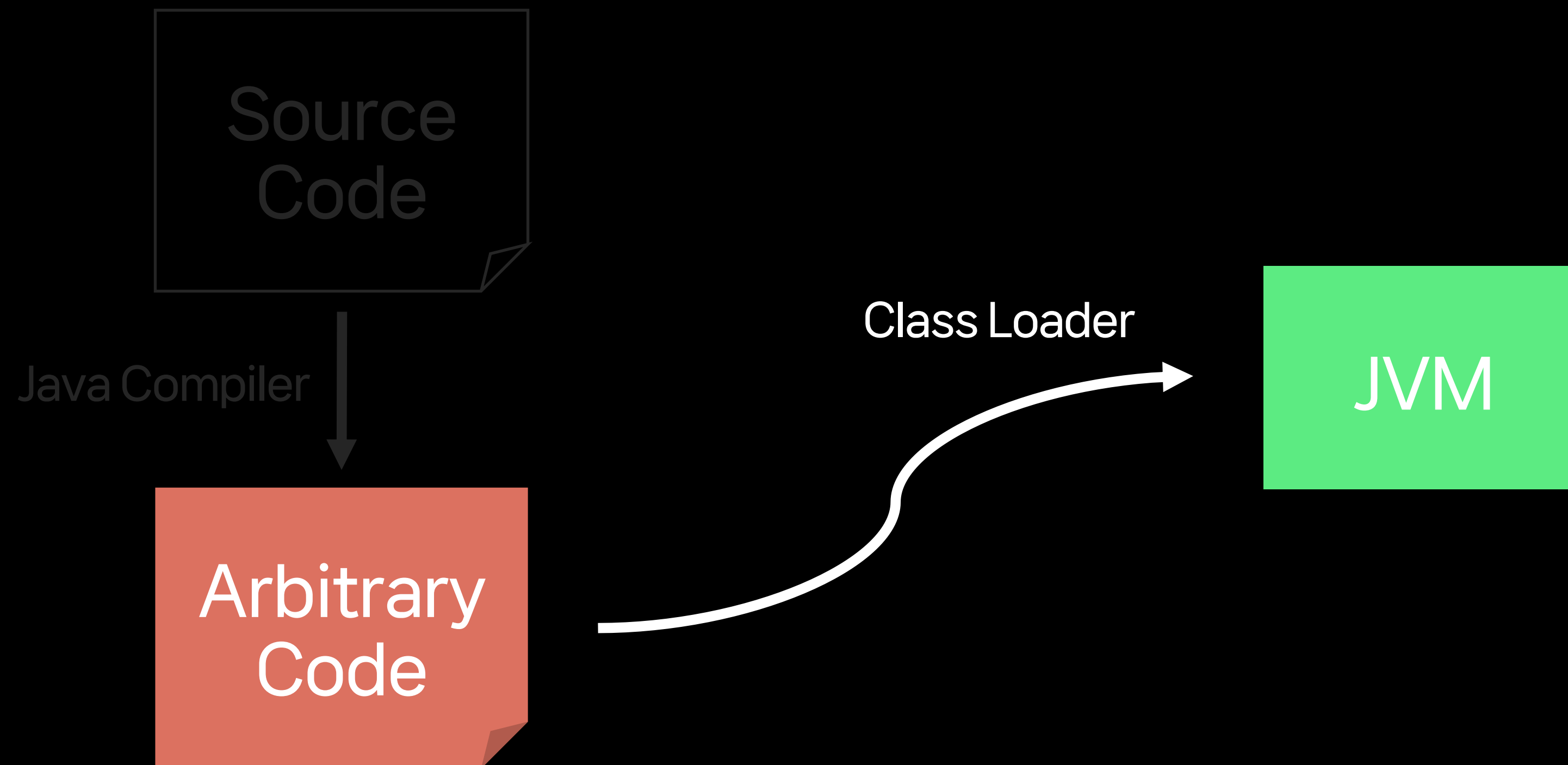
3.9 자바 실행 구조

- 컴파일된 바이트 코드를 JVM에 로딩하여 실행하는 과정으로 이루어진다



3.9 자바 실행 구조

- 이때 클래스 로더에게 우리가 원하는 임의의 코드를 넘겨줄 수 있다면?

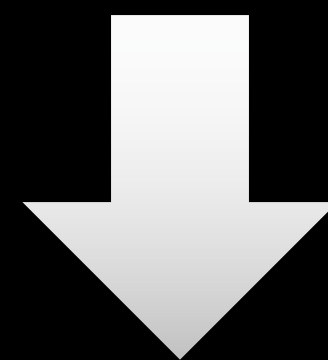


3.10 해당 방법의 장점

- 클래스나 메서드를 런타임에 바꿔 치울 수 있음
- 소스코드 없이 프로그램의 행위를 확장하거나 변경할 수 있음
- 패치를 플러그인처럼 적용할 수 있음

3.10 해당 방법의 장점

- 클래스나 메서드를 런타임에 바꿔 치울 수 있음
- 소스코드 없이 프로그램의 행위를 확장하거나 변경할 수 있음
- 패치를 플러그인처럼 적용할 수 있음



사용자의 코드 변경 없이 Scavenger를 적용할 수 있다!

3.11 직접 수정해 보기

- 바이트 코드를 직접 수정해서 프로그램의 behavior를 바꿔보자!

```
public class Test {  
    public int test(int a) {  
        return a + a;  
    }  
}
```



```
public class Test {  
    public int test(int a) {  
        return a * a;  
    }  
}
```


3.11 직접 수정해 보기

```

> xxd Test.class
00000000: cafe babe 0000 003b 000f 0a00 0200 0307 .....;.....
00000010: 0004 0c00 0500 0601 0010 6a61 7661 2f6c .....java/l
00000020: 616e 672f 4f62 6a65 6374 0100 063c 696e ang/Object...<in
00000030: 6974 3e01 0003 2829 5607 0008 0100 0454 it>...()V.....T
00000040: 6573 7401 0004 436f 6465 0100 0f4c 696e est...Code...Lin
00000050: 654e 756d 6265 7254 6162 6c65 0100 0474 eNumberTable...t
00000060: 6573 7401 0004 2849 2949 0100 0a53 6f75 est...(I)I...Sou
00000070: 7263 6546 696c 6501 0009 5465 7374 2e6a rceFile...Test.j
00000080: 6176 6100 2100 0700 0200 0000 0000 0200 ava.!.....
00000090: 0100 0500 0600 0100 0900 0000 1d00 0100 .....
000000a0: 0100 0000 052a b700 01b1 0000 0001 000a .....*.....
000000b0: 0000 0006 0001 0000 0001 0001 000b 000c .....
000000c0: 0001 0009 0000 001c 0002 0002 0000 0004 .....
000000d0: 1b1b 60ac 0000 0001 000a 0000 0006 0001 ..`.....
000000e0: 0000 0003 0001 000d 0000 0002 000e .....

```

3.11 직접 수정해 보기

```

> xxd Test.class
00000000: cafe babe 0000 003b 000f 0a00 0200 0307 .....;.....
00000010: 0004 0c00 0500 0601 0010 6a61 7661 2f6c .....java/l
00000020: 616e 672f 4f62 6a65 6374 0100 063c 696e ang/Object...<in
00000030: 6974 3e01 0003 2829 5607 0008 0100 0454 it>...()V.....T
00000040: 6573 7401 0004 436f 6465 0100 0f4c 696e est...Code...Lin
00000050: 654e 756d 6265 7254 6162 6c65 0100 0474 eNumberTable...t
00000060: 6573 7401 0004 2849 2949 0100 0a53 6f75 est...(I)I...Sou
00000070: 7263 6546 696c 6501 0009 5465 7374 2e6a rceFile...Test.j
00000080: 6176 6100 2100 0700 0200 0000 0000 0200 ava!.....
00000090: 0100 0500 0600 0100 0900 0000 1d00 0100 .....
000000a0: 0100 0000 052a b700 01b1 0000 0001 000a .....*.....
000000b0: 0000 0006 0001 0000 0001 0001 000b 000c .....
000000c0: 0001 0009 0000 001c 0002 0002 0000 0004 .....test()
000000d0: 1b1b 60ac 0000 0001 000a 0000 0006 0001 .. \ .....
000000e0: 0000 0003 0001 000d 0000 0002 000e .....

```

3.11 직접 수정해 보기

```

> xxd Test.class
00000000: cafe babe 0000 003b 000f 0a00 0200 0307 .....;.....
00000010: 0004 0c00 0500 0601 0010 6a61 7661 2f6c .....java/l
00000020: 616e 672f 4f62 6a65 6374 0100 063c 696e ang/Object...<in
00000030: 6974 3e01 0003 2829 5607 0008 0100 0454 it>...()V.....T
00000040: 6573 7401 0004 436f 6465 0100 0f4c 696e est...Code...Lin
00000050: 654e 756d 6265 7254 6162 6c65 0100 0474 eNumberTable...t
00000060: 6573 7401 0004 2849 2949 0100 0a53 6f75 est...(I)I...Sou
00000070: 7263 6546 696c 6501 0009 5465 7374 2e6a rceFile...Test.j
00000080: 6176 6100 2100 0700 0200 0000 0000 0200 ava.!.....
00000090: 0100 0500 0600 0100 0900 0000 1d00 0100 .....
000000a0: 0100 0000 052a b700 01b1 0000 0001 000a .....*.....
000000b0: 0000 0006 0001 0000 0001 0001 000b 000c .....
000000c0: 0001 0009 0000 001c 0002 0002 0000 0004 .....
000000d0: 1b1b 60ac 0000 0001 000a 0000 0006 0001 ..`.....
000000e0: 0000 0003 0001 000d 0000 0002 000e .....

```

3.11 직접 수정해 보기

```

> xxd Test.class
00000000: cafe babe 0000 003b 000f 0a00 0200 0307 .....;.....
00000010: 0004 0c00 0500 0601 0010 6a61 7661 2f6c .....java/l
00000020: 616e 672f 4f62 6a65 6374 0100 063c 696e ang/Object...<in
00000030: 6974 3e01 0003 2829 5607 0008 0100 0454 it>...()V.....T
00000040: 6573 7401 0004 436f 6465 0100 0f4c 696e est...Code...Lin
00000050: 654e 756d 6265 7254 6162 6c65 0100 0474 eNumberTable...t
00000060: 6573 7401 0004 2849 2949 0100 0a53 6f75 est...(I)I...Sou
00000070: 7263 6546 696c 6501 0009 5465 7374 2e6a rceFile...Test.j
00000080: 6176 6100 2100 0700 0200 0000 0000 0200 ava!.....
00000090: 0100 0500 0600 0100 0900 0000 1d00 0100 .....
000000a0: 0100 0000 052a b700 01b1 0000 0001 000a .....*.....
000000b0: 0000 0006 0001 0000 0001 0001 000b 000c .....
000000c0: 0001 0009 0000 001c 0002 0002 0000 0004 .....
000000d0: 1b1b 60ac 0000 0001 000a 0000 0006 0001 ..`.....
000000e0: 0000 0003 0001 000d 0000 0002 000e .....

```

3.11 직접 수정해 보기

```
1b iload_1  
1b iload_1  
60 iadd  
ac ireturn
```

3.11 직접 수정해 보기

```
public class Test {  
    public int test(int a) {  
        return a + a;  
    }  
}
```

```
1b iload_1  
1b iload_1  
60 iadd  
ac ireturn
```

3.11 직접 수정해 보기

1b iload_1
1b iload_1
68 imul
ac ireturn

```
public class Test {  
    public int test(int a) {  
        return a * a;  
    }  
}
```

3.11 직접 수정해 보기

```

> xxd Test.class
00000000: cafe babe 0000 003b 000f 0a00 0200 0307 .....;.....
00000010: 0004 0c00 0500 0601 0010 6a61 7661 2f6c .....java/l
00000020: 616e 672f 4f62 6a65 6374 0100 063c 696e ang/Object...<in
00000030: 6974 3e01 0003 2829 5607 0008 0100 0454 it>...()V.....T
00000040: 6573 7401 0004 436f 6465 0100 0f4c 696e est...Code...Lin
00000050: 654e 756d 6265 7254 6162 6c65 0100 0474 eNumberTable...t
00000060: 6573 7401 0004 2849 2949 0100 0a53 6f75 est...(I)I...Sou
00000070: 7263 6546 696c 6501 0009 5465 7374 2e6a rceFile...Test.j
00000080: 6176 6100 2100 0700 0200 0000 0000 0200 ava.!.....
00000090: 0100 0500 0600 0100 0900 0000 1d00 0100 .....
000000a0: 0100 0000 052a b700 01b1 0000 0001 000a .....*.....
000000b0: 0000 0006 0001 0000 0001 0001 000b 000c .....
000000c0: 0001 0009 0000 001c 0002 0002 0000 0004 .....
000000d0: 1b1b 68ac 0000 0001 000a 0000 0006 0001 ..h.....
000000e0: 0000 0003 0001 000d 0000 0002 000a .....

```


3.11 직접 수정해 보기

```
000000d0: 1b1b 60ac  
000000e0: 0000 0003  
> java Main 1  
2
```

```
000000d0: 1b1b 68ac  
000000e0: 0000 0003  
> java Main 1  
1
```

3.12 Java Agent

- 자바에서는 프로그램에 붙어서 본 프로그램이 실행되기 전에 특정 작업을 할 수 있게 하는 자바 에이전트라는 것을 제공
- 이곳에서 instrumentation API를 제공함

```
// static load
public static void premain(String args, Instrumentation inst)

// dynamic load
public static void agentmain(String args, Instrumentation inst)
```

3.13 Instrumentation API

- 해당 API를 통해 transformer를 붙일 수 있다

Module java.instrument
Package java.lang.instrument
Interface Instrumentation

public interface **Instrumentation**

This class provides services needed to instrument Java programming language code. Instrumentation is the addition of byte-codes to methods for the purpose of gathering data to be utilized by tools. Since the changes are purely additive, these tools do not modify application state or behavior. Examples of such benign tools include monitoring agents, profilers, coverage analyzers, and event loggers.

There are two ways to obtain an instance of the Instrumentation interface:

1. When a JVM is launched in a way that indicates an agent class. In that case an Instrumentation instance is passed to the premain method of the agent class.
2. When a JVM provides a mechanism to start agents sometime after the JVM is launched. In that case an Instrumentation instance is passed to the agentmain method of the agent code.

These mechanisms are described in the [package specification](#).

Once an agent acquires an Instrumentation instance, the agent may call methods on the instance at any time.

Since:
1.5

Method Summary

All Methods	Instance Methods	Abstract Methods
Modifier and Type	Method	Description
void	<code>addTransformer(ClassFileTransformer transformer)</code>	Registers the supplied transformer.
void	<code>addTransformer(ClassFileTransformer transformer, boolean canRetransform)</code>	Registers the supplied transformer.
void	<code>addTransformer(ClassLoaderSearcher searcher, boolean canRetransform)</code>	Registers the supplied transformer.

3.13 Instrumentation API

```
fun premain(args: String?, inst: Instrumentation) {
    inst.addTransformer(
        object : ClassFileTransformer {
            override fun transform(
                loader: ClassLoader?,
                className: String?,
                classBeginRedefined: Class<*>?,
                protectionDomain: ProtectionDomain?,
                classFileBuffer: ByteArray?
            ): ByteArray {
                if (className == "Test" && classFileBuffer != null) {
                    classFileBuffer[210] = 0x68
                    return classFileBuffer
                }
                return super.transform(loader, className, classBeginRede
            }
        }
    )
}
```

3.13 Instrumentation API

```
fun premain(args: String?, inst: Instrumentation) {
    inst.addTransformer(
        object : ClassFileTransformer {
            override fun transform(
                loader: ClassLoader?,
                className: String?,
                classBeginRedefined: Class<*>?,
                protectionDomain: ProtectionDomain?,
                classFileBuffer: ByteArray?
            ): ByteArray {
                if (className == "Test" && classFileBuffer != null) {
                    classFileBuffer[210] = 0x68
                    return classFileBuffer
                }
                return super.transform(loader, className, classBeginRede
            }
        }
    )
}
```

3.13 Instrumentation API

Review

```
fun remain(args: String?, inst: Instrumentation) {  
    inst.  
    000000d0: 1b1b 60ac  
    000000e0: 0000 0003  
    > java Main 1  
    2  
  
    000000d0: 1b1b 68ac  
    000000e0: 0000 0003  
    > java Main 1  
    1  
  
    classBeginRedefined: Class<*>?,  
    protectionDomain: ProtectionDomain?,  
    ): ByteArray {  
        if (className == "Test" && classFileBuffer != null) {  
            classFileBuffer[210] = 0x68  
            return classFileBuffer  
        }  
        return super.transform(loader, className, classBeginRede  
    }  
}
```

손으로 직접 수정했던 것을 동적으로 적용할 수 있다!

3.13 Instrumentation API

```
> java Main 1  
2  
> java -javaagent:agent.jar Main 1  
1
```

3.14 Byte Buddy

- Scavenger에서는 Byte Buddy라는 라이브러리를 이용하여 BCI를 진행



The screenshot shows the homepage of the Byte Buddy project. At the top left is the logo "Byte Buddy" in orange. To the right is a navigation menu with "Welcome" (highlighted in orange), "Learn", "Develop", and "API". The main content area features a large orange ghost-like character with white eyes. To the left of the character, the text reads: "Welcome, I'm your Byte Buddy!". Below this, a paragraph describes Byte Buddy as a code generation and manipulation library for creating and modifying Java classes during runtime without a compiler. It mentions that Byte Buddy allows the creation of arbitrary classes and is not limited to implementing interfaces for runtime proxies. It also offers a convenient API for changing classes either manually, using a Java agent or during a build.

<https://bytebuddy.net>

3.15 실제 Scavenger 코드

```
AgentBuilder transform = new AgentBuilder.Default(new ByteBuddy().with(TypeValidation.DISABLED))
    .type(matcherBuilder.buildClassMatcher())
    .transform((builder, typeDescription, classLoader, module) ->
        builder.visit(advise.on(matcherBuilder.buildMethodMatcher(typeDescription)))
    );
```

```
@Advice.OnMethodEnter
public static void onInvocation(@Advice.Origin String signature) {
    String hash = getMethodRegistry().getHash(signature, isLegacyCompatibilityMode());
    getInvocationRegistry().register(hash);
}
```

데이터의 전송, 기록, 열람

3.16 데이터의 전송

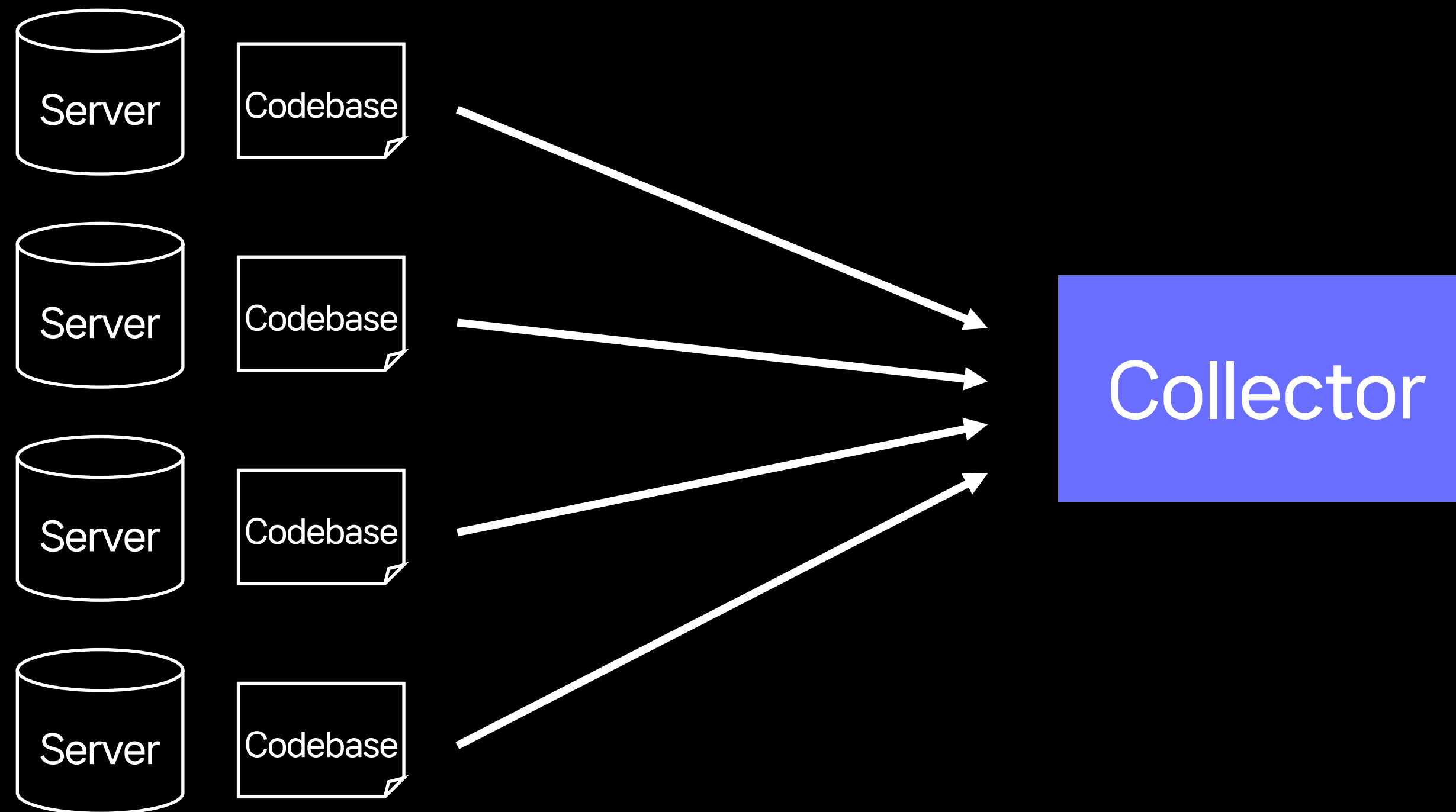
- Agent에서는 처음 실행될 때 codebase 정보를 1회 스캔하여 전송
- 이후에 invocation 정보를 주기적으로 일정 기간의 정보를 모아 전송
- 전송 주기는 동적으로 collector에 질의하여 설정

3.17 데이터의 기록

- Collector에서는 삽입만 수행하고 API에서는 읽기만 수행
- Collector는 모든 쿼리가 upsert 연산으로 이루어져 있음
- 동시성 문제와 성능 문제를 없애기 위해 atomic하게 구성

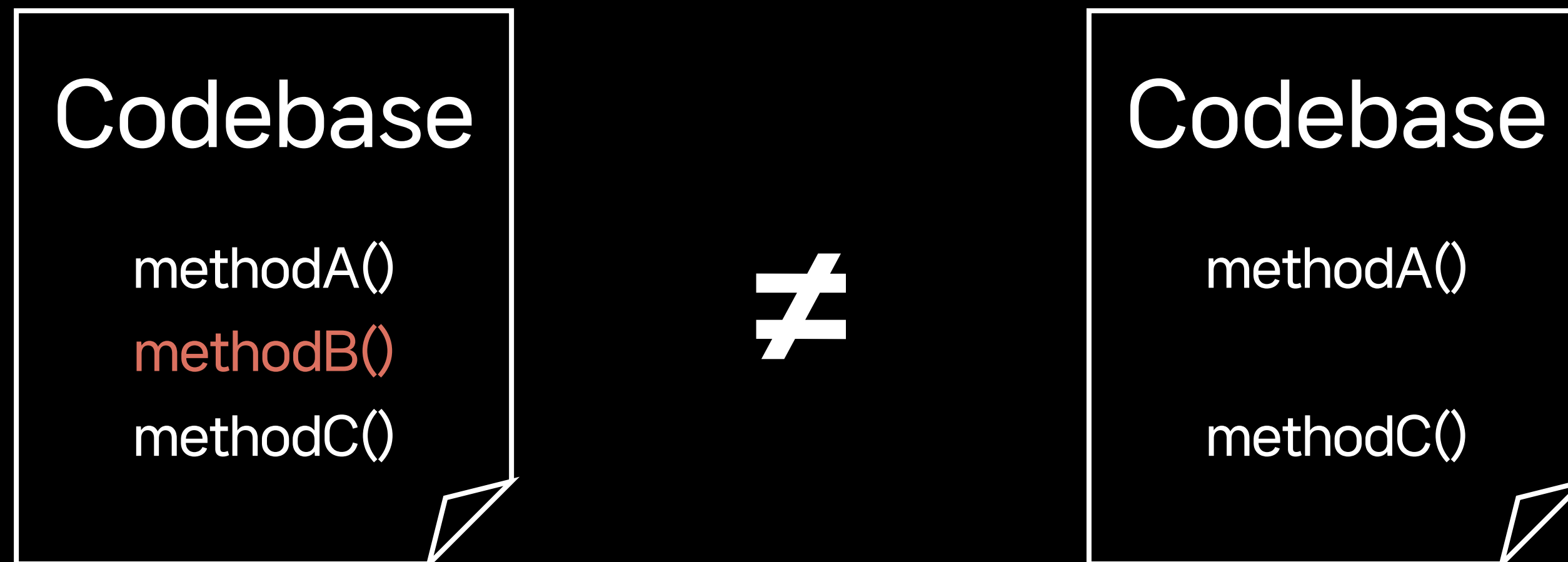
3.18 Codebase Fingerprint

- 동일한 코드가 여러 대의 서버에 배포된 경우



3.18 Codebase Fingerprint

- 기존 코드에서 메서드가 추가되거나 삭제된 경우



3.18 Codebase Fingerprint

- Codebase fingerprint를 부여하여 같은 codebase 여부를 구분함

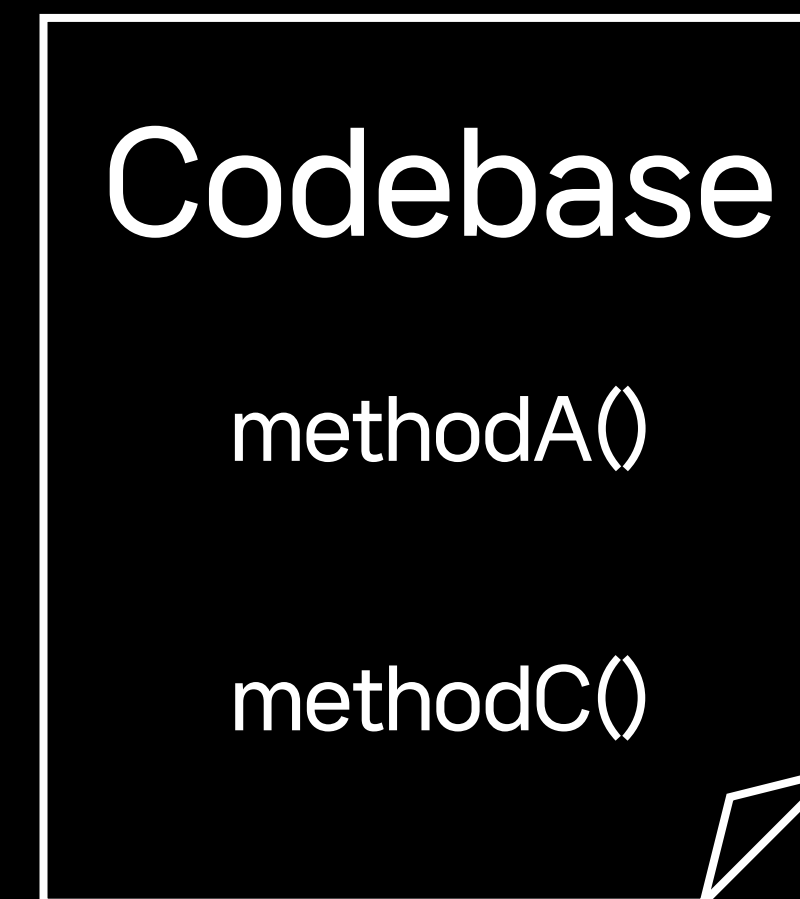
```
private static String calculateFingerprint(Config config, List<Method> methods) {
    MessageDigest md = MessageDigest.getInstance("SHA-256");
    md.update(config.getCodeBase().toString().getBytes(UTF_8));
    md.update(config.getPackages().toString().getBytes(UTF_8));
    md.update(config.getExcludePackages().toString().getBytes(UTF_8));
    md.update(config.getAdditionalPackages().toString().getBytes(UTF_8));
    md.update(config.getAnnotations().toString().getBytes(UTF_8));
    md.update(config.getMethodVisibility().toString().getBytes(UTF_8));
    md.update((byte)(config.isExcludeConstructors() ? 1 : 0));
    md.update((byte)(config.isExcludeGetterSetter() ? 1 : 0));
    md.update((byte)(config.isLegacyCompatibilityMode() ? 1 : 0));

    md.update(longToBytes(methods.size()));
    for (Method each : methods) {
        md.update(each.getSignature().getBytes(UTF_8));
    }

    return String.format("%x", new BigInteger(signum: 1, md.digest()));
}
```

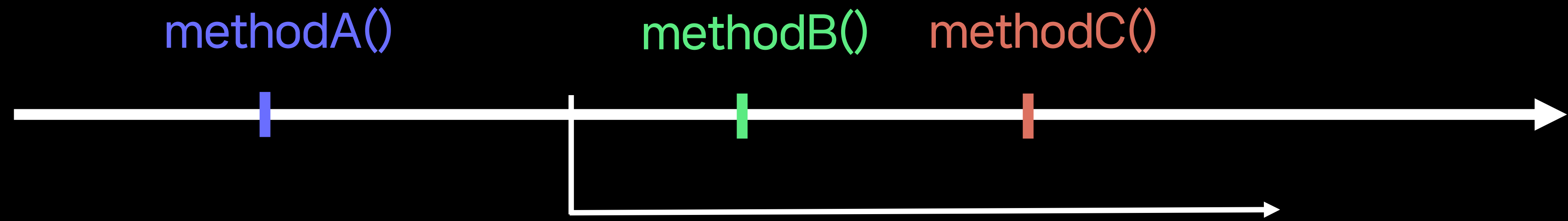
3.19 가비지 콜렉팅

- Codebase에서 삭제된 메서드들은 주기적으로 가비지 콜렉팅 실시
- 수집 중인 codebase의 생성 이전에 마지막으로 발견된 메서드 삭제



3.20 스냅샷

- 특정 시점 이후에 호출된 메서드만 조회하는 경우



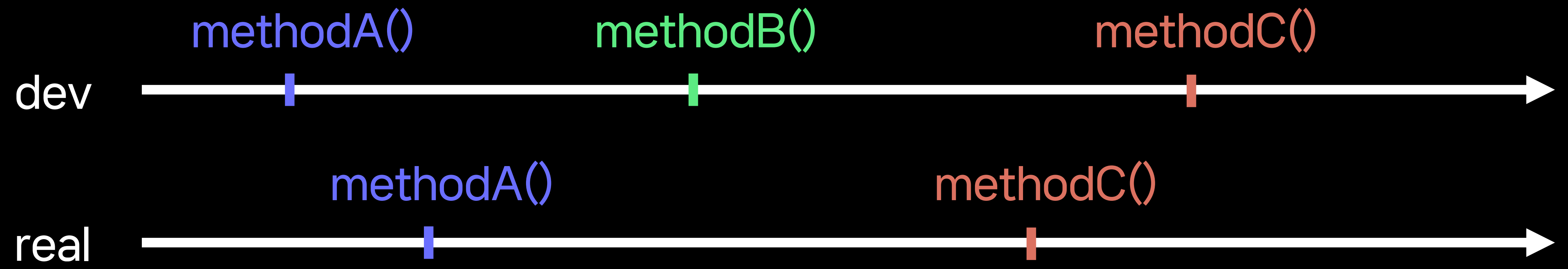
methodA() NOT INVOKED

methodB() INVOKED

methodC() INVOKED

3.20 스냅샷

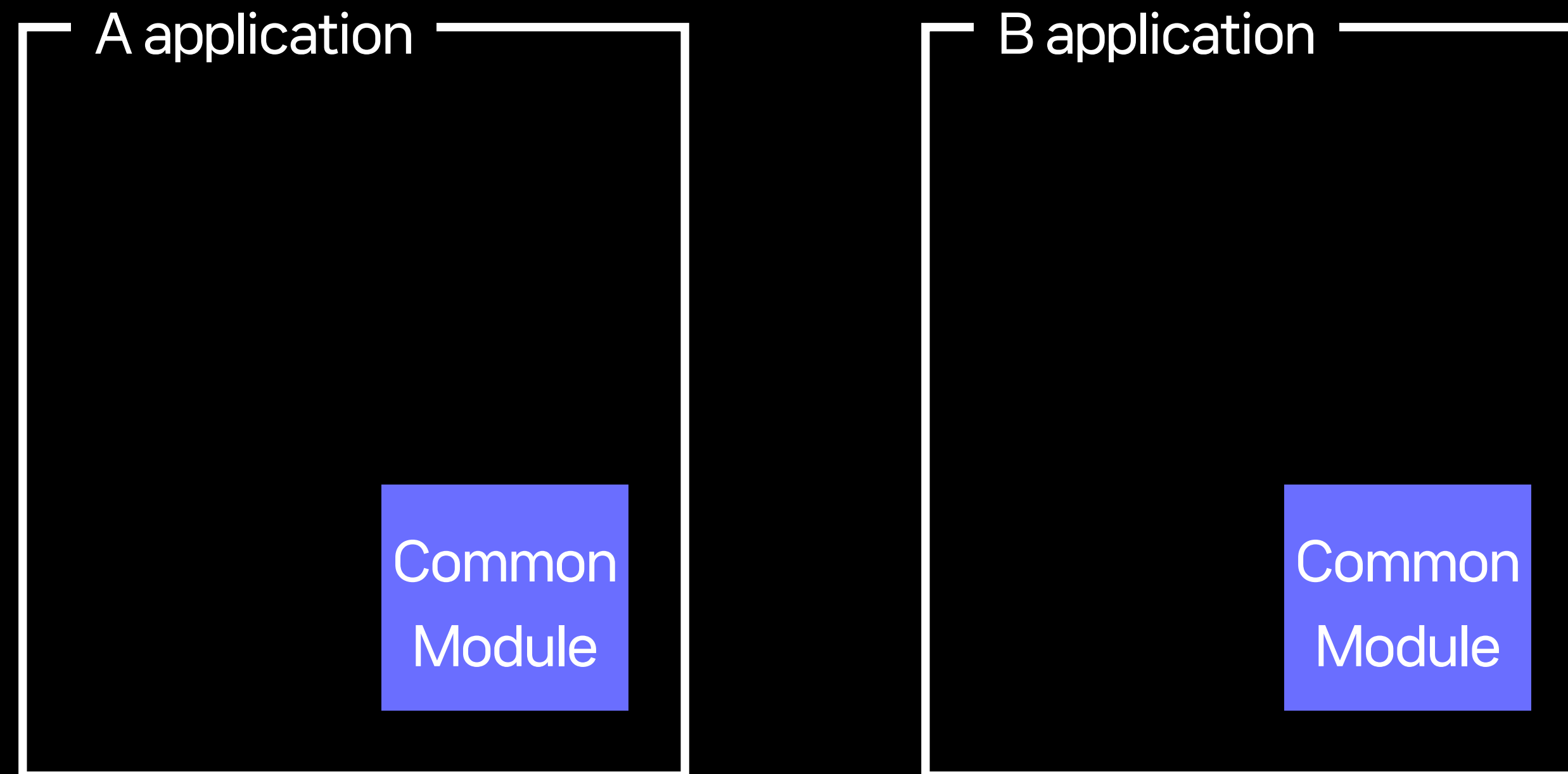
- 서로 다른 환경에 배포된 호출 내역을 통합하여 조회하고 싶은 경우



methodA() INVOKED
methodB() INVOKED
methodC() INVOKED

3.20 스냅샷

- 서로 다른 애플리케이션에 포함된 공통 모듈의 호출 내역을 조회하는 경우



3.20 스냅샷

- 이렇게 모인 호출 데이터들은 스냅샷이라는 형태로 사용자들에게 표현
- 특정 시점 이후에 호출된 메서드의 목록을 제공
- 여러 가지의 애플리케이션이나 환경을 aggregate하여 보는 것이 가능

4. 정리 및 마무리

4.1 Scavenger

- "런타임" Dead Code 분석 도구
- 실제 필요에 의해 개발되어
- 현재 80여개의 크고 작은 네이버 서비스에서 사용 중!

4.1 Scavenger

- 아래와 같은 분들의 노력으로 탄생했고 이제 여러분의 차례입니다



4.1 Scavenger

- 바로 지금 오픈 소스로 공개합니다! 많은 관심 부탁드립니다!!

<https://github.com/naver/scavenger>

- Scavenger에 의해서 애플리케이션이 다운되는 경우가 있을까요?
 - Scavenger의 호출 정보 수집은 애플리케이션과 완전히 별도의 스레드 1개만을 사용하고, 해당 스레드는 애플리케이션과의 완전히 독립적으로 동작하기에 scavenger 의 실행 실패가 애플리케이션에 영향을 미치지 않습니다.
- 어느 JVM 버전부터 지원하나요??
 - Java 1.8 / Java 11 / Java17 까지는 테스트 하였습니다.
Java 1.7 의 경우 구 에이전트(scavenger-old-agent-java)를 사용할 수 있습니다.
- 인터페이스만으로 자동 생성되는 JPA 나 OpenFeign등은 지원하나요?
 - 안타깝게도 지원하지 않습니다. JPA 나 OpenFeign 은 로딩 단계에서 구현체가 장착되는 방식이 아니고, Spring이 런타임에 해당 인터페이스에 Proxy 를 구현하는 방식으로 실행됩니다.
반대로 Scavenger 는 클래스 로딩 단계에서 수집 코드가 삽입됩니다.Scavenger 가 로딩 단계에서 처리할 때는 OpenFeign과 JPA에서 정의하는 인터페이스는 아직 수집 코드를 삽입할 클래스가 없기때문에, 처리가 불가능 합니다.