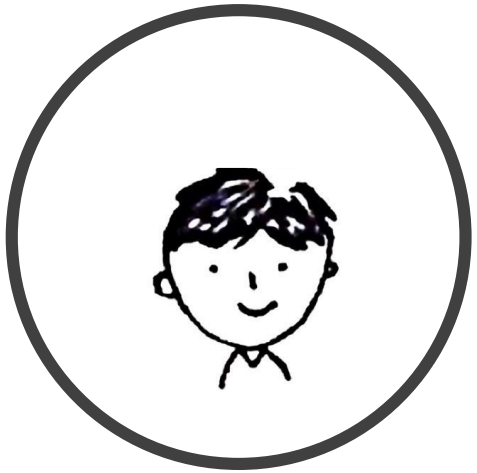
Java tutorial

Javacafe 지대철

소개

- Job : 4학년
- Email : puri8467@gmail.com
- Blog : https://blog.naver.com/puri8467
- 발표 예제 : https://github.com/Danpatpang/javacafe



목차

자바를 선택한 이유

Naver 지식in

자바를 배워야 하는 이유

안녕하세요.
현재 프로그래밍 언어를 공부 중인 대학생입니다.
문득 공부하다가 이런 생각이 들었습니다.
왜 Java 언어를 공부해야 할까요?
Java 언어를 공부했을 때 장단점은 무엇일까요?

Ξ 자바, JSP | #자바 #이유 #Java

비공개 · 4시간 전

1개

채택순 | 전체보기 | 최적 원문

CodingBear 님 답변 친구

초인 · 정보처리기사 · 채택답변수 772 · 자바, JSP23위, 자바스크립트48위, HTML35위

질문자채택

처음 프로그래밍 언어를 배우는데 C 언어도 좋겠지만 JAVA를 습득하시면..
타 프로그래밍 언어의 이해하는데 도움이 큽니다.
언어 하나만 하실 거는 아니죠 ㅎㅎ

4시간 전

Stackoverflow

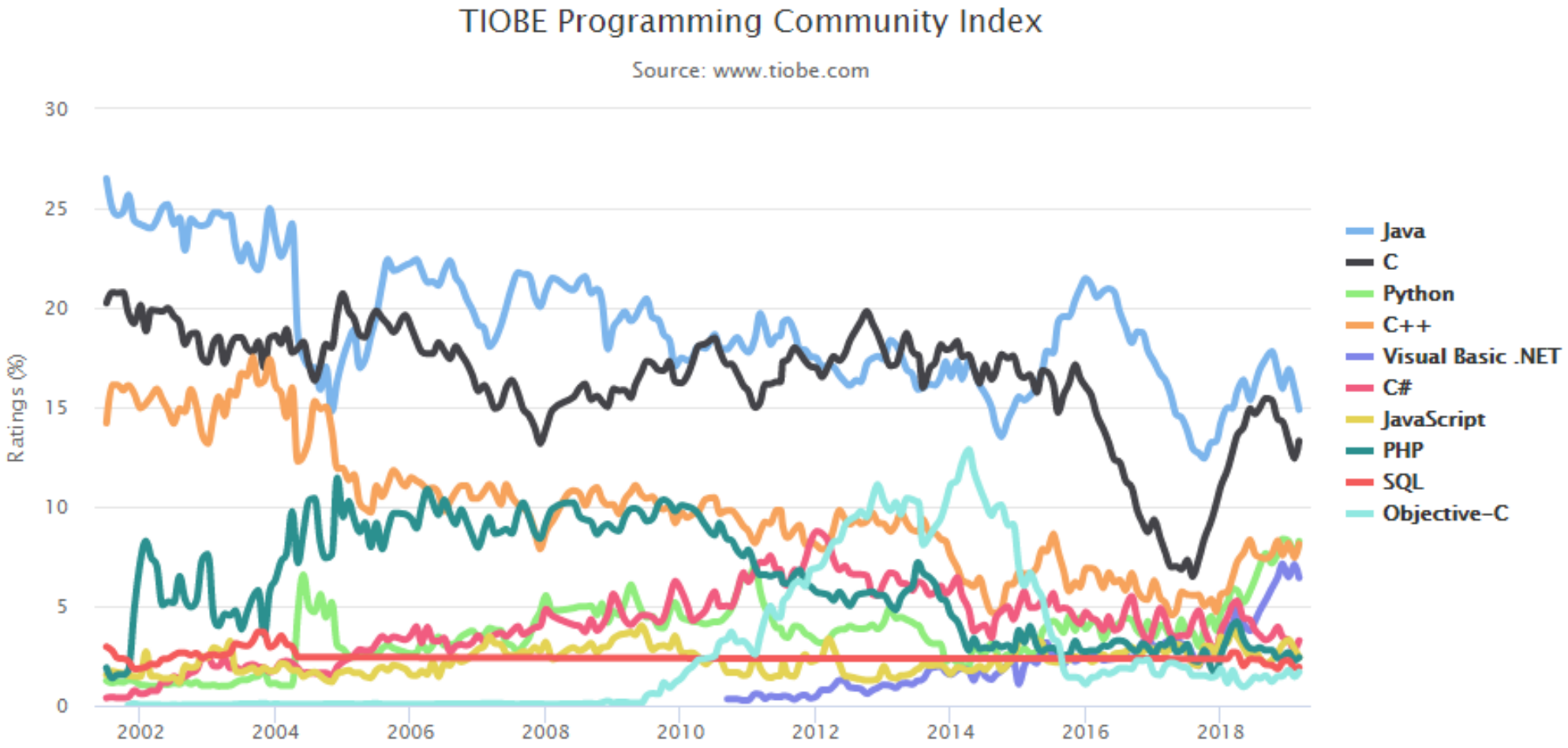
자바를 배워야 하는 이유

https://www.youtube.com/watch?v=lh934b4ds-4

1. 자바를 배우기 위한 수많은 자원
2. 많은 취업의 기회
3. 객체 지향 프로그래밍 언어
4. 자바는 오픈 소스
5. 풍부한 API
6. 강력한 개발 툴
7. 전세계적으로 사용
8. 커뮤니티의 훌륭한 지원
9. 실세계의 다양한 애플리케이션에서 사용
10. 배우기 쉬움 (C에 비해)

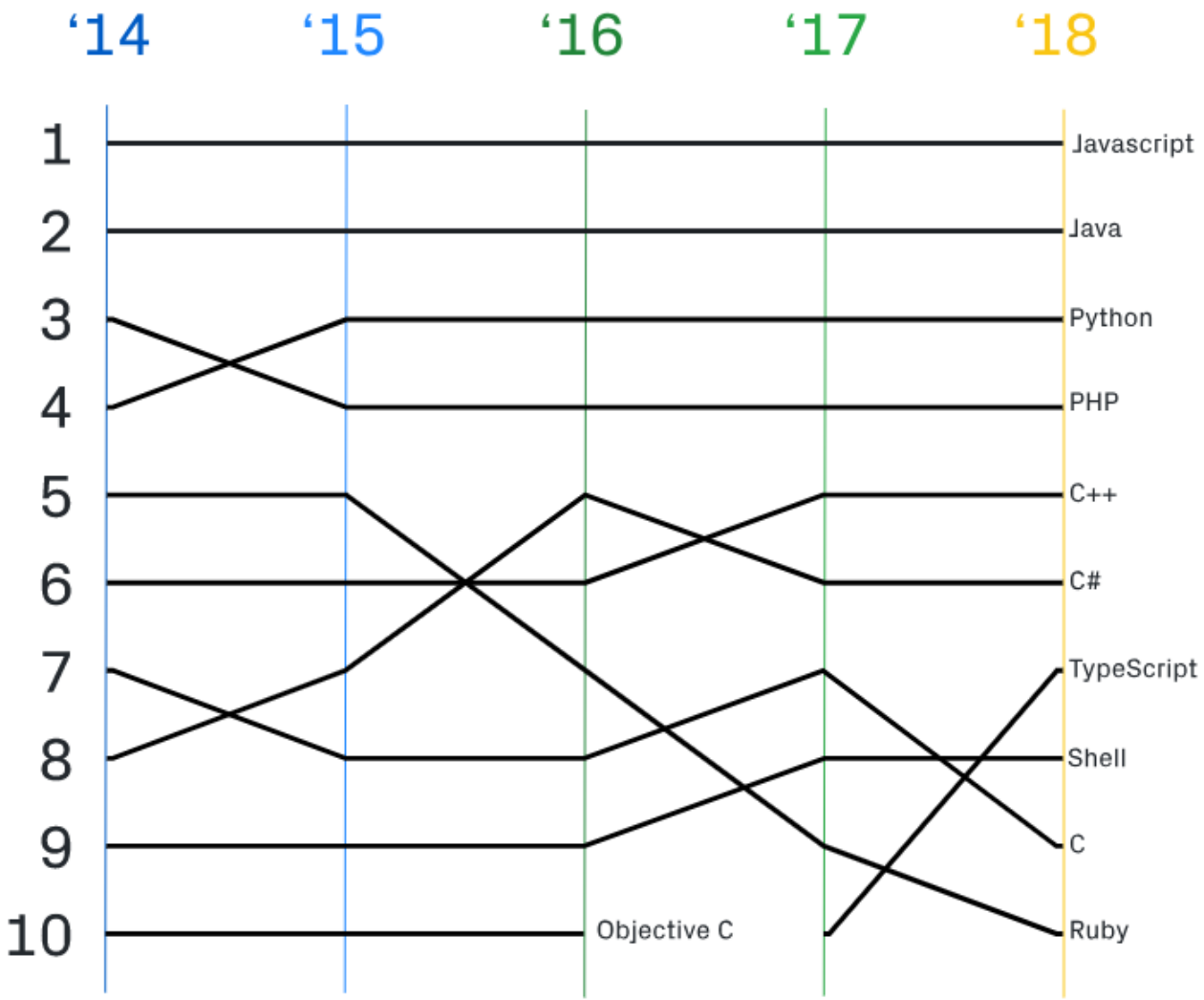
TIOBE 프로그래밍 언어 사용 순위

https://www.tiobe.com/tiobe-index/

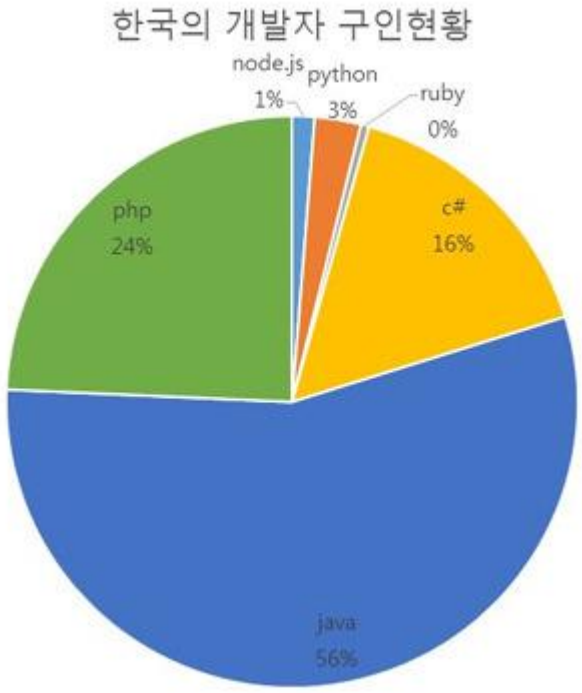


Github 2018 octoverse

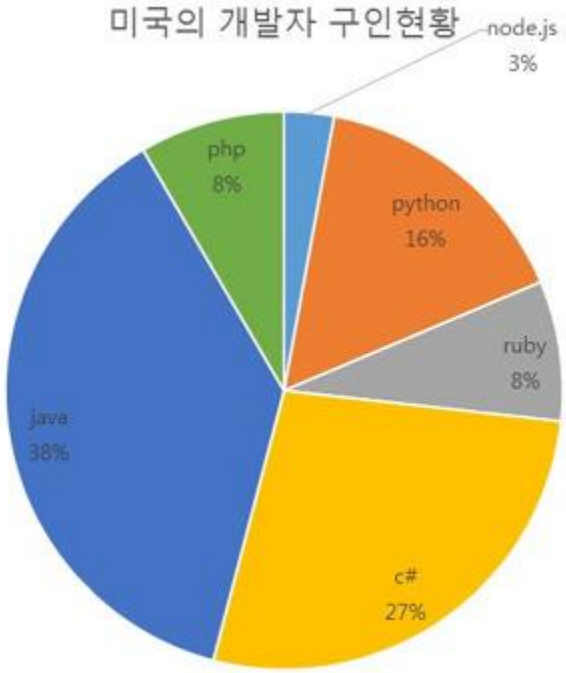
https://octoverse.github.com/projects#languages



개발자 구인 현황



잡코리아 키워드 검색 결과



Computerjobs에서 조회

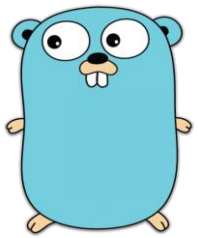
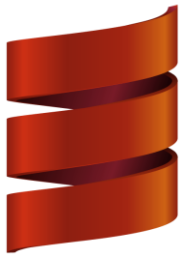
개발자 구인 현황

언어	잡코리아	사람in
Java	5442건	2273건
C, C++, C#	4623건	1865건
Python	733건	646건
JavaScript	1708건	696건
PHP	1179건	807건
SQL	1947건	833건

〈2019년 4월 2주차 기준〉

자바가 아닌 다른 언어를 배워야 하는 이유

1. 자바가 주류 언어가 되기까지 - 오픈 소스 진영과의 불편한 동거
2. 오라클 시대의 자바, 자바스크립트의 반격
3. JCP를 통한 관리, 다른 언어에 비해 느린 발전 속도



자바의 역사와 변화에 대해 알아보자.

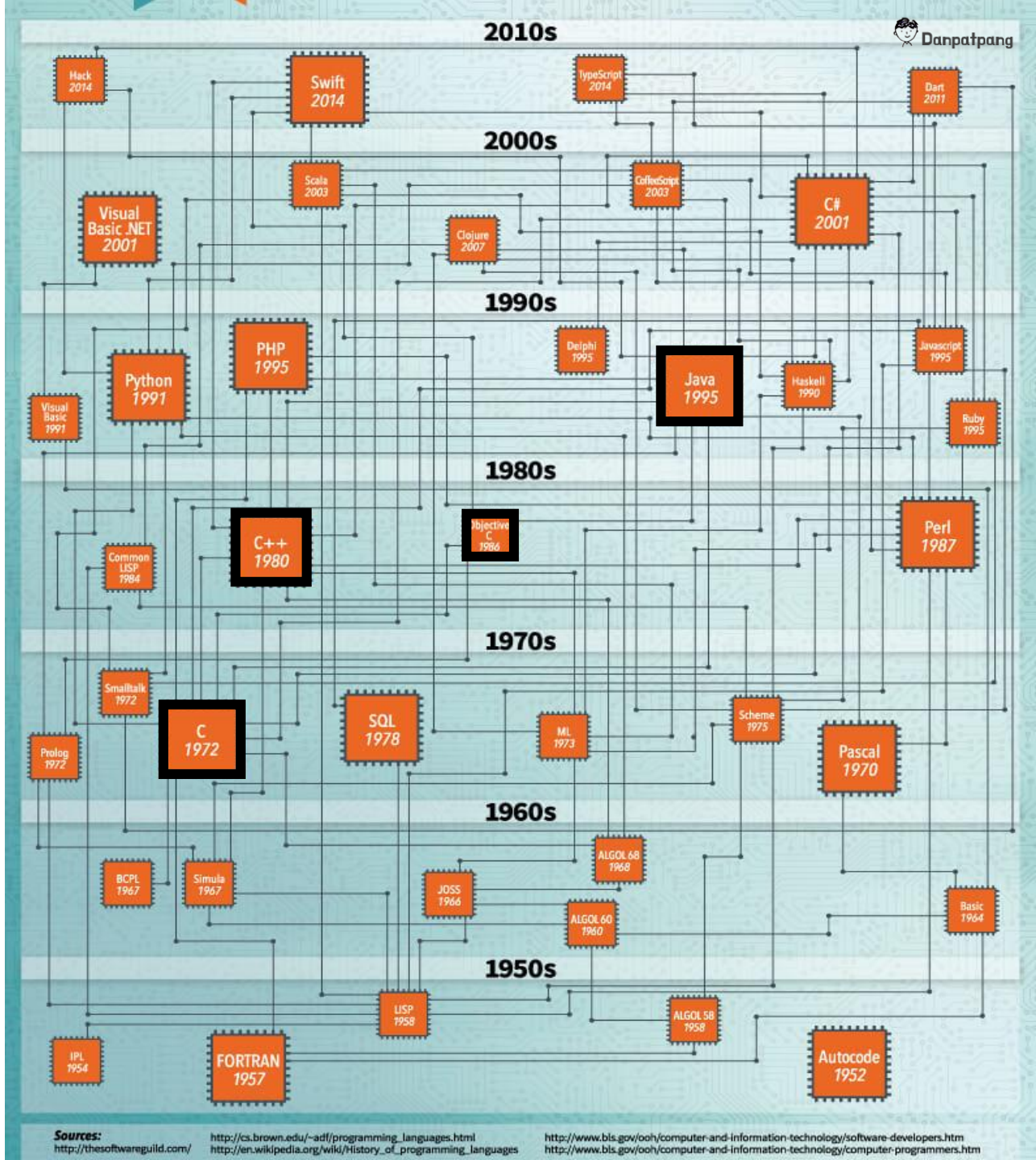
자바의 역사와 변화

1991년
Sun Microsystems사의
제임스 고슬링 외 4인이 개발 "Oak".

1995년
"Java"로 명칭 변경.

2009년
Sun Microsystems가 오라클과 인수 합병.

2017년
Java EE 포기.
이클립스 재단에서 JakartaEE로 변경.



C, C++와의 차이점

기존의 C에 객체지향 기능을 추가하여 C++을 만들어보자.

→ 포인터도 가져오고~ 객체지향도 추가하고~

→ 이거 어떻게 만들어? 너무 복잡한데...

→ 좀 어색해... 처음부터 객체 지향으로 만들자.

→ 포인터 포기하자... 복잡해지면 안 돼!

자바의 철학

플랫폼 독립적인 언어를 만들자!

Write Once Run Anywhere!

폭발적 성장

Programming Language	2019	2014	2009	2004	1999	1994	1989
Java	1	2	1	1	10	-	-
C	2	1	2	2	1	1	1
C++	3	4	3	3	2	2	3
Python	4	7	5	10	29	21	-
Visual Basic .NET	5	11	-	-	-	-	-
C#	6	5	7	7	27	-	-
JavaScript	7	8	8	8	18	-	-
PHP	8	6	4	5	-	-	-
SQL	9	-	-	6	-	-	-
Objective-C	10	3	36	45	-	-	-
COBOL	25	19	15	11	3	9	13
Lisp	27	13	18	14	13	5	2
Pascal	204	14	14	96	6	3	7

Tiobe 과거 프로그래밍 언어 순위

인터넷과 자바의 만남

자바의 세부 부분 개발되고 있을 무렵. WWW의 출현.

(1991년 뉴스를 통해 대중화. 1993년 서버 수 50대 → 500대)

이식 가능한 언어를 찾습니다.

(인터넷은 다양한 컴퓨터. OS. CPU로 넘쳐나는 광대한 분산 시스템)

가전 제품에서 인터넷으로 초점 변경.

Java 주요 릴리즈 히스토리

Version	Date	Issues
1.0	1996년 1월	Oak로 출시. 1.0.2 버전부터 Java로 불리기 시작
1.1	1997년 2월	AWT, Inner class, JDBC, RMI, JIT 컴파일러, 유니코드 통합
1.2	1998년 12월	애플릿, Swing GUI, Collections 추가, J2SE 1.2로 표기
1.3	2000년 3월	HotSpot JVM, JNDI, JavaSound 추가
1.4	2002년 2월	정규 표현식, NIO, XML 파서 통합, IPv6 지원, Java Web Start
1.5	2004년 9월	Generics, Autoboxing/Unboxing, Enumerations, static imports
1.6	2006년 12월	Security, JavaSE 1.6으로 표기
1.7	2011년 7월	Multi Exception catch, Type Inference, Null safe Method invocation, String in Switch, Automatic Resource Management
1.8	2014년 3월	Lambda Expression, Streams, Method Reference
1.9	2017년 9월	Jshell, HTTP/2, private 인터페이스 메서드

업데이트가 늦어진 이유


ORACLE
Google


Java 1.0 1996 1.2 1998 1.3 2000 1.4 2002 5.0 2004 6.0 2006 7.0 2011 8.0 2013
Scala 2001 1.0 2003 2.0 2006 2.10 2013
2003 1.0 2007 2.0 2012
FANTOM 2005 THOMPSON 2008 1.0.64 2012
Clojure 2007 1.0 2009 1.4 2012
ceylon 2010 M1 2011 M1 2012
PROJECT Kotlin 2011 M1 2012 M5 2013
Xtend 1.0 2011 2.3 2012


MySQL®
MariaDB

Java 에디션 분류

Java SE

자바 표준 에디션.

Jakarta EE (구 JavaEE)

기업에서 운영하는 서버 페이지에 특화된 에디션.

Java ME

임베디드 시스템 환경에 특화된 에디션.

Java FX

데스크톱 앱 개발 및 배포를 위한 에디션. (GUI 라이브러리 제공)

Jakarta EE(Java EE)

Share

COMMUNITY, JAVAEE | Thursday, August 17, 2017

Opening Up Java EE

By: David Delabassee | Software Evangelist

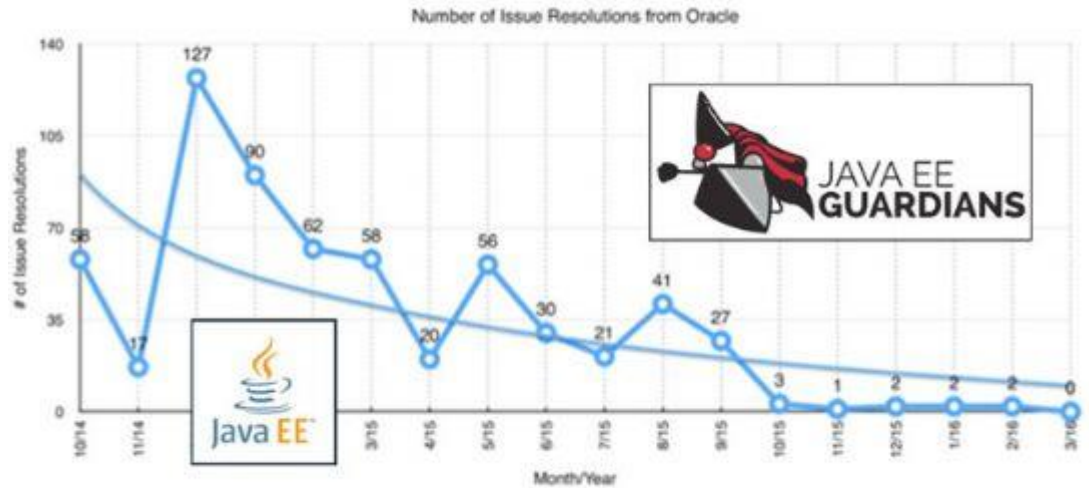
We continue to make great progress on Java EE 8. Specifications are nearly complete, and we expect to deliver the reference implementation this summer. As we approach the delivery of Java EE 8 and the JavaOne 2017 conference, we believe there is an opportunity to rethink how Java EE is developed in order to make it more agile and responsive to changing industry and technology demands.

Java EE is enormously successful, with a competitive market of compatible implementations, broad adoption of individual technologies, a huge ecosystem of frameworks and tools, and countless applications delivering value to enterprises and end users. But although Java EE is developed in open source with the participation of the Java EE community, often the process is not seen as being agile, flexible or open enough, particularly when compared to other open source communities. We'd like to do better.

We are discussing how we can improve the Java EE development process following the delivery of Java EE 8. We believe that moving Java EE technologies including reference implementations and test compatibility kit to an open source foundation may be the right next step, in order to adopt more agile processes, implement more flexible licensing, and change the governance process. We plan on exploring this possibility with the community, our licensees and several candidate foundations to see if we can move Java EE forward in this direction.

We intend to meet ongoing commitments to developers, end users, customers, technology consumers, technology contributors, partners and licensees. And we

오라클 소프트웨어에서 2017년 8월 포스팅



2014년 10월 ~ 2016년 3월 오라클이 해결한 EE 이슈

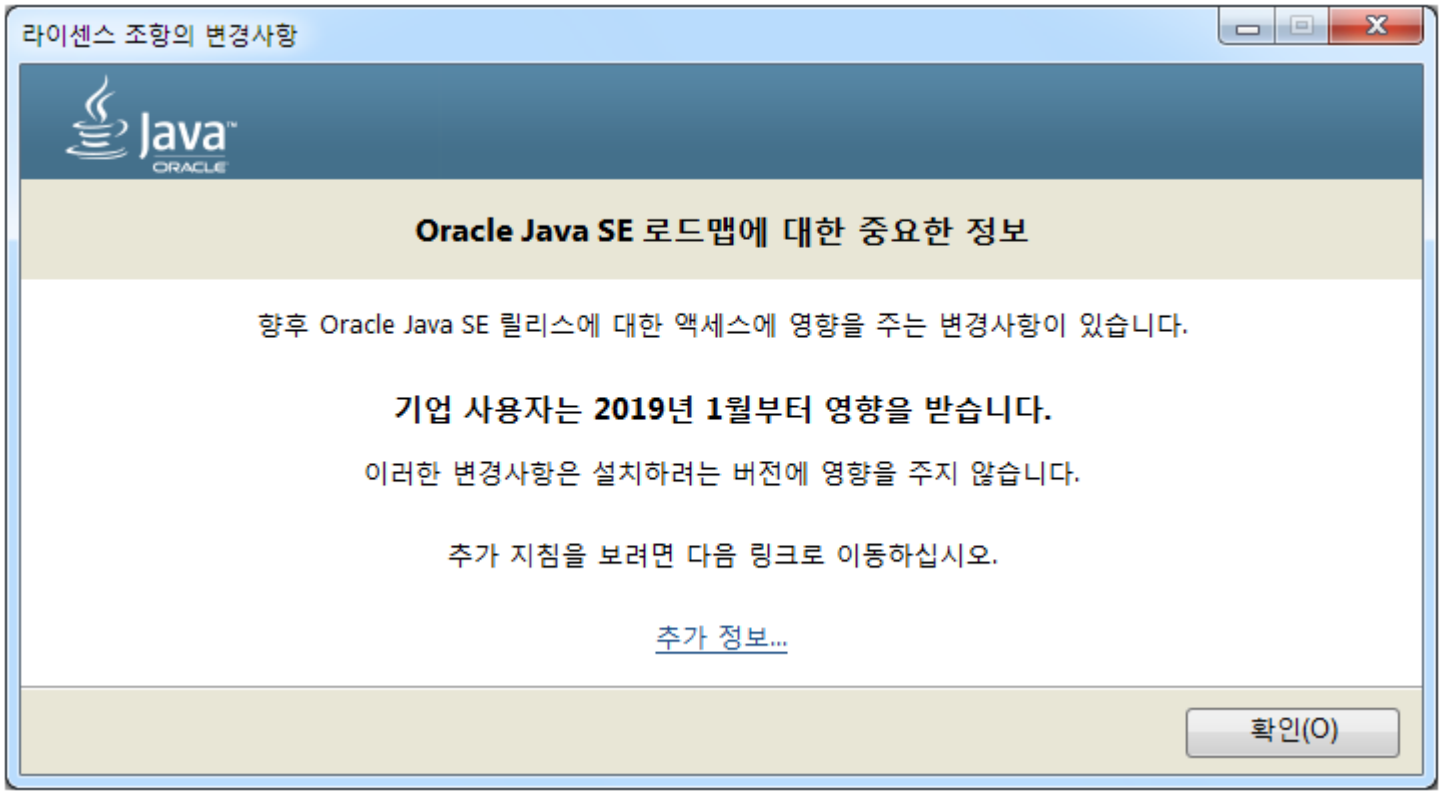
자바의 미래

자바 유료화?

자바는 무료지만, JDK는 유료였다.

자바 언어는 오픈소스로 누구나 무료로 사용할 수 있다.

오라클은 JDK 과금 방식을 변경했다. (영구 라이선스 → 구독 모델)



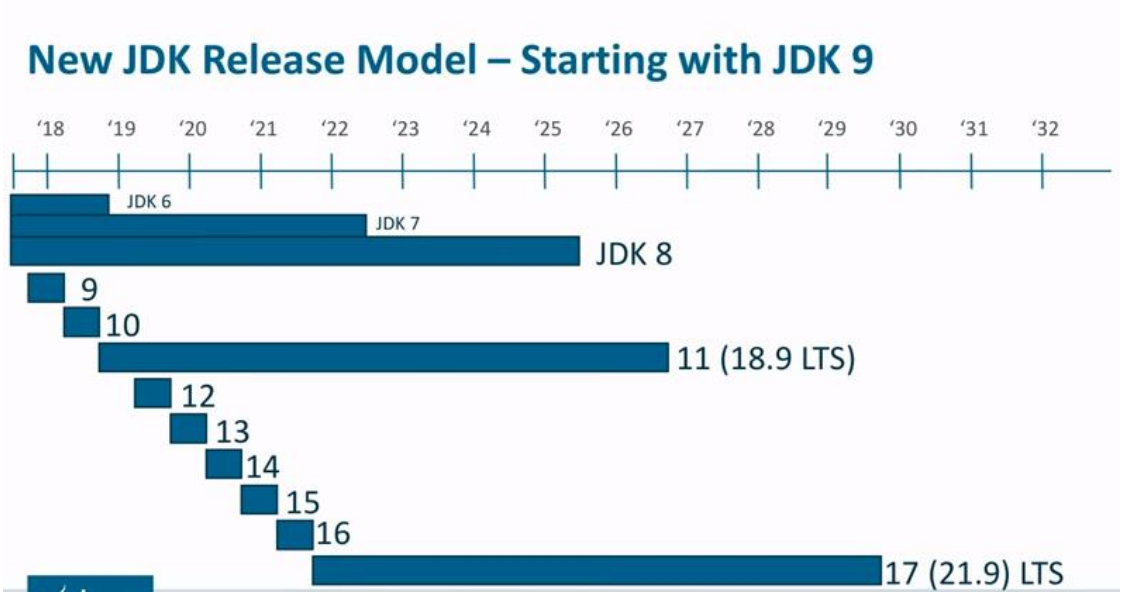
구독 요금

Oracle Java SE Support Roadmap*†				
Release	GA Date	Premier Support Until**	Extended Support Until**	Sustaining Support**
6	December 2006	December 2015	December 2018	Indefinite
7	July 2011	July 2019	July 2022	Indefinite
8	March 2014	March 2022	March 2025	Indefinite
9 (non-LTS)	September 2017	March 2018	Not Available	Indefinite
10 (18.3^) (non-LTS)	March 2018	September 2018	Not Available	Indefinite
11 (18.9^ LTS)	September 2018	September 2023	September 2026	Indefinite
12 (19.3^ non-LTS)	March 2019***	September 2019	Not Available	Indefinite

Java SE Subscription Pricing		
Volume	Subscription Metric	Monthly Subscription Price
1-99	Processor	$25.00
100-249	Processor	$23.75
250-499	Processor	$22.50
500-999	Processor	$20.00
1,000-2,999	Processor	$17.50
3,000-9,999	Processor	$15.00
10,000-19,999	Processor	$12.50
20,000+	Contact for details	

Java SE Desktop Subscription Pricing		
Volume	Subscription Metric	Monthly Subscription Price
1-999	Named User Plus	$2.50
1,000-2,999	Named User Plus	$2.00
3,000-9,999	Named User Plus	$1.75
10,000-19,999	Named User Plus	$1.50
20,000-49,999	Named User Plus	$1.25
50,000+	Contact for details	

릴리즈의 주기별 업데이트



Overview | Downloads | Documentation | Community | Technologies | Training

Java SE Development Kit 12 Downloads

Thank you for downloading this release of the Java™ Platform, Standard Edition Development Kit (JDK™). The JDK is a development environment for building applications, and components using the Java programming language.

The JDK includes tools useful for developing and testing programs written in the Java programming language and running on the Java platform.

Important changes in Oracle JDK 12 License

Starting with JDK 11 Oracle has updated the license terms on which we offer the Oracle JDK.
The new Oracle Technology Network License Agreement for Oracle Java SE is substantially different from the licenses under which previous versions of the JDK were offered. Please review the new terms carefully before downloading and using this product.

Oracle also offers this software under the GPL License on jdk.java.net/12

See also:

- Java Developer Newsletter: From your Oracle account, select **Subscriptions**, expand **Technology**, and subscribe to **Java**.
- Java Developer Day hands-on workshops (free) and other events
- Java Magazine

JDK 12 checksum

Java SE Development Kit 12

You must accept the Oracle Technology Network License Agreement for Oracle Java SE to download this software.

○ Accept License Agreement ◉ Decline License Agreement

Product / File Description	File Size	Download
Linux	154.11 MB	jdk-12_linux-x64_bin.deb
Linux	162.53 MB	jdk-12_linux-x64_bin.rpm
Linux	181.21 MB	jdk-12_linux-x64_bin.tar.gz
macOS	173.38 MB	jdk-12_osx-x64_bin.dmg
macOS	173.69 MB	jdk-12_osx-x64_bin.tar.gz
Windows	158.49 MB	jdk-12_windows-x64_bin.exe
Windows	179.44 MB	jdk-12_windows-x64_bin.zip

유료화에 대한 대응

1. 기존에 라이선스를 사서 구매하던 기업은 큰 변화 X.

2. 돈 내기가 부담스러운 기업은 OpenJDK를 사용.

3. 스타트업은 언어 변경.

OpenJDK란?

Java SE의 자유-오픈 소스 구현체.

GPLv2 라이선스 + 추가 라이선스를 따르고 있다.

종류 : Oracle, Zulu, Red Hat, Red Hat, IBM, IntelliJ 등

OpenJDK의 문제점

실제 적용 사례가 많지 않다.

호환성에 문제가 없는가?

오픈 소스이기에 참여자가 없으면 자연스럽게 죽는다.

문제가 생겼을 때 누가 처리하고 책임을 질 것인가?

안정성과 성능 차이는 없는가?

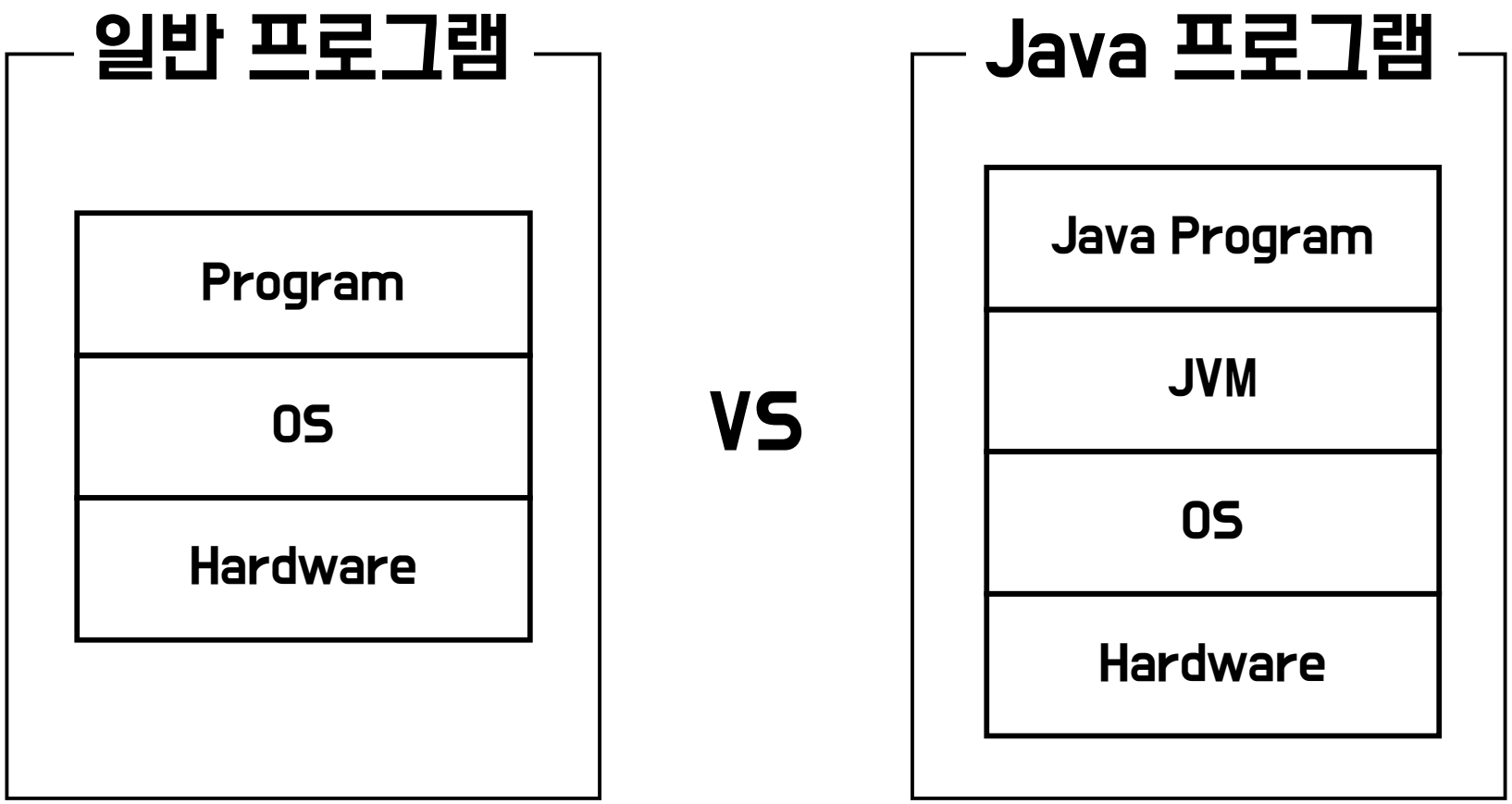
자바의 미래

어떻게 흘러갈지는 아무도 모른다...

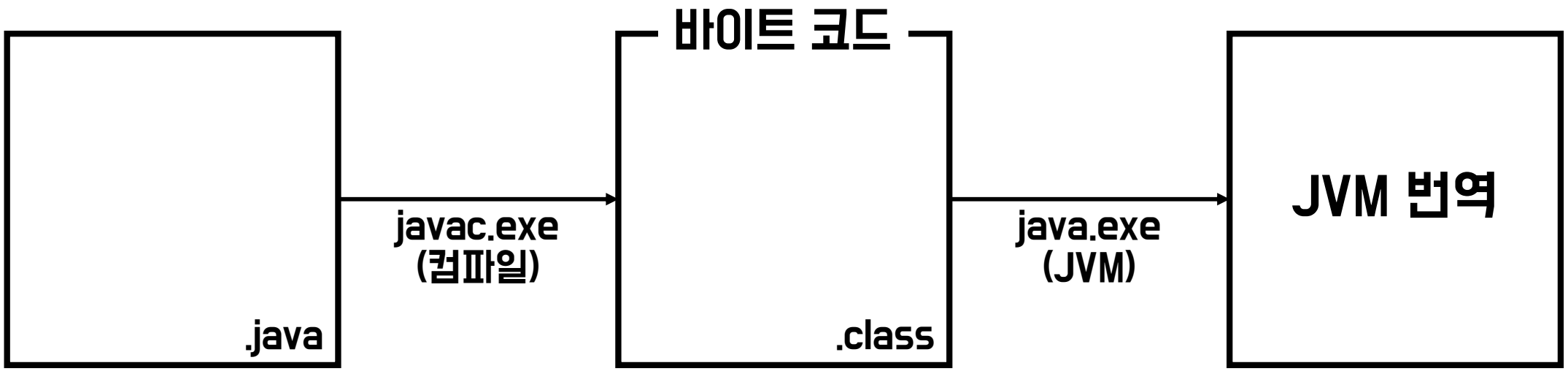
자바의 특징

1. 이식성이 높은 언어
2. 객체 지향 언어
3. 함수적 스타일 코딩 지원
4. 메모리 자동 관리
5. 다양한 애플리케이션에서 개발 가능
6. 멀티 스레드를 쉽게 구현
7. 동적 로딩 지원
8. 막강한 오픈 소스 라이브러리

자바의 동작 과정


일반 프로그램
Program
OS
Hardware
VS
Java 프로그램
Java Program
JVM
OS
Hardware

자바의 동작 과정



직접 해보자.

1. 자바 다운로드 및 설치

   https://www.oracle.com/technetwork/java/javase/downloads/

2. 환경 변수 설정

   https://macchiato.tistory.com/9

3. 제대로 설치됐는지 확인

   java -version

   javac –version

4. 명령 프롬프트에서 java 명령어 실행

JRE, JDK, JVM

JRE(Java Runtime Environment)

물리적으로 존재하는 JVM을 구현하는 역할.

동작을 위한 라이브러리 + JVM

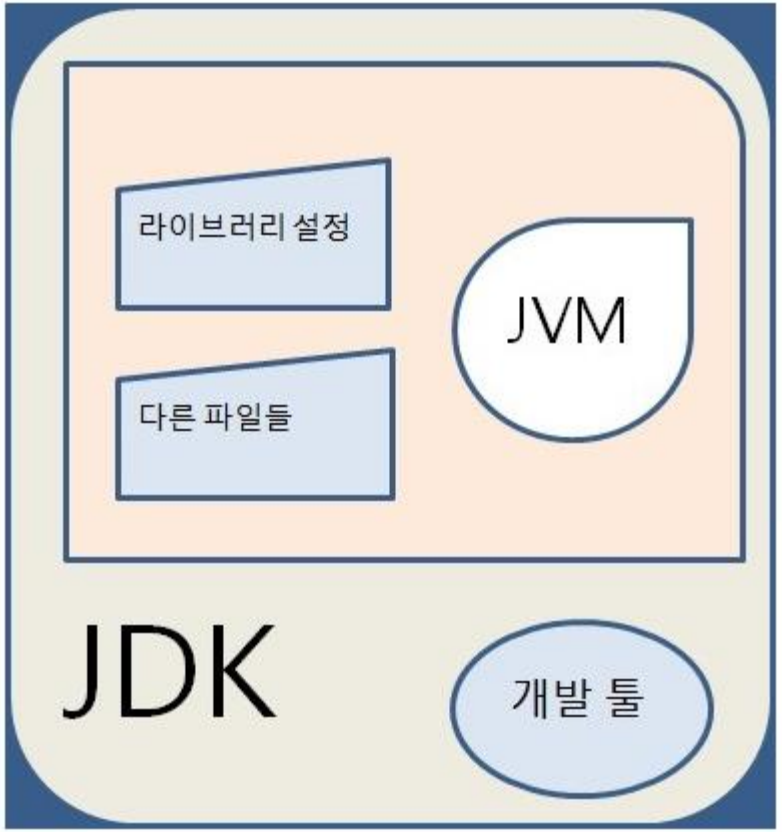
JDK(Java Development Kit)

JRE + 개발 툴

JVM(Java Virtual Machine)

추상적 머신.

바이트 코드가 실행될 수 있는 런타임 환경 제공.



코드 소개

클래스(class)

메서드(method)

명령문(statement)

```java
/**
 * This class is Test class.
 * @author danpatpang
 */
public class CommandTest {

    public static void main(String[] args) {
        System.out.println("Hello world");
        printApple();
    }

    /**
     * This function print "Apple".
     */
    public static void printApple() {
        System.out.println("Apple");
    }
}
```

클래스, 객체, 인스턴스

https://alfredjava.wordpress.com/2008/07/08/class-vs-object-vs-instance/

Class vs Object vs Instance

with 9 comments

In OO Programming, we often hear of terms like "Class", "Object" and "Instance"; but what actually is a Class / Object / Instance?

In short, An **object** is a software bundle of related state and behavior. A **class** is a blueprint or prototype from which objects are created. An **instance** is a single and unique unit of a class.

Example, we have a blueprint (class) represents student (object) with fields like name, age, course (class member). And we have 2 students here, Foo and Bob. So, Foo and Bob is 2 different instances of the class (Student class) that represent object (Student people).

Let me go into details...

Object
Real world objects shares 2 main characteristics, *state* and *behavior*. Human have state (name, age) and behavior (running, sleeping). Car have state (current speed, current gear) and state (applying brake, changing gear). Software objects are conceptually similar to real-world objects: they too consist of state and related behavior. An object stores its state in *fields* and exposes its behavior through *methods*.

Class
Class is a "template" / "*blueprint*" that is used to create objects. Basically, a class will consists of field, static field, method, static method and constructor. Field is used to hold the state of the class (eg: name of Student object). Method is used to represent the behavior of the class (eg: how a Student object going to stand-up). Constructor is used to create a new Instance of the Class.

Instance
An instance is a unique copy of a Class that representing an Object. When a new instance of a class is created, the JVM will allocate a room of memory for that class instance.

클래스, 객체, 인스턴스

Object(객체)

실 세계의 객체는 상태와 행동이라는 2가지의 주요 특징을 가지고 있다.
사람은 이름, 나이와 같은 상태를 가지며 걷기, 자기와 같은 행동을 가진다.
자동차의 경우 현재 속도, 엔진과 같은 상태를 가지며, 브레이크 동작, 기어 변화와 같은 행동이 있다.
소프트웨어의 객체는 실세계의 객체와 개념적으로 유사하다.
둘 다 상태, 상태와 연관된 행동으로 구성되어 있다.
소프트웨어의 객체는 필드에 상태를 저장하고 메서드를 통해 행동을 노출시킨다.

Class(클래스)

클래스는 객체를 만드는 데 사용되는 "템플릿" 또는 "청사진"이다.
기본적으로 클래스는 필드, 정적 필드, 메서드, 정적 메서드, 생성자로 구성이 된다.
필드는 클래스의 상태(학생의 이름)를 유지하는데 사용되며, 메서드는 클래스의 행동(이름 쓰기)을 나타내는데 사용된다.
생성자는 클래스의 새로운 인스턴스를 만드는 데 사용된다.

Instance(인스턴스)

인스턴스는 객체를 나타내는 클래스의 고유한 복사본이다.
클래스의 새 인스턴스가 생성되면, JVM은 해당 클래스의 인스턴스에 대해 메모리 공간을 할당해준다.

직접 해보자.

1. 자바 파일 컴파일

   javac CommandTest.java

2. 바이트 코드 실행

   java CommandTest

   java -classpath ./ test.CommandTest

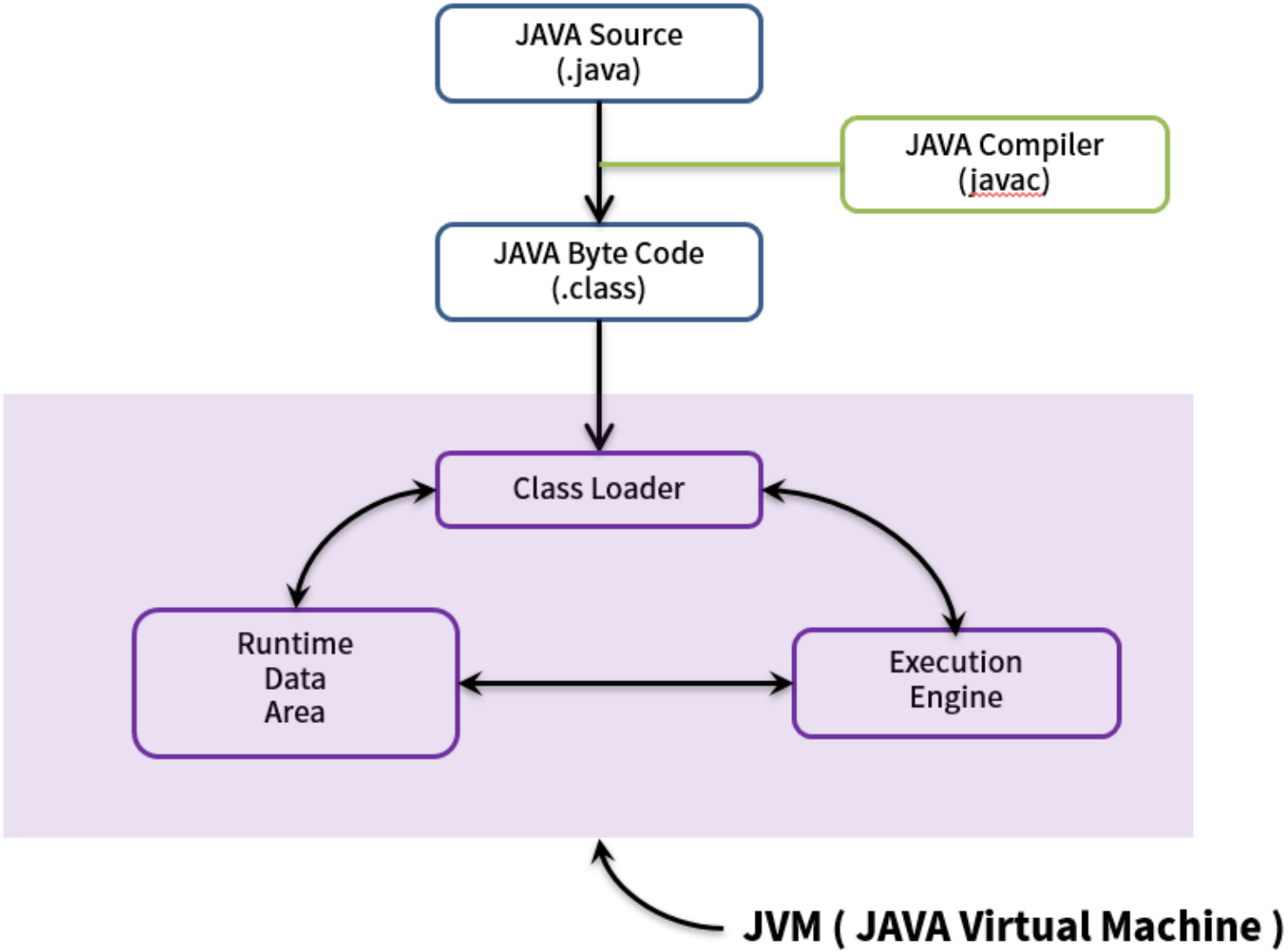
3. 바이트 코드 디컴파일

   javap CommandTest.class

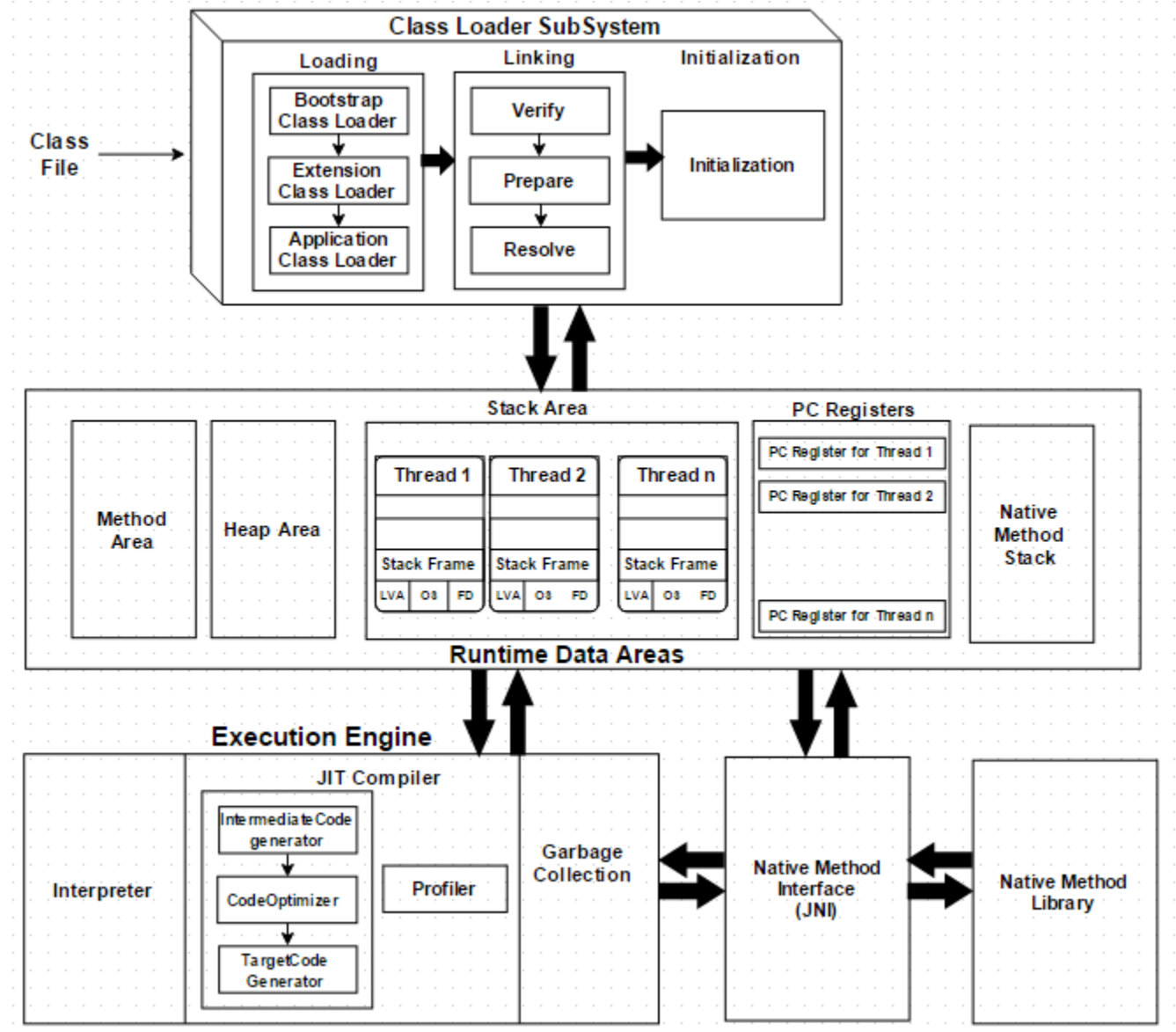
4. Javadoc 생성

   javadoc CommandTest.java

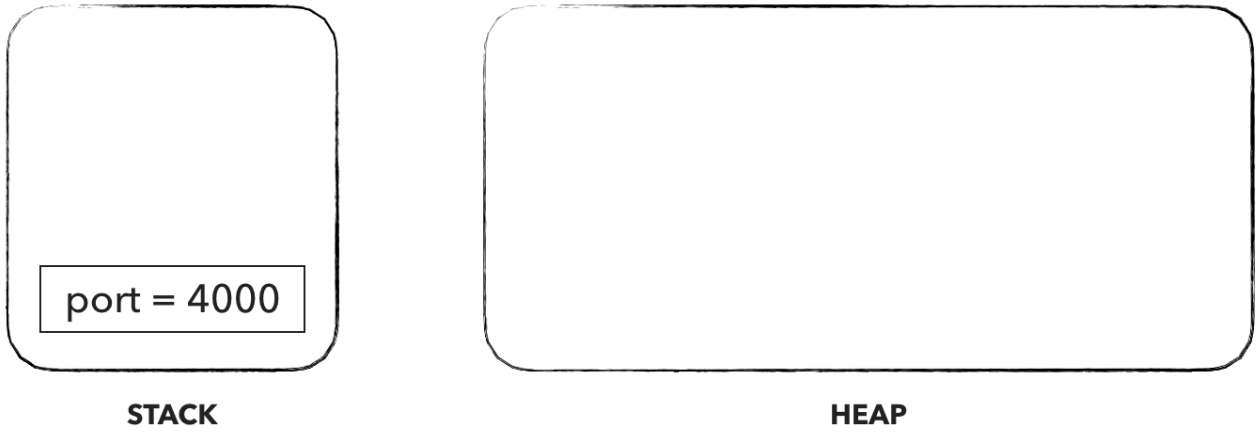
자바의 동작 과정


JAVA Source
(.java)
JAVA Compiler
(javac)
JAVA Byte Code
(.class)
Class Loader
Runtime
Data
Area
Execution
Engine
JVM (JAVA Virtual Machine)

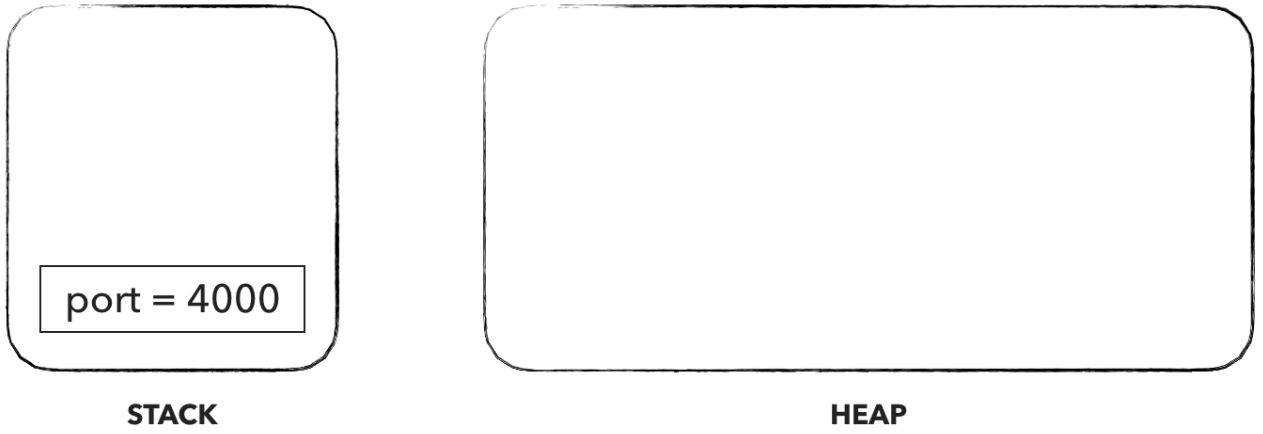
자바의 동작 과정

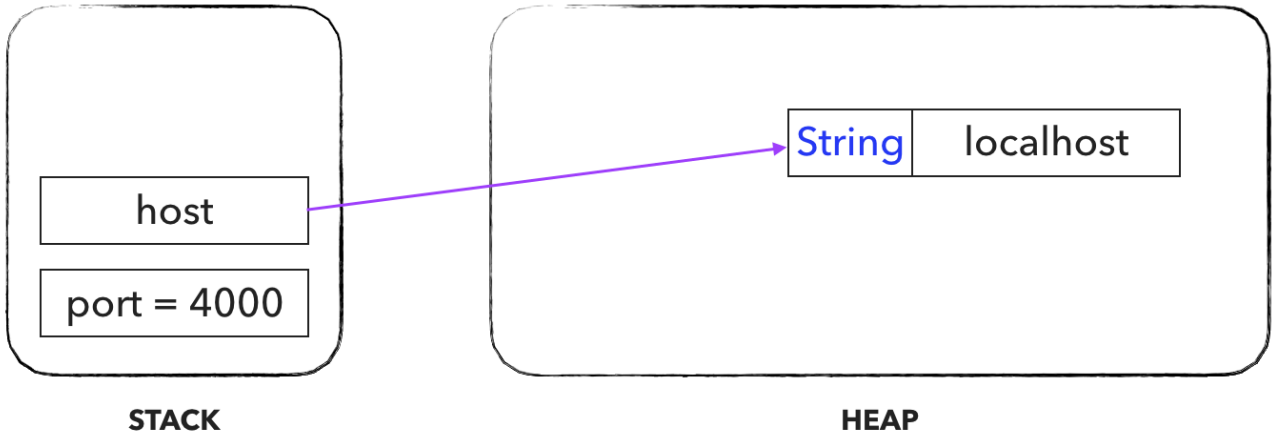

Class Loader SubSystem
Loading
Linking
Initialization
Class File
Bootstrap Class Loader
Extension Class Loader
Application Class Loader
Verify
Prepare
Resolve
Initialization
Stack Area
PC Registers
Method Area
Heap Area
Thread 1
Thread 2
Thread n
Stack Frame
LVA
OS
FD
PC Register for Thread 1
PC Register for Thread 2
PC Register for Thread n
Native Method Stack
Runtime Data Areas
Execution Engine
JIT Compiler
Intermediate Code generator
CodeOptimizer
TargetCode Generator
Profiler
Interpreter
Garbage Collection
Native Method Interface (JNI)
Native Method Library

Stack and Heap

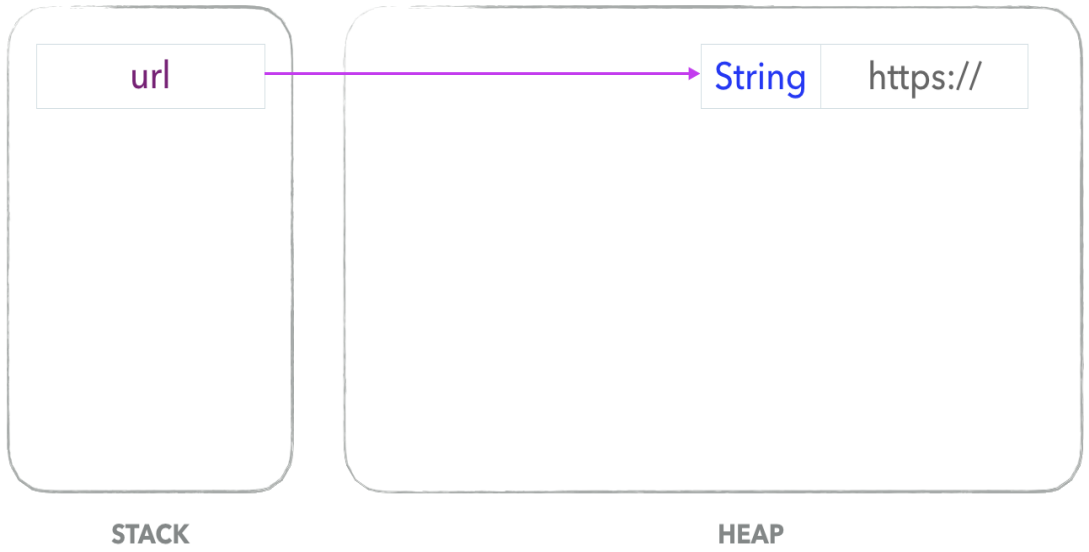


Stack and Heap

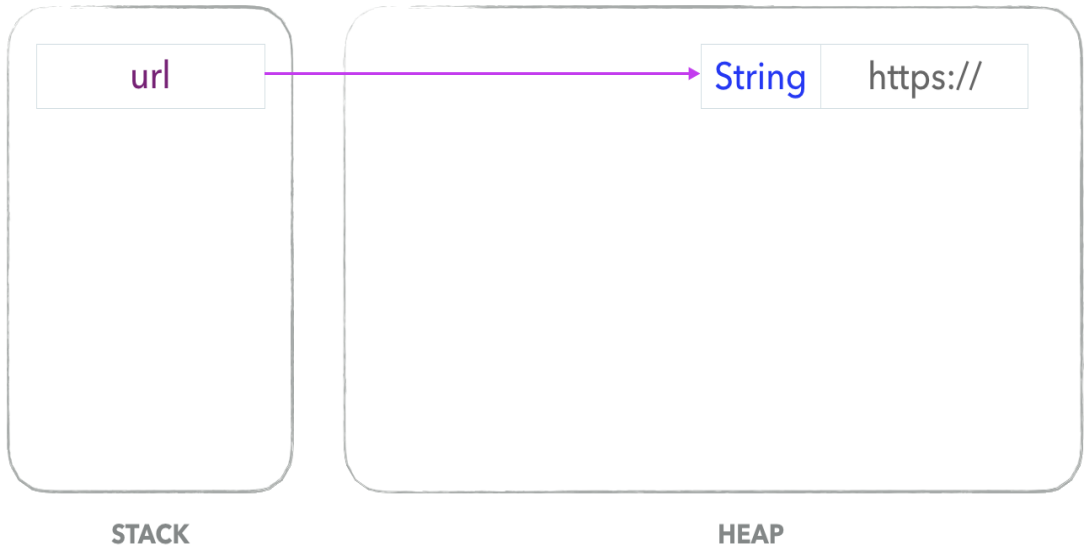


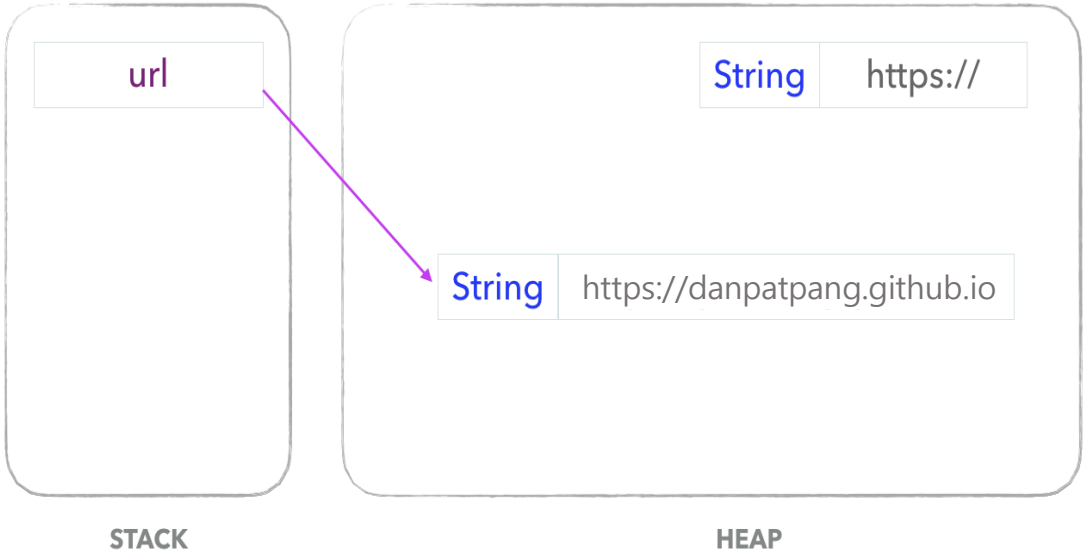


Stack and Heap (GC)

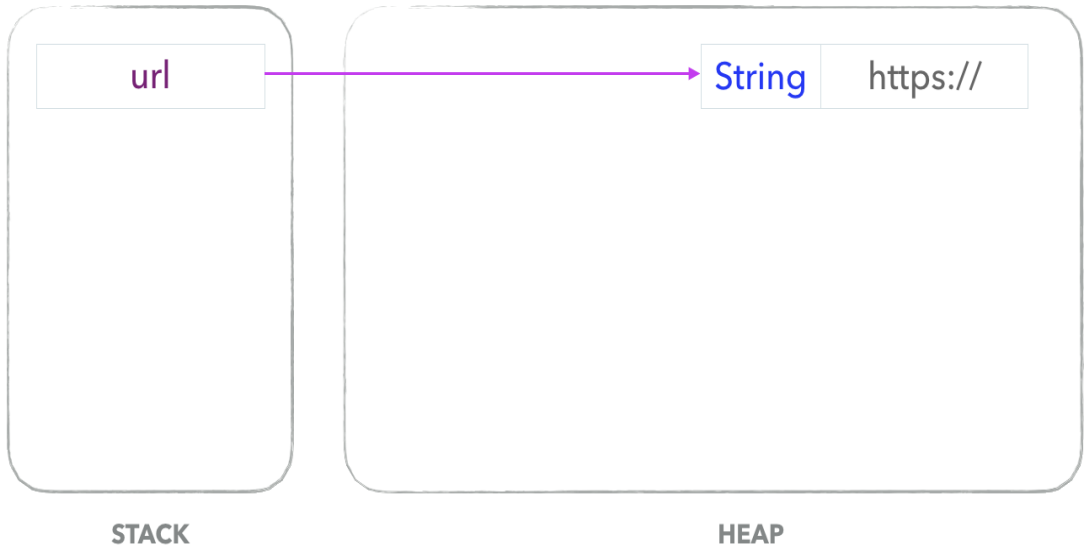


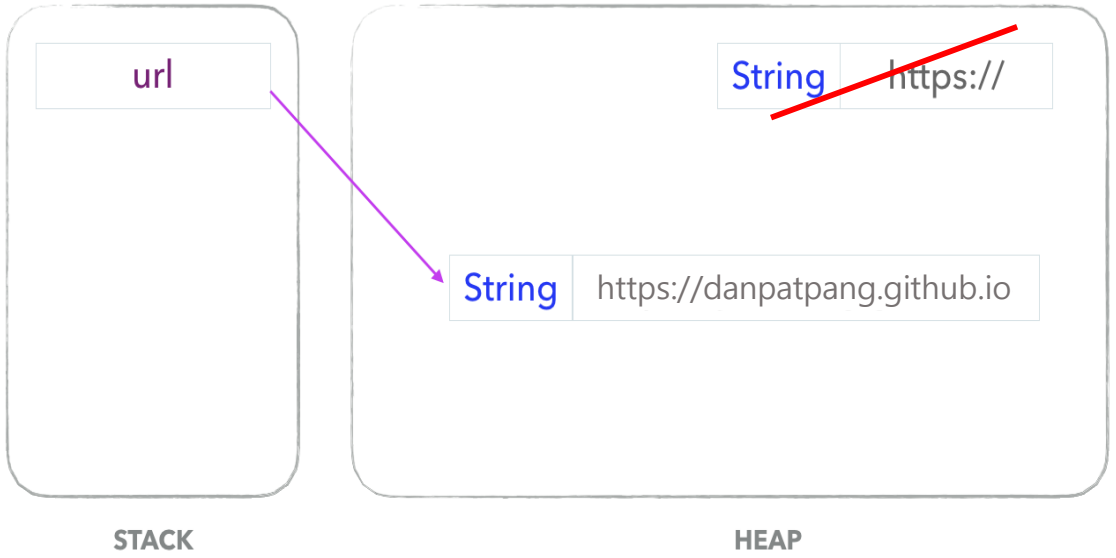
Stack and Heap (GC)

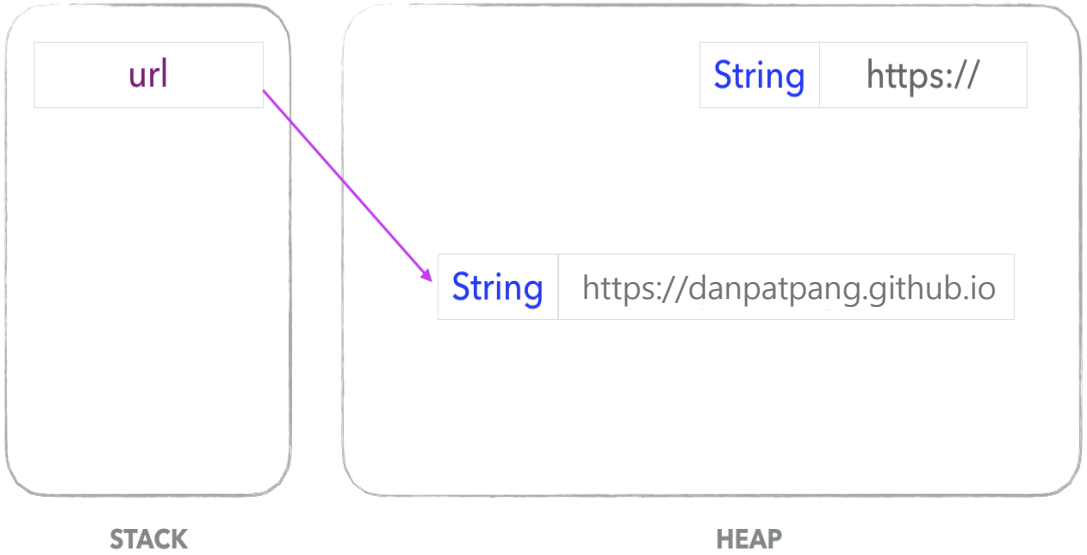




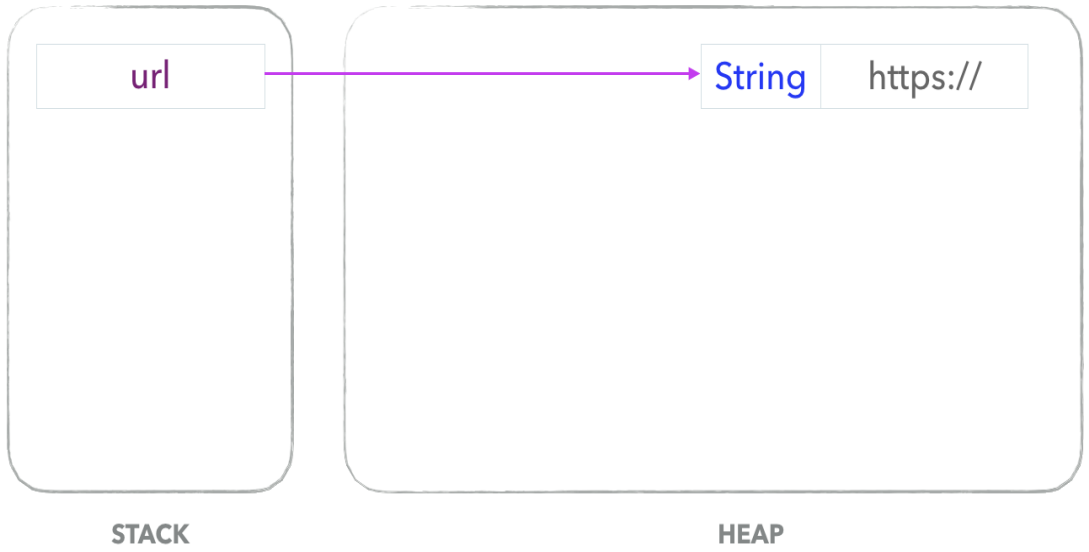
Stack and Heap (GC)

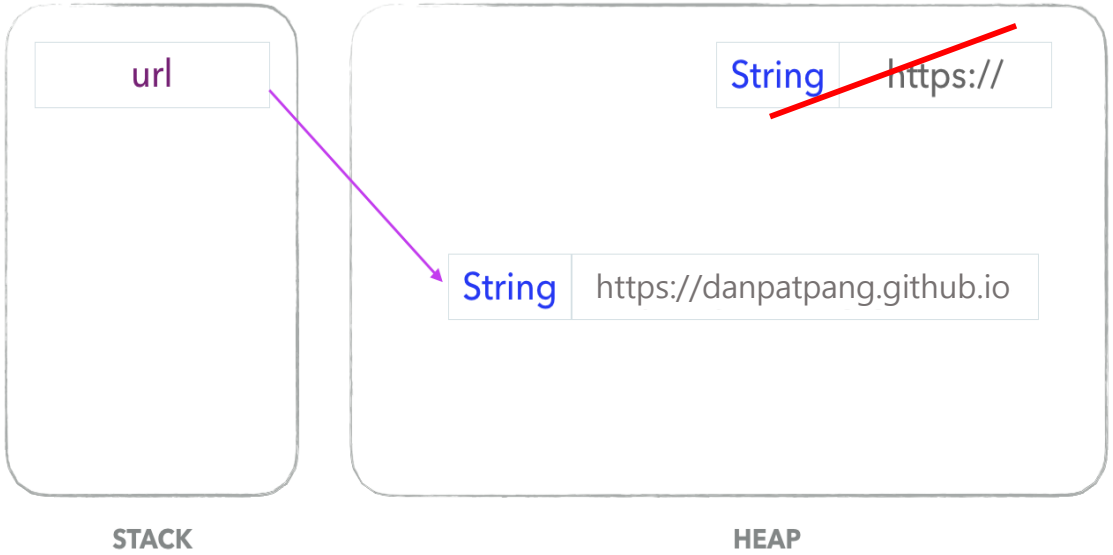


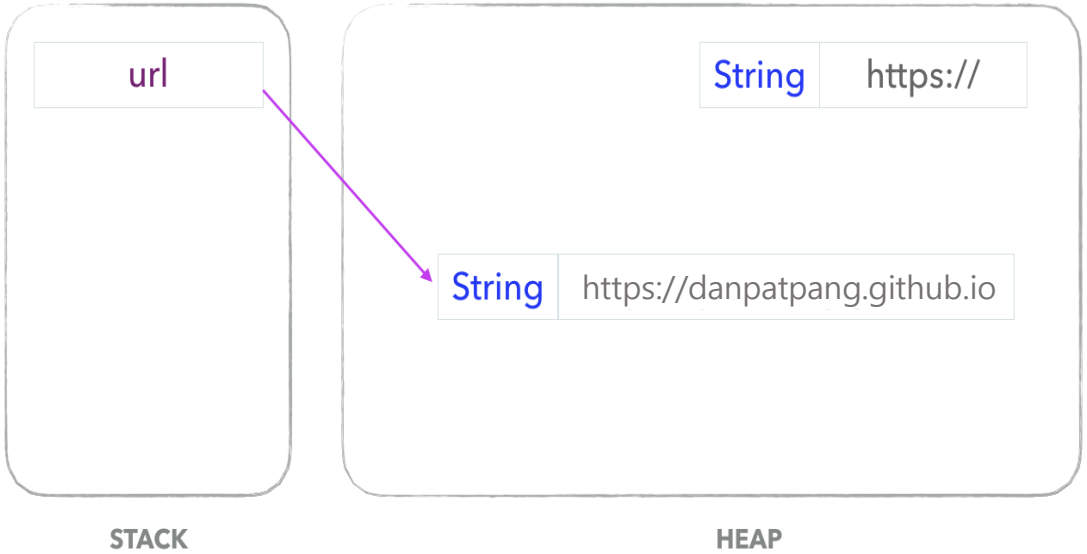


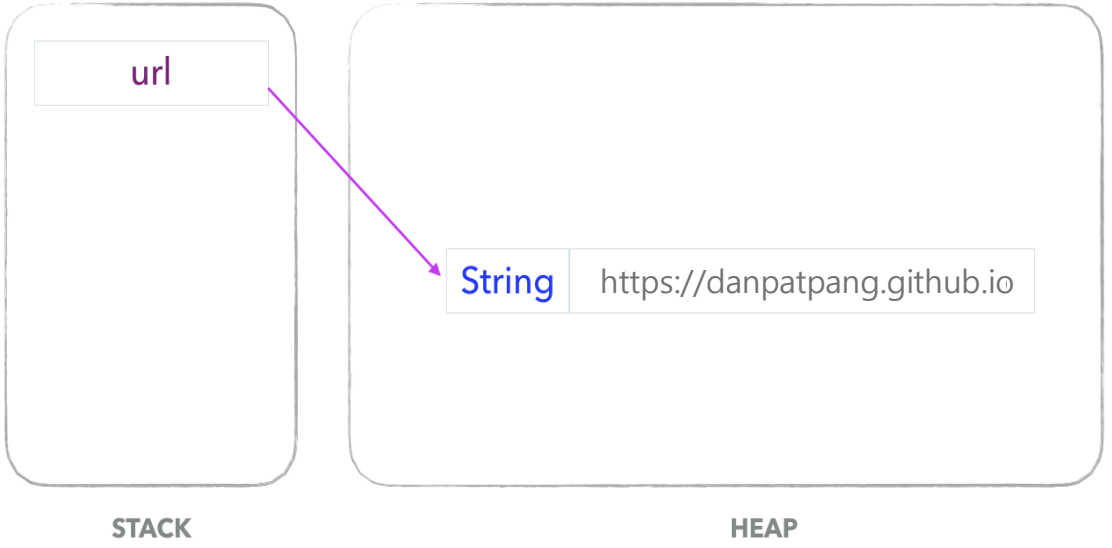


Stack and Heap (GC)









변수

"값을 저장할 수 있는 메모리 공간"

변수

일반 규칙

1. 대소문자를 구분할 수 있다.
2. 유니코드이므로 한글 변수가 가능하다.
3. 숫자로 시작할 수 없다.
4. 특수 문자는 _, $만 허용한다.
5. 예약어는 금지한다.

코딩 가이드 규칙

1. 클래스 이름의 첫 글자는 항상 대문자를 사용한다. (ex. class JavaExam01)
2. 여러 단어로 이루어진 변수의 이름은 카멜 표기법을 사용한다. (ex. numOfApple)
3. 상수는 전부 대문자, 띄어쓰기는 _로 대체한다.

주석

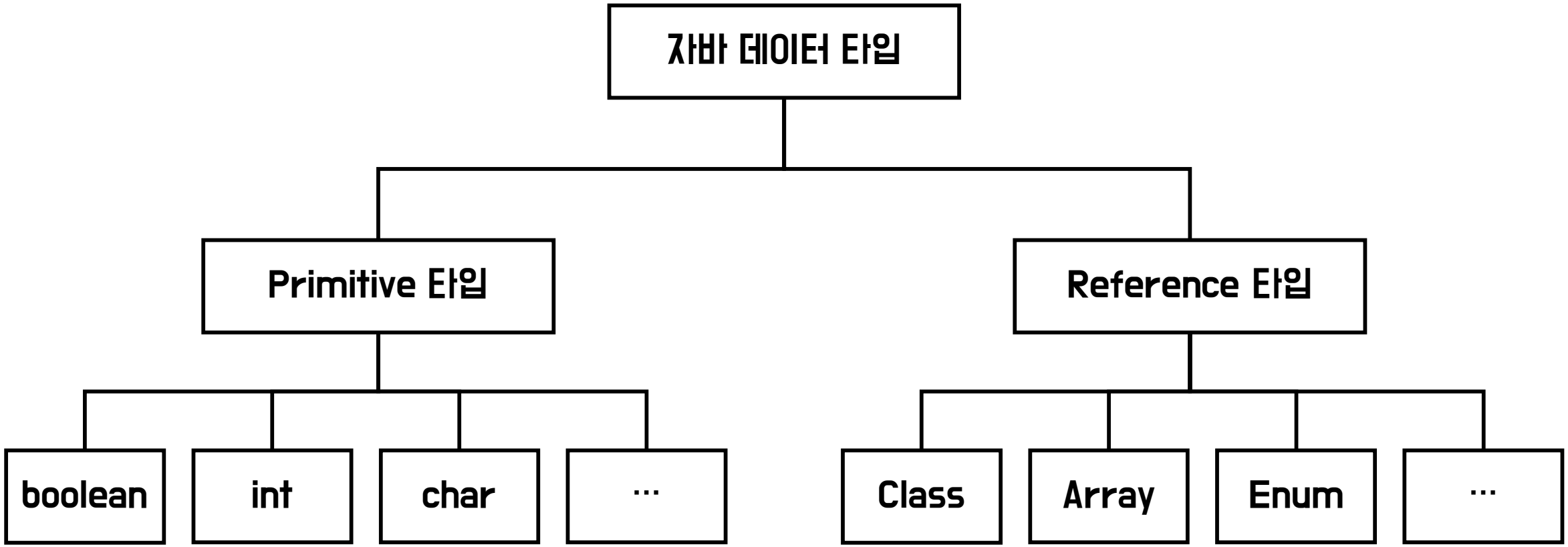
- //. /* */. /** */를 사용한다.
- javadoc을 활용하자!
- 유니코드를 주의하자!

어노테이션	설명
@author	작성자
@exception	메서드에서의 예외 확인
@param	메서드의 매개변수
@return	메서드의 반환 값
@see	다른 주제에 관한 링크
@since	릴리즈 기록
@throws	메서드에서의 예외
@version	클래스의 버전

주석

comment.UnicodeExam

데이터 타입

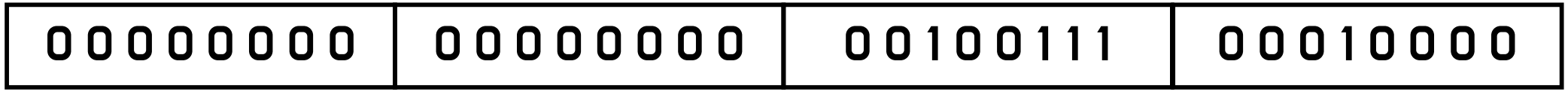

자바 데이터 타입
Primitive 타입
Reference 타입
boolean
int
char
…
Class
Array
Enum
…

데이터 타입

타입	데이터	메모리 크기	범위
boolean	참과 거짓	1 Byte	true, false
char	문자	2 Byte	유니코드 문자
byte	정수	1 Byte	-128 ~ 127
short		2 Byte	-32,768 ~ 32,767
int		4 Byte	-2,147,483,648 ~ 2,147,483,647
long		8 Byte	-9,223,372,036,854,775,808 ~ 9,223,372,036,854,775,807
float	실수	4 Byte	-3.4E38 ~ 3.4E38
double		8 Byte	-1.7E308 ~ 1.7E308

타입 변환

int num = 10000;

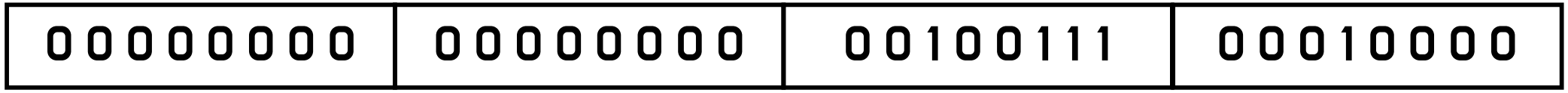


byte num2 = (byte) num;

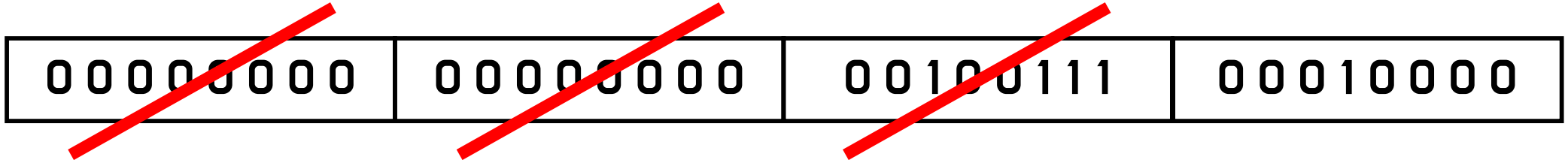
?

타입 변환

int num = 10000;



byte num2 = (byte) num;



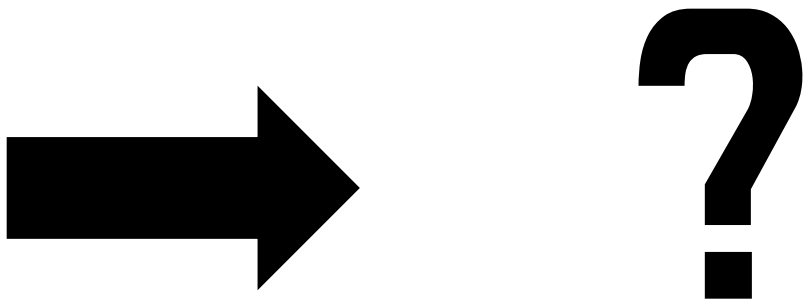
타입 변환

```
int num1 = (int) (char) (byte) -1;

int num2 = (int) (short) (byte) -1;

System.out.println(num1);

System.out.println(num2);
```

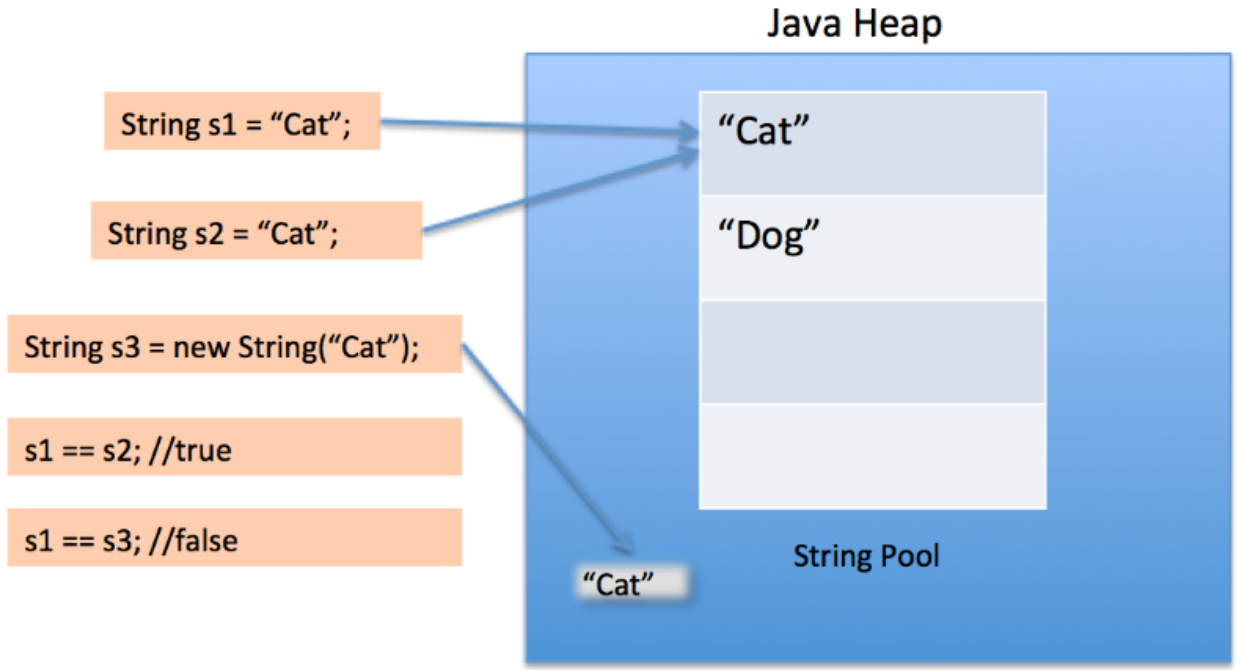


String

일반적으로 문자들의 배열.

자바에서는 문자들의 배열을 나타내는 객체.

(메모리를 더 효율적으로 관리하기 위해!)



연산자

1. 증감 연산자
2. 산술 연산자
3. 비교 연산자
4. 비트 연산자
5. 논리 연산자
6. 대입 연산자
7. 삼항 연산자

연산자 우선 순위

우선 순위	연산자	내용
1(높음)	( ), [ ], .	괄호, 대괄호, 소수점
2	!(논리의 not), ~(비트의 not), +(양수), -(음수), ++, --	단항 연산자
3	*, /, %(나머지 값)	산술 연산자
4	+, -	
5	<<, >>, <<<	쉬프트 연산자
6	<, <=, >, >=	관계 연산자
7	==, !=	
8	&(AND)	비트 연산자
9	^(XOR)	
10	|(OR)	
11	&&(AND)	논리 연산자
12	||(OR)	
13	조건? A:B (조건이 참이면 A, 거짓이면 B를 실행)	삼항 연산자
14	=, +=, -=, *=, %=, <<=, >>=, &=, ^=, ~=	대입, 할당 연산자

증감 연산자

연산자	사용법	설명
++	++op1 op1++	op1 = op1 + 1
--	--op1 op1--	op1 = op1 - 1

```java
public class IncreaseOperator {
  public static void main(String[] args) {
    int num = 1;
    System.out.println(++num);
    System.out.println(num);
    System.out.println(num++);
    System.out.println(num);
  }
}
```

산술 연산자

연산자	사용법	설명
+	op1 + op2	
-	op1 - op2	
*	op1 * op2	
/	op1 / op2	
%	op1 % op2	op1을 op2로 나눈 나머지

산술 연산자

operator.OperatorExam02

산술 연산자

자바의 산술 연산자는 **int** 형

(단, **리터널** 연산의 경우 예외)

```
public class OperatorExam2 {
  public static void main(String[] args) {
    char x = 'A';
    char y = 'A' + 1;
    char z = (char) (x + 1);
    System.out.println(y);
    System.out.println(z);
  }
}
```

비교 연산자

연산자	사용법	설명
〉	op1 〉 op2	op1이 op2보다 큰 경우 true
〉=	op1 〉= op2	op1이 op2보다 크거나 같은 경우 true
〈	op1 〈 op2	op1이 op2보다 작은 경우 true
〈=	op1 〈= op2	op1이 op2보다 작거나 같은 경우 true
==	op1 == op2	Op1과 op2가 같은 경우 true
!=	op1 != op2	Op1과 op2가 다를 경우 true

비트 연산자

연산자	사용법	설명
&	op1 & op2	비트 단위의 논리곱(AND)
|	op1 | op2	비트 단위의 논리합(OR)
^	op1 ^ op2	비트 단위의 배타적 논리합(XOR)
~	~op1	비트 단위의 보수(부정)
>>	op1 >> op2	op1을 op2만큼 오른쪽으로 이동(부호 확장O)
<<	op1 << op2	op1을 op2만큼 왼쪽으로 이동 (이동 시 빈칸은 0으로 채운다.)
>>>	op1 >>> op2	op1을 op2만큼 오른쪽으로 이동(부호 확장X)

논리 연산자

연산자	사용법	설명
&&	op1 && op2	op1과 op2 모두 true일 경우 true
||	op1 || op2	op1과 op2 중 하나라도 true일 경우 true
!	!op1	op1이 true 이면 false, false이면 true

대입 연산자

연산자	사용법	설명
+=	op1 += op2	op1 = op1 + op2
-=	op1 -= op2	op1 = op1 - op2
*=	op1 *= op2	op1 = op1 * op2
/=	op1 /= op2	op1 = op1 / op2
%=	op1 %= op2	op1 = op1 % op2
&=	op1 &= op2	op1 = op1 & op2
|=	op1 |= op2	op1 = op1 | op2
^=	op1 ^= op2	op1 = op1 ^ op2

A += B와 A = A + B는 같은 것일까?

A += B와 A = A + B

(1) A += B

A = (Type) ((A) op (B))

Type은 A를 따라간다.

(2) A = A + B

A = A op B

자바의 기본 산술 연산은 int.

```
public class OperatorExam {
  public static void main(String[] args) {
    short x = 0;
    short y = 0;
    int i = 123456;

    x += i;
    System.out.println(x);
    y = y + i;
    System.out.println(y);
  }
}
```

삼항 연산자

조건? 수식 1 : 수식 2;

- 조건이 true일 경우 수식 1 실행.
- 조건이 false일 경우 수식 2 실행.

```java
public class TernaryOperator {
  public static void main(String[] args) {
    int num = 3;

    num = (num == 3) ? 1 : 5;
    System.out.println(num);

    num = (num == 3) ? 1 : 5;
    System.out.println(num);
  }
}
```

삼항 연산자

결과는?

```
public class TernaryOperator {
  public static void main(String[] args) {
    char x = 'X';
    int i = 0;
    System.out.print(true? x : 0);
    System.out.print(false? i : x);
  }
}
```

삼항 연산자

결과는? X88

1. 피연산자의 자료형이 같은 경우,
해당 자료형으로 결과를 낸다.

2. 피연산자가 int로 변환 가능한 상수라면,
수식 1의 자료형으로 결과를 낸다.

3. 위의 경우에 해당하지 않는 경우,
피연산자 중 큰 자료형으로 결과를 낸다.

```java
public class TernaryOperator {
  public static void main(String[] args) {
    char x = 'X';
    int i = 0;
    System.out.print(true? x : 0);
    System.out.print(false? i : x);
  }
}
```

Scanner

입력을 받을 때 사용하는 객체.

jdk1.5부터 추가.

java.util.Scanner에 위치.

메서드	설명
Next()	다음 토큰을 가져온다.
nextInt()	다음 int 토큰을 가져온다.
nextLine()	다음 라인을 가져온다.
hasNextLine()	다음 라인이 있으면 true.

```java
import java.util.Scanner;

public class ScannerExam {
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);

        System.out.println(scan.nextLine());
        System.out.println(scan.next());
        System.out.println(scan.nextInt());

        scan.close();
    }
}
```

Scanner

scanner.ScannerExam01, 02

결과는?

```java
public class ScannerTest {
  public static void main(String[] args) {
    Scanner scan = new Scanner(System.in);
    int num;
    String str;

    System.out.print("num 입력 : ");  // 10
    num = scan.nextInt();
    System.out.print("str 입력 : ");  // apple
    str = scan.nextLine();

    System.out.println();
    System.out.println("num : " + num);
    System.out.println("str : " + str);

    scan.close();
  }
}
```

num 입력 : 10

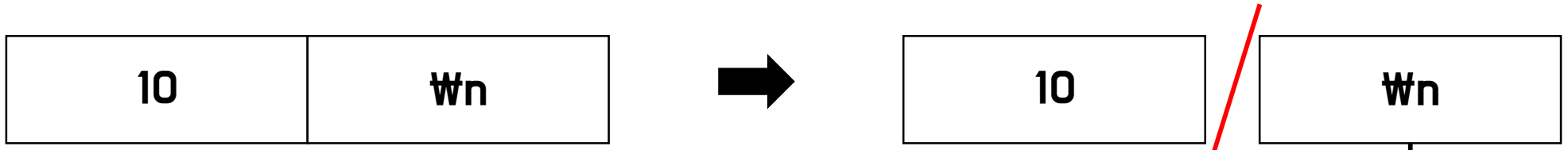
str 입력 :

num : 10

str :

결과는?

Scanner.nextInt()



Scanner.nextLine()

if 문

condition.ConditionExam01

switch-case 문

condition.ConditionExam02

for 문

loop.LoopExam01, 02

for 문

loop.LoopExam01, 02

while 문, do while 문

loop.LoopExam03

배열

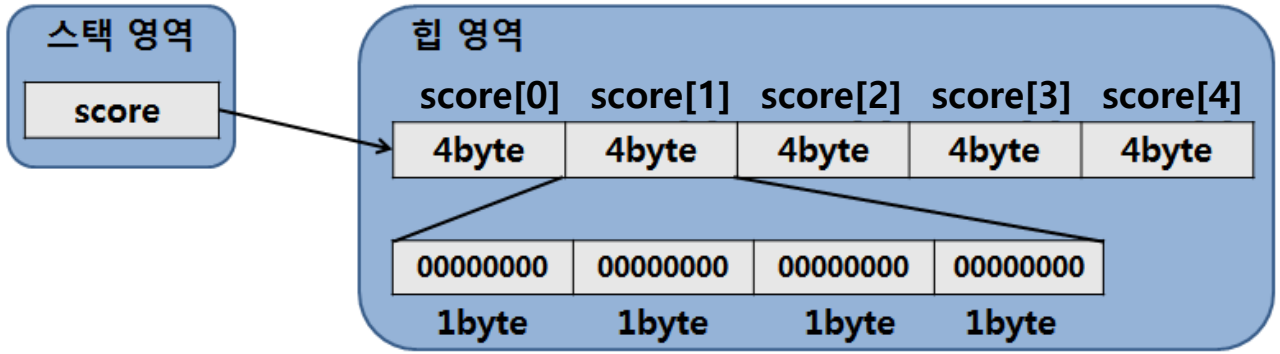
"같은 타입의 여러 변수를 하나의 묶음으로 다루는 것"

int score[] 또는 int[] score로 선언

초기화 방법

1. int[] score = {1, 2, 3, 4, 5}
2. int[] score = new int[5];
3. int[] score;

   score = new int[] {1, 2, 3, 4, 5};



배열의 초기 값

분류	데이터 타입	초기값
기본 타입(정수)	Byte[] Char[] Short[] Int[] Long[]	0 '₩u0000' 0 0 0L
기본 타입(실수)	Float[] Double[]	0.0f 0.0
기본 타입(논리)	Boolean[]	False
참조 타입	클래스[] 인터페이스[]	Null Null

자바 코딩 스타일

- 오라클 자바 코딩 스타일 가이드

https://www.oracle.com/technetwork/java/codeconventions-150003.pdf

- 구글 자바 코딩 스타일 가이드

https://google.github.io/styleguide/javaguide.html

Q&A

감사합니다!