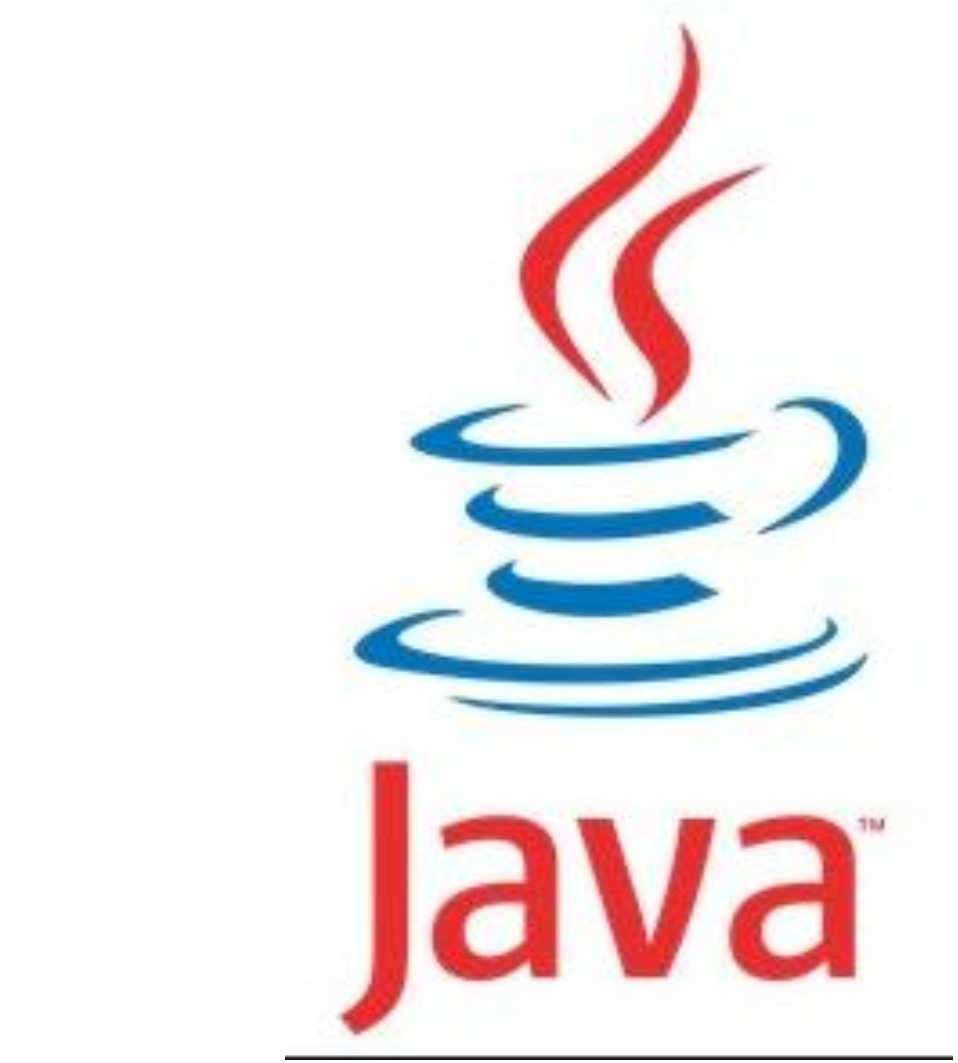


2. 변수와 자료형

1. 컴퓨터 데이터 표현
2. 변수
3. 자료형
4. 상수와 리터럴
5. 형변환
6. Scanner 클래스

컴퓨터에서 데이터 표현하기

컴퓨터는 **0**과 **1**로만 데이터를 저장 함

bit(비트**) :** 컴퓨터가 표현하는 데이터의 최소 단위로 **2**진수 하나의 값을 저장할 수 있는 메모리의 크기

byte(바이트**) : 1byte = 8bit**

0과 1의 표현– 2진수

컴퓨터는 0과 1로 자료를 표현한다. 따라서 숫자나 문자도 0과 1의 조합으로 표현된다.

10진수	2진수
0	0000000
1	0000001
2	0000010
3	0000011
4	0000100
5	0000101

10진수와 16진수

2진수로 표현하면 길이가 길어지므로 8진수나 16진수를 사용하기도 한다.

10진수	16진수
9	9
10	A
11	B
12	C
13	D
14	E
15	F
16	10

10진수, 16진수, 8진수

숫자 10을 10진수, 8진수, 16진수로 출력해보자

```
package chapter2;
public class BinaryTest {
		public static void main(String[] args) {
			int num = 10;
			int bNum = 0B1010;
			int oNum = 012;
			int hNum = 0XA;
			System.out.println(num);
			System.out.println(bNum);
				System.out.println(oNum);
				System.out.println(hNum);
		}
}
```

음의 정수 어떻게 표현할까?

정수의 가장 왼쪽에 존재하는 비트는 부호비트 입니다.

- MSB (Most Significant Bit) 가장 중요한 비트라는 뜻

음수를 만드는 방법은 2의 보수를 취한다.

0	0	0	0	0	1	0	1

1의 보수를 취한다

1	1	1	1	1	0	1	0

1을 더한다

1	1	1	1	1	0	1	1

양수와 음수 더 하기

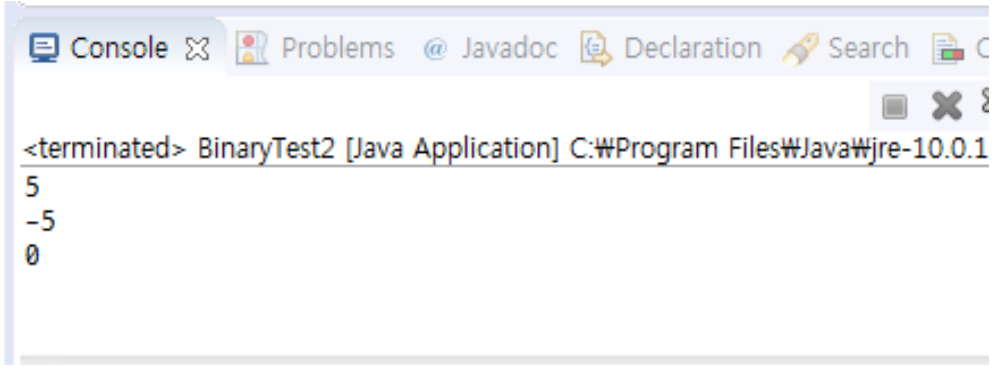
```
package chapter2;

public class BinaryTest2 {

    public static void main(String[] args) {

        int num1 = 0B00000000000000000000000000000101;  //이진수 5
        int num2 = 0B11111111111111111111111111111011;  //이진수 -5

        int sum = num1 + num2;
        System.out.println(num1);
        System.out.println(num2);
        System.out.println(sum);
    }
}
```



양수와 음수 더 하기

0	0	0	0	0	1	0	1

+

1	1	1	1	1	0	1	1

0	0	0	0	0	0	0	0

변수

사람의 나이는 해가 바뀌면 변한다. (age)

두 수를 이용하여 사칙 연산을 하면 그 결과 값은 연산자에 따라 달라진다. (result)

게임을 하면 게임 레벨이 점점 올라간다 (level)

프로그래밍에서 값(Data)를 사용
하기 위해 선언하는 것을 **변수**라 한다.

변수

프로그램에서 사용되는 자료를 저장하기 위한 공간

할당 받은 메모리의 주소 대신 부르는 이름

프로그램 실행 중에 값 변경 가능

사용되기 이전에 선언 되어야 한다.

variable 이라 함

변수의 선언과 초기화

```
int level ;          //level 이라는 이름의 변수 선언
int level = 0;     //level 변수 선언과 동시에 0 으로 초기화
```

```
int level;
level = 10 ;
```

int의 역할 : level 변수의 데이터 타입을 정의
int의 의미 : level은 정수이며 4바이트의 메모리 공간을 사용한다.

변수 선언 시 유의점

변수의 이름은 알파벳, 숫자, _, $로 구성된다.

대소문자를 구분한다.

변수의 이름은 숫자로 시작할 수 없고, 키워드도 변수의 이름으로 사용할 수 없다.

이름 사이에 공백이 있을 수 없다.

변수의 이름을 정할 때는 변수의 역할에 어울리는, 의미 있는 이름을 지어야 한다.

변수가 저장되는 공간의 특성 - 자료형

	정수형	문자형	실수형	논리형
1바이트	byte	–	–	boolean
2바이트	short	char	–	–
4바이트	int	–	float	–
8바이트	long	–	double	–

변수가 사용할 공간의 크기와 특성에 따라 자료형을 사용하여변수를 선언한다.

예) int num;

정수 자료형

자료형	바이트 크기	수의 범위
byte	1	-2^{7}~$2^{7}-1$
short	2	-2^{15}~$2^{15}-1$
int	4	-2^{31}~$2^{31}-1$
long	8	-2^{63}~$2^{63}-1$

int 로 10을 표현 할 때

byte 와 short

byte: 1바이트 단위의 자료형

동영상, 음악 파일등 실행 파일의 자료를 처리 할 때 사용하기 좋은 자료형

short: 2바이트 단위의 자료형

주로 **c/c++** 언어와의 호환 시 사용

int

자바에서 사용하는 정수에 대한 기본 자료 형

4바이트 단위의 자료 형

프로그램에서 사용하는 모든 숫자(리터럴) 은 기본적으로 int (4바이트)로 저장됨

32 비트를 초과하는 숫자는 long 형으로 처리해야 함

long

8바이트 자료형

가장 큰 정수 자료 형

숫자의 뒤에 **L** 또는 **l** 을 써서 **long** 형임을 표시해야 함

예) **int num = 12345678900;** //오류 남 **int** 의 범위 넘은 값 대입

long num = 12345678900; // 오류 남

숫자(리터럴) **12345678900** 은 기본형이 **int** 인데 **int** 의 범위가 넘는 수

=> 숫자(리터럴) **12345678900**을 **long**으로 처리하도록 명시

long num = 12345678900L; // 소문자 **l**을 써도 되지만 **1** 과 구분하기 위해 대문자로 씀

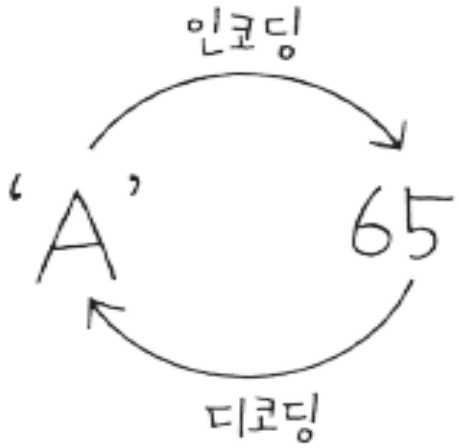
char - 문자 자료형

컴퓨터에서는 문자도 내부적으로는 비트의 조합으로 표현

자바에서는 문자를 2 바이트로 처리

인코딩 - 각 문자에 따른 특정한 숫자 값(코드 값)을 부여

디코딩 – 숫자 값을 원래의 문자로 변환



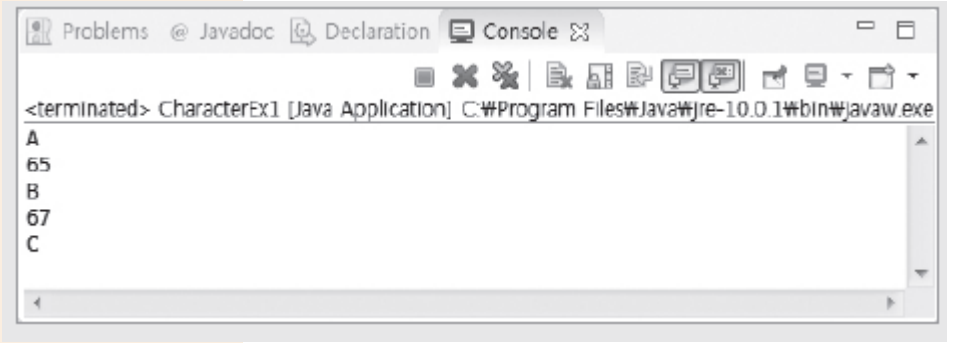
char 사용하기

```
package chapter2;

public class CharacterEx1 {
   public static void main(String[ ] args) {
      char ch1 = 'A';
      System.out.println(ch1);          //문자 출력
      System.out.println((int)ch1);     //문자에 해당하는 정수 값(아스키 코드 값) 출력

      char ch2 = 66;                    //정수 값 대입
      System.out.println(ch2);          //정수 값에 해당하는 문자 출력

      int ch3 = 67;
      System.out.println(ch3);          //문자 정수 값 출력
      System.out.println((char)ch3);    //정수 값에 해당하는 문자 출력
   }
}
```



문자 세트

문자세트 : 문자를 위한 코드 값 (숫자 값) 들을 정해 놓은 세트

아스키(ASCII) : 1 바이트로 영문자, 숫자, 특수문자 등을 표현 함

유니코드 (Unicode) : 한글과 같은 복잡한 언어를 표현하기 위한 표준 인코딩

UTF-8, UTF-16 이 대표적 (https://www.unicode.org/charts/PDF/UAC00.pdf 참고)

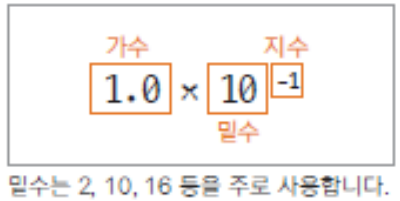
문자를 변수에 저장하면? 문자에 해당하는 코드 값이 저장됨

자바는 유니코드 UTF-16 인코딩 사용 함

float, double - 실수 자료형

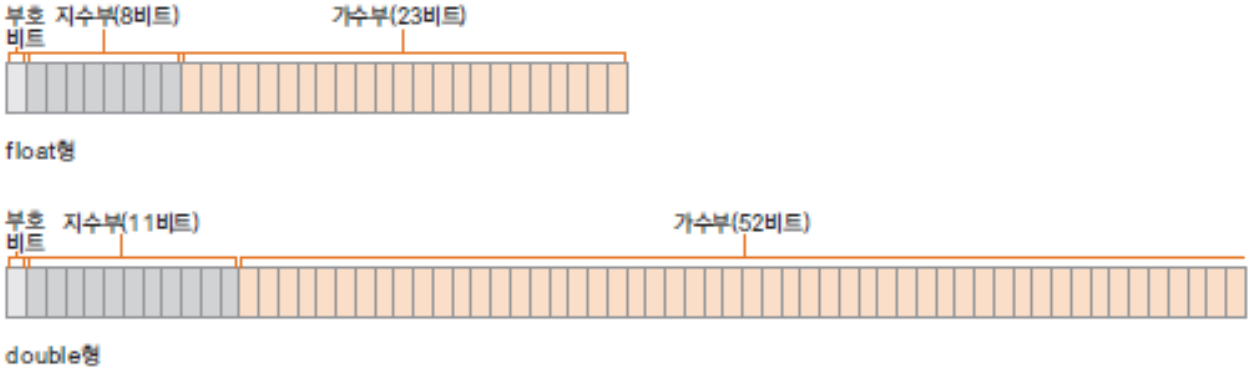
부동 소수점 방식: 실수를 지수부와 가수부로 표현함 무한의 실수를 표현하기 위한 방식

0.1 을 표현하는 방식



밑수는 2, 10, 16 등을 주로 사용합니다.

실수 자료형 : **float(4바이트) double(8바이트)**



float, double - 실수 자료형

실수는 기본 적으로 **double** 로 처리 함

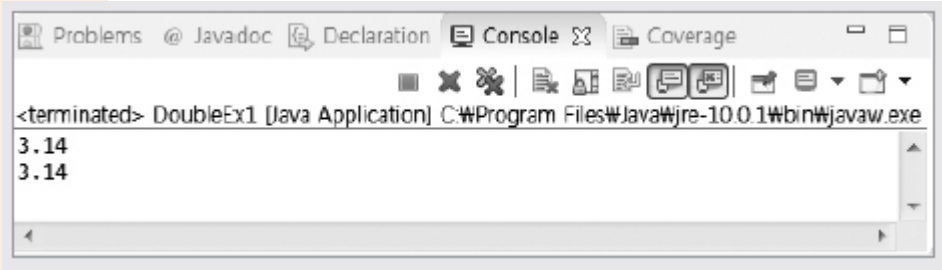
float 형으로 사용하는 경우 숫자에 **f, F** 를 명시 함

```
package chapter2;

public class DoubleEx1 {
   public static void main(String[ ] args) {
      double dnum = 3.14;
      float fnum = 3.14F;

      System.out.println(dnum);
      System.out.println(fnum);
   }
}
```

식별자



부동 소수점 방식의 오류

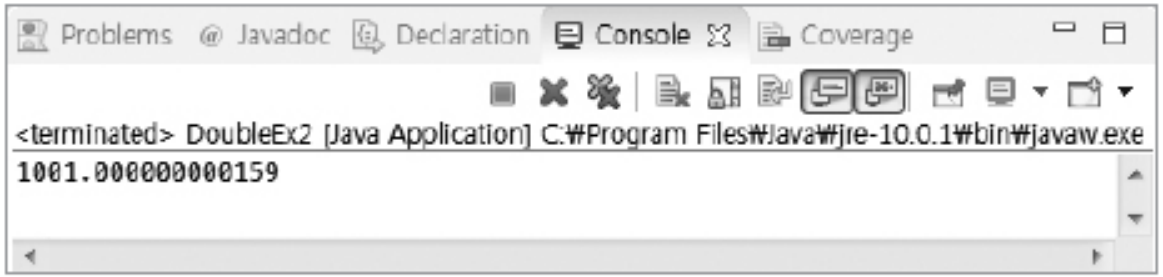
지수와 가수로 표현 되는 부동 소수 점은 0을 표현할 수 없음, 따라서 부동 소수점 방식에서는 약간의 오차가 발생할 수 있음

```
package chapter2;

public class DoubleEx2 {
    public static void main(String[ ] args) {
       double dnum = 1;

       for(int i = 0; i < 10000; i++) {
           dnum = dnum + 0.1;
       }
       System.out.println(dnum);
    }
}
```

for문은 지정한 문장을 정해진 횟수만큼 반복해서 수행하는 반복문입니다. 여기에서는 더하기를 10,000번 반복하라는 의미입니다. '04장 제어 흐름 이해하기'에서 자세히 배웁니다.



결과 값이 1001이 아님, 오차를 감수하더라도 넓은 범위의 수를 표현하기 위해 사용

boolean - 논리형

논리값 **true (참) , false(거짓)** 을 표현하는 자료형

boolean 으로 선언

```
package chapter2;

public class BooleanEx {
   public static void main(String[ ] args) {
      boolean isMarried = true;      //boolean 변수를 선언하고 초기화
      System.out.println(isMarried);
   }
}
```

자료형 없이 변수 사용하기 (자바 10)

자료형이 필요한 이유:

변수를 선언 할 때는 변수가 사용할 메모리 크기와 타입을 구분하기 위해 자료형을 사용

지역 변수 자료형 추론 **(local variable type inference) :**

변수에 대입되는 값을 보고 컴파일러가 추론

```
var num = 10;
var dNum = 10.0;
var str = "hello";
```

```
int num = 10;
double dNum = 10.0;
String str = "hello";
```

상수

상수 : 변하지 않는 값 (cf 변수 : 변하는 값)

상수를 선언 : final 키워드 사용

```
final double PI = 3.14;
final int MAX_NUM = 100;
```

final 로 선언된 상수는 다른 값을 대입 할 수 없음

```
PI = 3.15; // 에러 남
```

프로그램 내에서 변경되지 말아야 하는 값을 상수로 선언 해 두고 혹시 변경되는 경우 선언된 값만 수정

리터럴(literal)

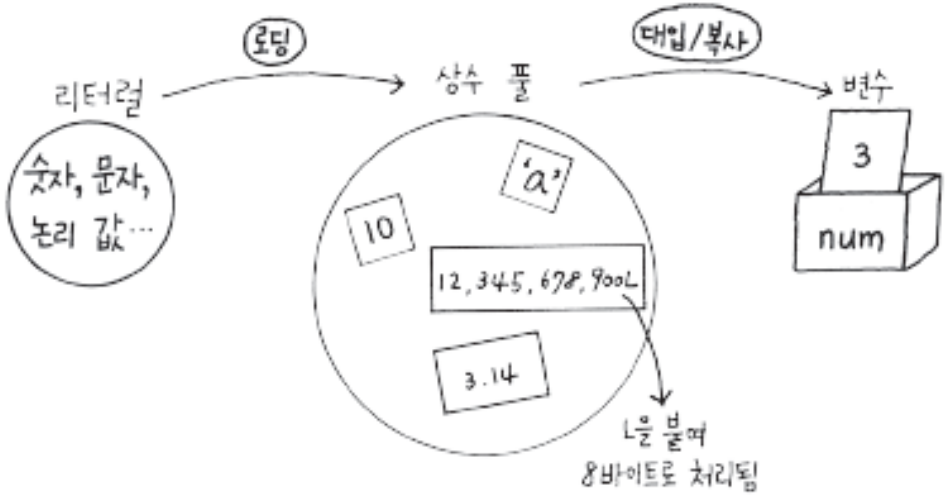
리터럴 : 프로그램에서 사용하는 모든 숫자, 값, 논리 값(예: 10, 3.14, ‘A’, true)

리터럴에 해당되는 값은 특정 메모리 공간인 상수 풀(constant pool)에 있음

필요한 경우 상수 풀에서 가져와서 사용

상수 풀에 저장 할 때 정수는 int로 실수는 double로 저장

따라서 long 이나 float 값으로 저장해야 하는 경우 식별자 (L, l, F, f)를 명시해야 함

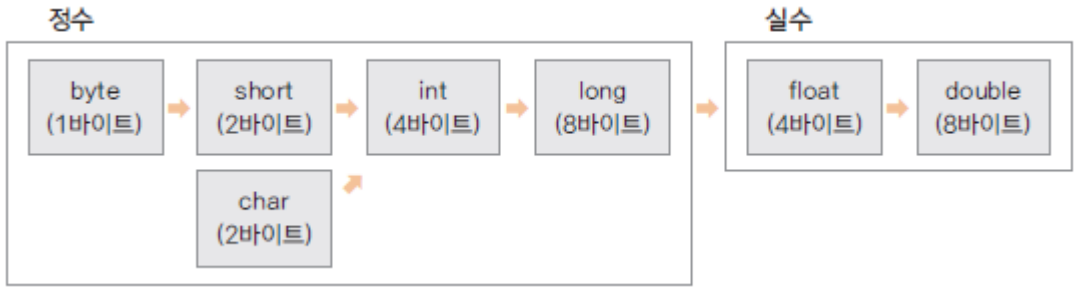


형 변환(type conversion)

자료형은 각각 사용하는 메모리 크기와 방식이 다름

서로 다른 자료형의 값이 대입되는 경우 형 변환이 일어 남

묵시적 형변환 : 작은 수 에서 큰 수로 덜 정밀한 수에서 더 정밀한 수로 대입되는 경우



예) **long num = 3; // int 값에서 long으로 자동 형 변환, L, l 을 명시하지 않아도 됨**

명시적 형 변환: 묵시적 형 변환의 반대의 경우, 변환 되는 자료 형을 명시해야 함 자료의 손실이 발생할 수 있음

예) **double dNum = 3.14;**

int num = (int)dNum; //자료형 명시

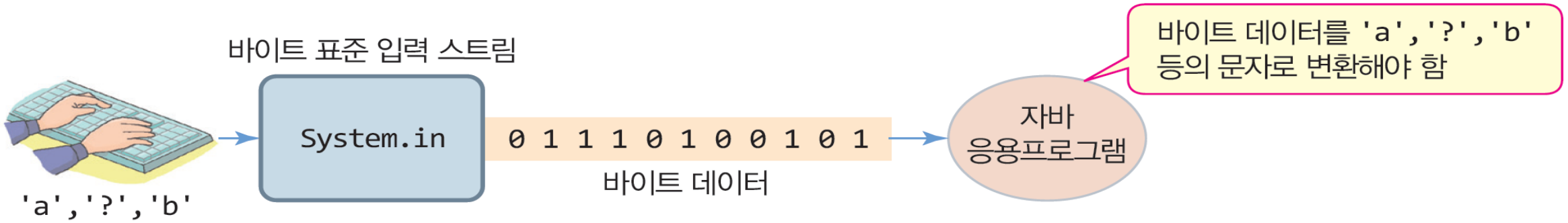
자바에서 키 입력

System.in

- 키보드로부터 직접 읽는 자바의 표준 입력 스트림
- 키 값을 바이트(문자 아님)로 리턴

System.in을 사용할 때 문제점

- 키 값을 바이트 데이터로 넘겨주므로 응용프로그램이 문자 정보로 변환해야 함



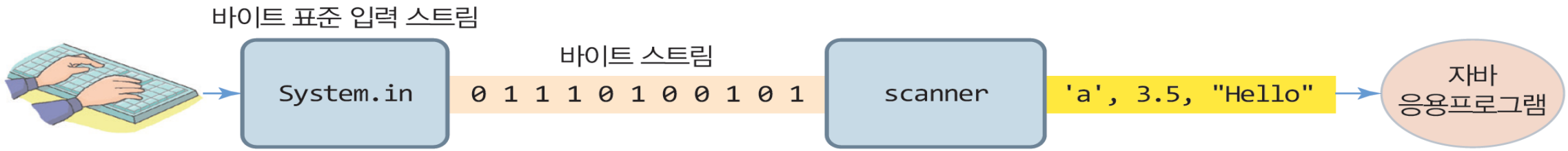
SCANNER로 쉽게 키 입력

Scanner 클래스

- System.in에게 키를 읽게 하고, 읽은 바이트를 문자, 정수, 실수, 불린, 문자열 등 다양한 타입으로 변환하여 리턴
  - java.util.Scanner 클래스

객체 생성

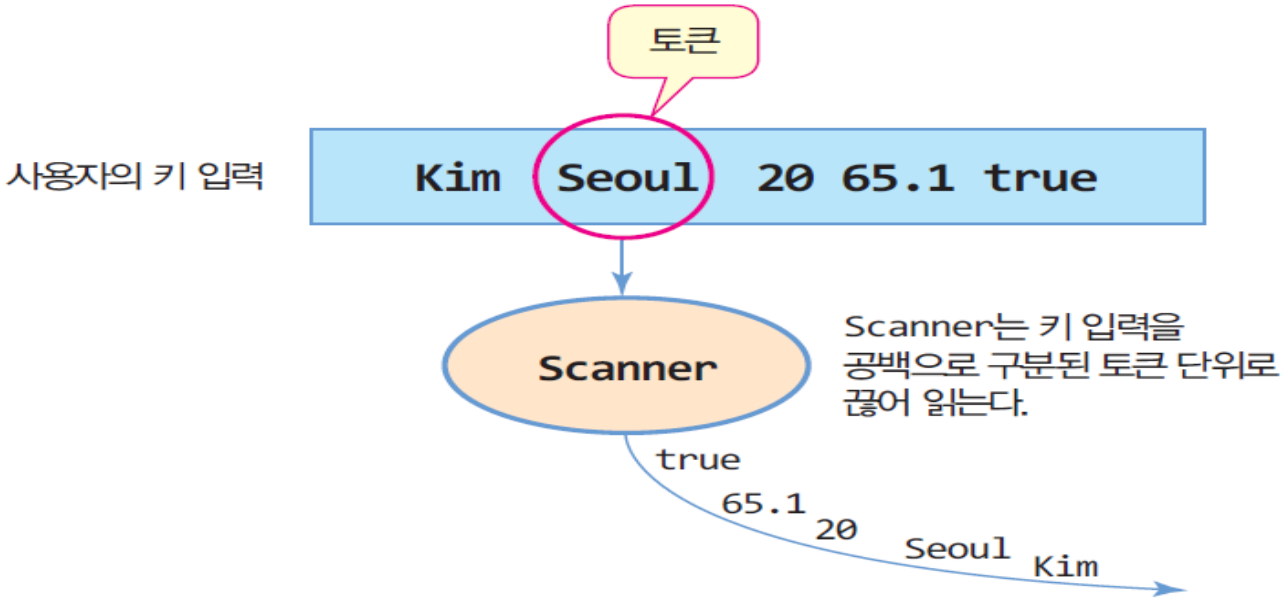
```
import java.util.Scanner; // import 문 필요
...
Scanner a = new Scanner(System.in); // Scanner 객체 생성
```



- System.in에게 키를 읽게 하고, 원하는 타입으로 변환하여 리턴

SCANNER를 이용한 키 입력

- Scanner에서 키 입력 받기
  - Scanner는 입력되는 키 값을 공백으로 구분되는 아이템 단위로 읽음
  - 공백 문자 : '\t', '\f', '\r', ' ', '\n'
- 개발자가 원하는 다양한 타입의 값으로 바꾸어 읽을 수 있음




```
Scanner scanner = new Scanner(System.in);

String name = scanner.next();              // "Kim"
String city = scanner.next();              // "Seoul"
int age = scanner.nextInt();               // 20
double weight = scanner.nextDouble(); // 65.1
boolean single = scanner.nextBoolean(); // true
```

SCANNER 주요 메소드

메소드	설명
String next()	다음 토큰을 문자열로 리턴
byte nextByte()	다음 토큰을 byte 타입으로 리턴
short nextShort()	다음 토큰을 short 타입으로 리턴
int nextInt()	다음 토큰을 int 타입으로 리턴
long nextLong()	다음 토큰을 long 타입으로 리턴
float nextFloat()	다음 토큰을 float 타입으로 리턴
double nextDouble()	다음 토큰을 double 타입으로 리턴
boolean nextBoolean()	다음 토큰을 boolean 타입으로 리턴
String nextLine()	'\n'을 포함하는 한 라인을 읽고 '\n'을 버린 나머지 문자열 리턴
void close()	Scanner의 사용 종료
boolean hasNext()	현재 입력된 토큰이 있으면 true, 아니면 입력 때까지 무한정 대기, 새로운 입력이 들어올 때 true 리턴. crtl-z 키가 입력되면 입력 끝이므로 false 리턴

SCANNER를 이용한 키 입력 예

Scanner를 이용하여 이름, 도시, 나이, 체중 독신 여부를 입력 받고 다시 출력하는 프로그램을 작성하라.

```
import java.util.Scanner;

public class ScannerEx {
    public static void main(String args[]) {
        System.out.println("이름, 도시, 나이, 체중, 독신 여부를 빈칸으로 분리하여 입력하세요");
        Scanner scanner = new Scanner(System.in);

        String name = scanner.next(); // 문자열 읽기
        System.out.print("이름은 " + name + ", ");

        String city = scanner.next(); // 문자열 읽기
        System.out.print("도시는 " + city + ", ");

        int age = scanner.nextInt(); // 정수 읽기
        System.out.print("나이는 " + age + "살, ");

        double weight = scanner.nextDouble(); // 실수 읽기
        System.out.print("체중은 " + weight + "kg, ");

        boolean single = scanner.nextBoolean(); // 논리값 읽기
        System.out.println("독신 여부는 " + single + "입니다.");

        scanner.close(); // scanner 닫기
    }
}
```

```
이름, 도시, 나이, 체중, 독신 여부를 빈칸으로 분리하여 입력하세요.
Kim Seoul 20 65.1 true
이름은 Kim, 도시는 Seoul, 나이는 20살, 체중은 65.1kg, 독신 여부는 true입니다.
```