

JAVA

1. 참조 타입

2. 참조 타입의 연산

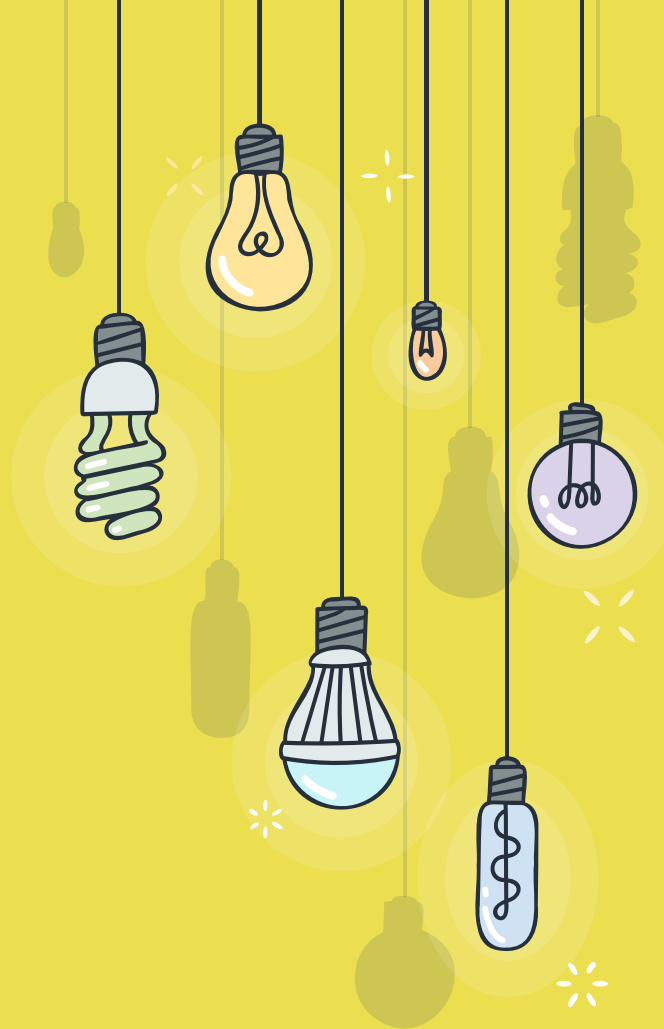
3. STRING 타입

4. ARRAY 타입



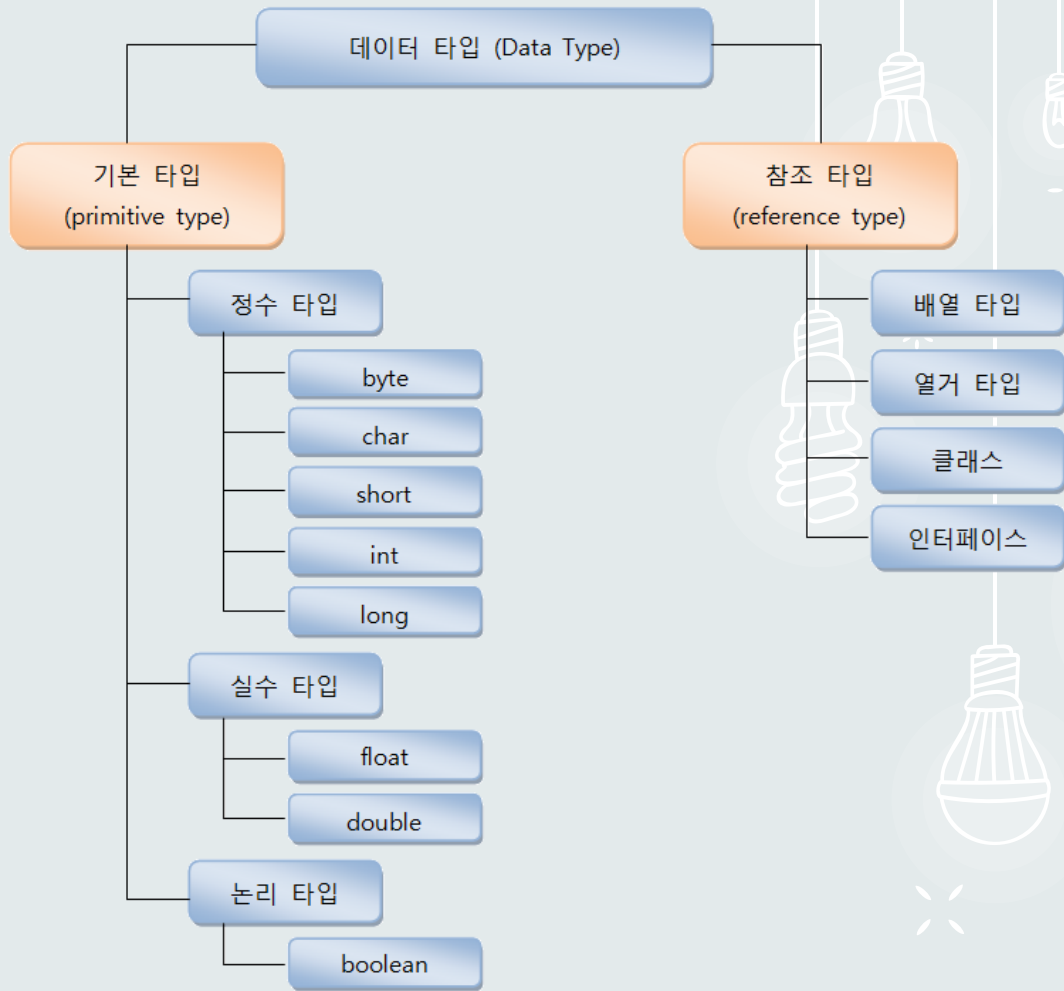
1

참조 타임



* 참조 타입

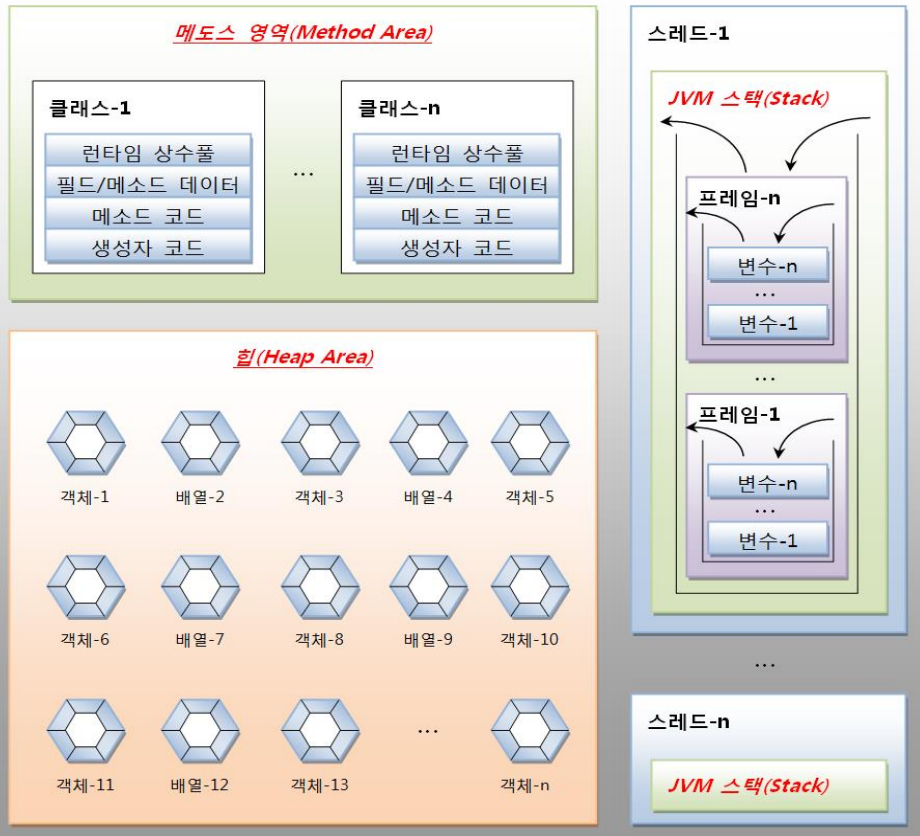
+ 데이터 타입 분류



* 메모리 영역

- + JVM이 사용하는 메모리 영역
 - × OS에서 할당 받은 메모리 영역(Runtime Data Area)을 세 영역으로 구분

Runtime Data Area



* 메모리 영역

+ JVM이 사용하는 메모리 영역

× 메소드 영역

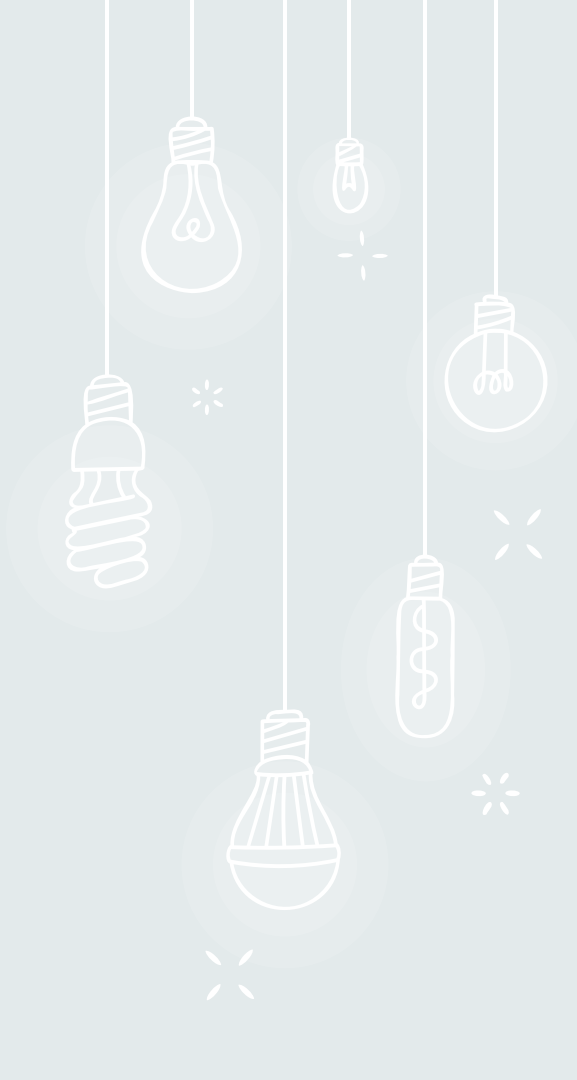
- ◆ JVM 시작할 때 생성
- ◆ 로딩된 클래스 바이트 코드 내용을 분석 후 저장
- ◆ 모든 스레드가 공유

× 힙 영역

- ◆ JVM 시작할 때 생성
- ◆ 객체/배열 저장
- ◆ 사용되지 않는 객체는 Garbage Collector가 자동 제거

× JVM 스택

- ◆ 스레드 별 생성
- ◆ 메소드 호출할 때마다 Frame을 스택에 추가(push)
- ◆ 메소드 종료하면 Frame 제거(pop)



* 메모리 영역

+ 변수의 메모리 사용

- ✕ 기본 타입 변수 - 실제 값을 변수 안에 저장
- ✕ 참조 타입 변수 - 주소를 통해 객체 참조

[기본 타입 변수]

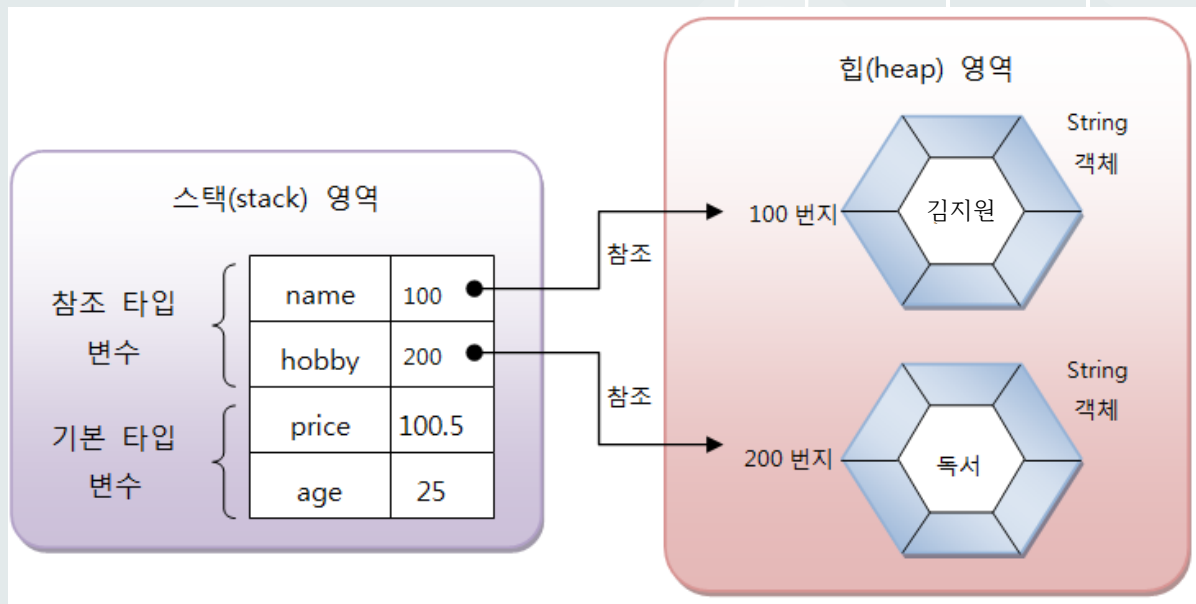
```
int age = 25;
```

```
double price = 100.5;
```

[참조 타입 변수]

```
String name = "김지원"
```

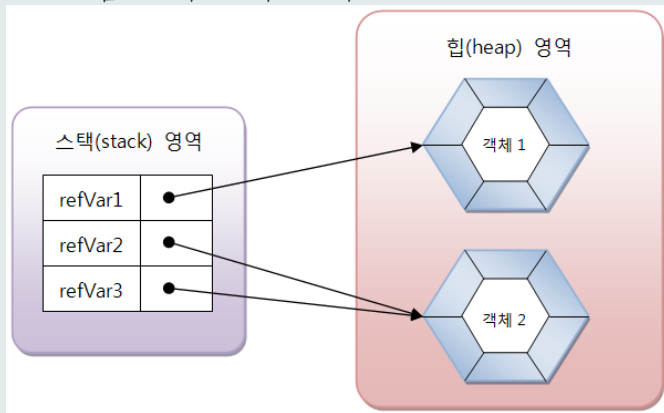
```
String hobby = "독서"
```



* 기본타입과 참조타입의 비교연산

+ 변수에서의 ==, != 연산

- ✕ 기본 타입: byte, char, short, int, long, float, double, boolean
 - ◆ 의미: 변수의 값이 같은지 다른지 조사
- ✕ 참조 타입: 배열, 열거, 클래스, 인터페이스
 - ◆ 의미: 동일한 객체를 참조하는지 다른 객체를 참조하는지 조사

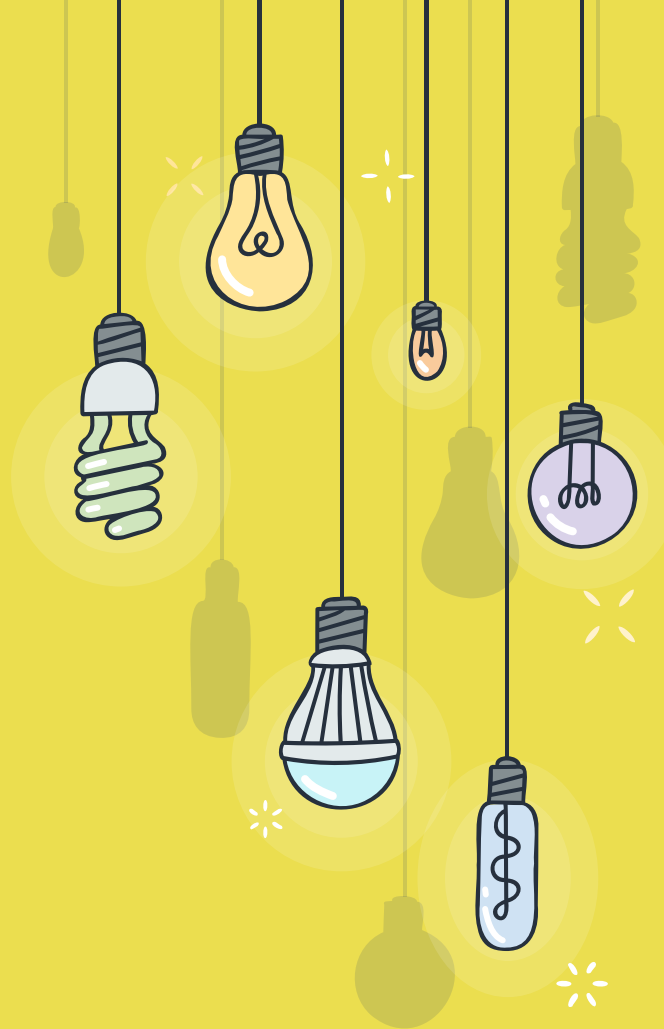


refVar1 == refVar2 결과: false
refVar1 != refVar2 결과: true

refVar2 == refVar3 결과: true
refVar2 != refVar3 결과: false

2

STRING

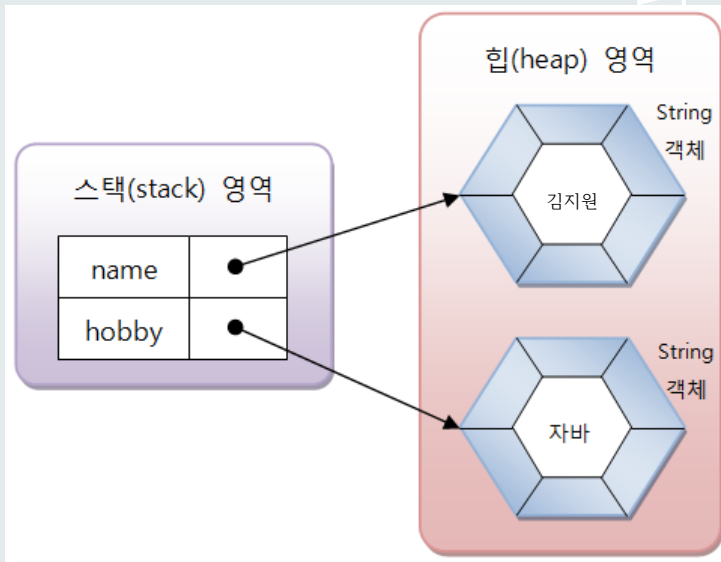


* STRING

+ String 타입

× 문자열을 저장하는 클래스 타입

```
String name = "김지원"  
String hobby = "독서"
```



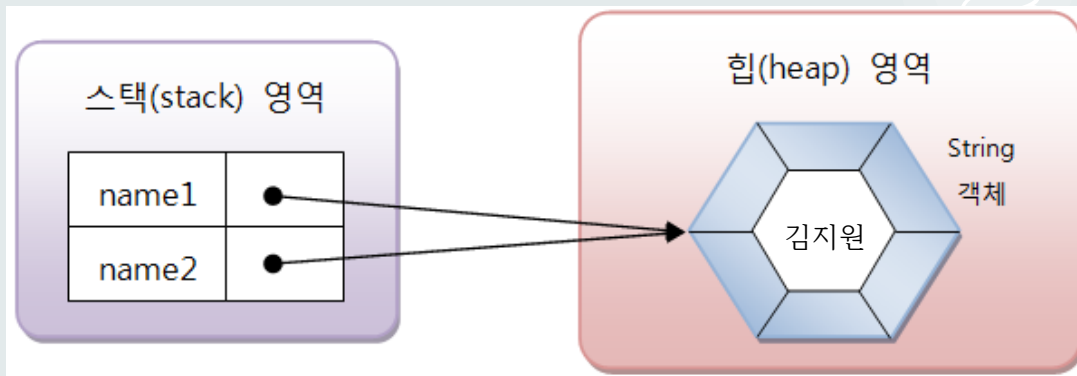
* STRING

+ String 타입

- × 문자열 리터럴 동일하다면 String 객체 공유

```
String name1 = "김지원"
```

```
String name2 = "김지원"
```



* STRING

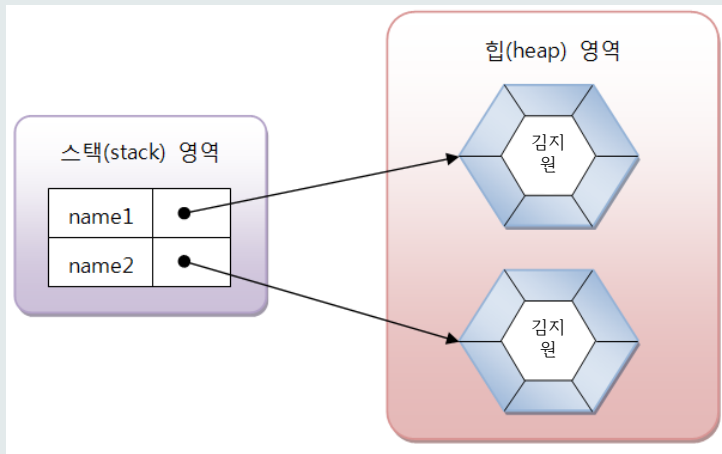
+ String 타입

× new 연산자를 이용한 String 객체 생성

- ◆ 힙 영역에 새로운 String 객체 생성
- ◆ String 객체를 생성한 후 번지 리턴

```
String name1 = new String("김지원")  
String name2 = new String("김지원")
```

```
name1.equals(name2)
```



* STRING

+ String 클래스의 주요 메서드

메서드	설명
char charAt(index)	문자열에서 해당 위치(index)에 있는 문자 반환
int length()	문자열의 길이 반환
boolean equals(String str)	문자열의 내용이 같은지 확인

* STRING 타입 변환

변환타입	사용법
String → byte	byte value = Byte.parseByte("10");
String → short	short value = Short.parseShort ("10");
String → int	int value = Integer.parseInt("10");
String → long	long value = Long.parseLong("10000000000000000");
String → float	float value = Float.parseFloat ("12.345");
String → double	double value = Double.parseDouble("12.345");
String → boolean	boolean value = Boolean.parseBoolean ("true");

+ 기본타입을 문자열로 변환할 때 String.valueOf()
메서드 사용

3

ARRAY TYPE

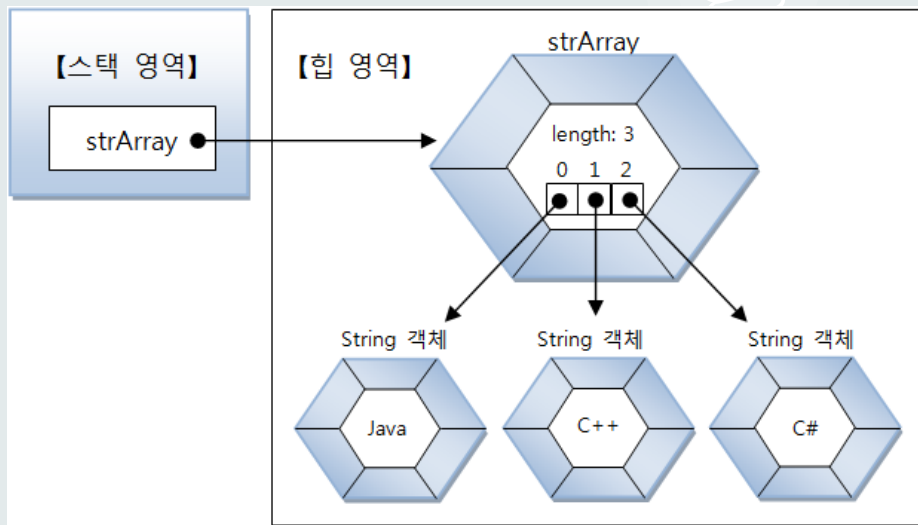


* 배열 타입

+ 배열 타입

- ✕ 기본 타입(byte, char, short, int, long, float, double, boolean) 배열
 - ◆ 각 항목에 직접 값을 가지고 있음
- ✕ 참조 타입(클래스, 인터페이스) 배열 - 각 항목에 객체의 번지 가짐

```
String[] strArray = new String[3];  
strArray[0] = "Java";  
strArray[1] = "C++";  
strArray[2] = "C#";
```



* 배열타입

+ String배열의 초기화

```
String[] name = new String[] {"Kim", "Park", "Choi"};
```

```
String[] name = {"Kim", "Park", "Choi"};
```

```
String[] name = new String[3];
```

```
name[0] = new String("Kim");
```

```
name[1] = new String("Park");
```

```
name[2] = new String("Choi");
```

```
String[] name = new String[3];
```

```
name[0] = "Kim";
```

```
name[1] = "Park";
```

```
name[2] = "Choi";
```



THANKS!

+ Any questions?

