JAVA 1강 요약

강사 박주병

1. JAVA란?

기존의 절차지향적 프로그래밍 언어의 한계를 개선하고자 개발된 객체지향적 프로그래밍 언어이다.
이전에 존재했던 C, C++ 과 같은 언어들의 복잡함과 문제점들을 개선하였고 현재는 가장 많이 쓰이고 있는 프로그래밍 언어이다.

가. 플랫폼 독립적이다.

플랫폼이란 운영체제(OS)를 말하는 것 이며 C ,C++ 같이 직접적으로 .exe 같은 실행파일을 만들어내는 언어들은 운영체제에 종속적일 수밖에 없다. 따라서 운영체제에 따라 코드 및 컴파일러가 달라져야만 했다.
그러나 JAVA는 JVM이라는 가상머신을 이용해 실행됨으로 운영체제로부터 독립적인 프로그래밍이 가능하다.
즉 하나의 소스코드로 모든 플랫폼에서 동일하게 실행이 가능한 것이다.

JAVA 프로그램

⋙

JVM(java virtual machine)

⋙

일반 프로그램

⋙

나. 객체지향적 언어이다.

기존의 절차지향적 방식의 문제점을 개선한 방식이다. 객체라는 것은 세상에 존재하는 모든 사물들을 뜻하며 그것을 다음과 같이 정의 하는 방식이다.

객체 = 데이터+기능

세상의 모든 것은 데이터+기능으로 이루어져 있다고 보고 프로그래밍에서도 이것을 그대로 반영한다는 것이다.
자동차라는 것은 기름량, 소유자이름, 차량번호, 모델명 등의 데이터와 달리다, 멈추다, 후진, 시동을 켠다와 같은 기능으로 이루어져 있다. 이러한 자동차를 클래스라고 부르며 자동차를 실제 사용 할 수 있는 형태로 실체화한 것(번호판이 부여된 k3, 아반떼, 소나타 등등 실제 존재 하는 자동차)을 객체라고 부른다.

클래스

⋙

객체(인스턴스)

클래스는 붕어빵 틀과 같은 객체를 만들기 위한 설계도라고 보면 된다. 해당 설계도를 가지고 실질적인 붕어빵을 만들어 낸다면 붕어빵 하나 하나가 모두 객체가 되는 것이다.

물론 지금 당장 객체지향적 프로그래밍을 배우기는 무리가 있다. 절차지향방식으로 기본기를 익히고 차후 객체지향

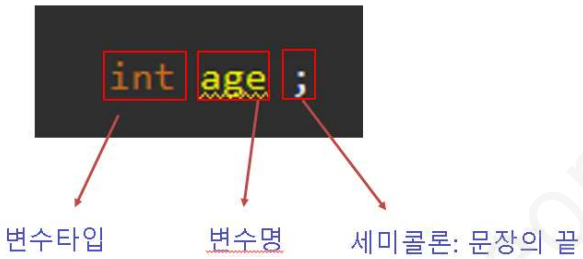
에 대해 자세히 다루도록 한다.

2. 변수

변수란 하나의 값을 저장하는 공간이다. 기존 수학의 방정식에서 쓰이던 변수와 동일하다고 보면 된다.
아래와 같은 방정식에서 x,y가 변수(변하는수)인 것이다. x의 값은 정해지지 않고 변경될수 있는 것이다.
프로그래밍에서는 이러한 변수를 이용해 값을 저장해두고 사용한다.

$$Y = x+1$$

가. 변수의 선언과 초기화



변수를 쓰기 위해선 우선 변수를 선언하여 만들어야 한다. 변수 선언 방법은 위의 그림과 같다.

int -> 변수타입은 해당 변수와 어떤 형태의 데이터를 저장할지를 정하는 것이다. int 라는 것은 정수를 저장할 때 쓴다.

age -> 변수는 값을 넣어두는 공간이라고 했다. 해당공간을 이용하기 위해선 이름이 있어야할 것이다. 변수 선언 이후에 해당 공간에 값을 넣거나 가져올 때 변수명을 이용한다.

; -> 은 모든 문장의 끝에서 반드시 적어줘야 한다. 그렇지 않으면 컴퓨터는 해당 문장이 끝난다는 것을 알수 없다. 글에서 마침표를 찍는 것과 같은 것이다.

변수는 물을 담아두는 컵이라고 생각하면 된다. 컵을 만들기만 했다면 안에 내용물이 비어 있는 상태이다.
이제 내용물을 채워 넣어보자.

20 -> 변수에 저장될 값

= -> 이것은 수학에서의 의미와 다르다. 수학에서는 같다라는 의미지만 프로그래밍에서는 우변에 있는 값을 좌변에 저장한다 라는의미이다. 이것을 대입연산자 라고 부른다.
정리 하자면 위의 코드는 int라는 데이터타입의 변수 age를 선언하고 여기에 20이라는 값을 넣은 것이다.

```
int a,b,c ;

int d=0,e=10,f=13;
```

한번에 여러 개의 변수 선언 및 초기화도 가능하다.

```
int t;

t=30;
```

선언과 초기화를 따로 해줘도 된다.

해당 변수를 사용하는 방법으로는 그냥 변수명을 쓰면 되는 것이다. 그림7은 해당 변수의 값을 모니터로 출력하는 코드이다.(System.out.println() 에 대해서는 차후에 자세히 다루도록하고 현재는 모니터에 괄호안의 값을 출력하는 기능이다 라고만 이해하자)

```
int a=10;

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

그림 7

나. 변수명 규칙

- 대소문자를 구분한다.
- 자바에서 미리 지정한 예약어는 사용할수 없다
- 숫자로 시작할수 없다.
- 특수문자는 _ 와 $만 사용 가능하다.

다. 변수명 권장사항

- 단어의 첫글자는 대문자로 한다. ex) Age, Name ,NodeList
- 상수는 모두 대문자로 한다(상수 : 값이 고정된 변수) ex) PI = 3.14, MAX, MIN

라. 주석

주석은 코드로 인식되지 않는다. 코드에 대한 설명을 달아두거나 코드 자체를 주석 처리하여 잠깐 프로그램에서 제외하기도 한다.

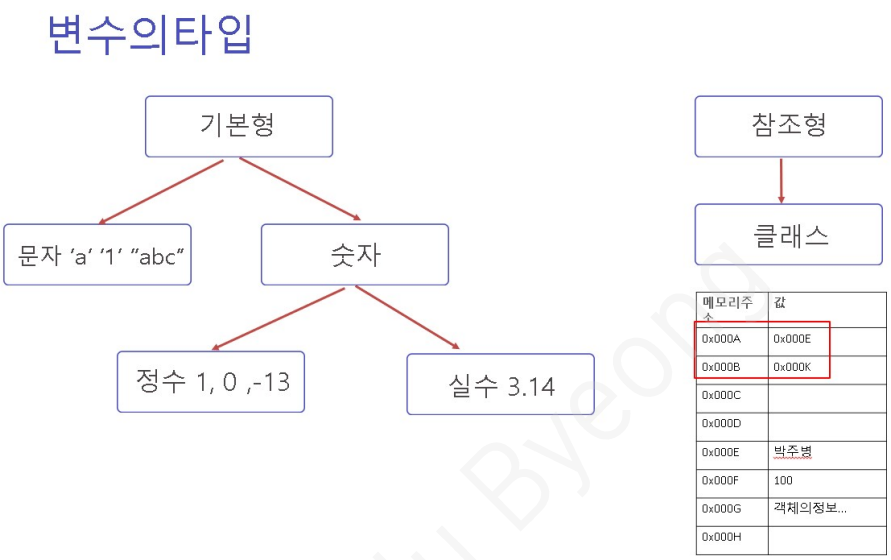
/**/ -> 이 사이를 모두 주석으로 처리한다. 줄넘김도 가능하다.
// -> 해당 줄만 주석으로 처리한다. 다음줄은 다시 코드로 인식된다.

주석

```java
package joo;

/*

Java를 처음 시작하는 예제

 */
public class Main {

    public static void main(String[] args) {

        int a=10,b=50, temp;

        temp = a;
```

```java
package joo;

//자바를 처음시작하는 예제
public class Main {

    public static void main(String[] args) {

        int Age=10;  //나이

        //System.out.println("a: "+a + " b: " + b);
```

마. 변수의 타입



변수들은 어떤 데이터를 넣을 것인가에 따라 타입을 다르게 줘야 한다. JAVA에서의 데이터 타입들은 크게 위의 그림과 같이 분류 할 수 있다. 기본형 8가지를 제외 하곤 모두 참조형이다.

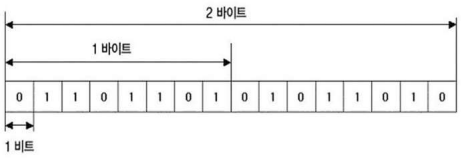
기본형 데이터 타입

종류	이름	값
논리형	boolean	True, false
문자형	char	'a' , '가' , '1'
정수형	byte, short, int, long	12 , 0 , -3
실수형	float, double	3.14

```java
boolean isEqual = true;
char a = '가';
int age = 30;
long age2 = 32;
double PI = 3.141592;
```

데이터 타입별 메모리 크기

데이터타입	크기
Boolean, byte	1 바이트
char, short	2바이트
int , float	4바이트
long, double	8바이트



바. 상수와 리터럴

앞서 상수를 잠깐 언급하였는데 변수는 선언과동시에 혹은 그 이후 언제든지 값을 변경 할 수 있다.
하지만 값이 한번 정해지면 바뀌지 말아야 할 데이터도 있을 것이다. 가령 원주율 같은것들이다.

그림12는 상수를 만드는 방법이다. 그 외에 우변에 있는 3.14처럼 데이터를 직접적으로 코드내에 적어놓은 형태를 리터럴 이라고 부른다.

어차피 값이 바뀌지 않으면 그냥 리터럴로 3.14 적어놓으면 되지 않는가 하겠지만 상수를 이용함으로써 데이터의 의미를 명확하게 표기하기에 가독성이 좋아지며 혹시나 값이 바뀌게 될 경우 해당 변수에 초기화 부분만 수정을 해주면 되기에 상수를 쓰는 것이 좋다.

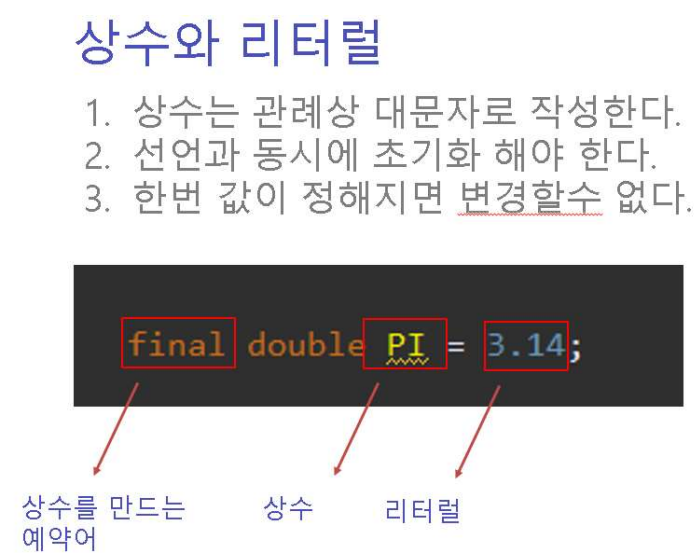
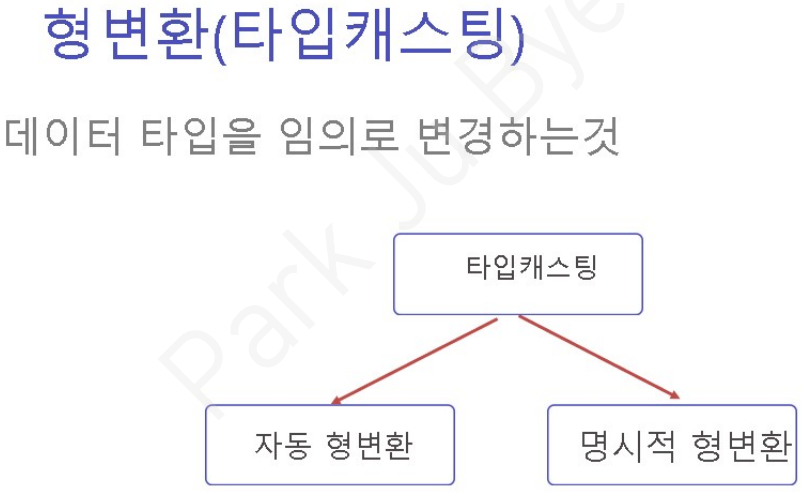


그림 12

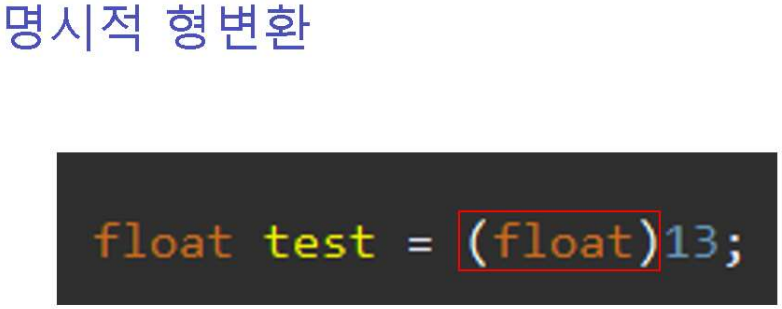
3. 형변환

앞서 배운 데이터타입 int를 char 등으로 변경하는 것을 말한다. JAVA에서 자동으로 형 변환을 해주는 경우가 있고 개발자가 명시적으로 적어주어서 변환해주는 경우가 있다. 자동 형 변환은 특정조건에서만 가능하며 자동형변환이 안된다고 해서 형변환이 불가능한 것은 아니다. 명시적으로 작성하여 변환이 가능하다. 또한 명시적으로 변환이 불가능한 데이터 타입도 있다.



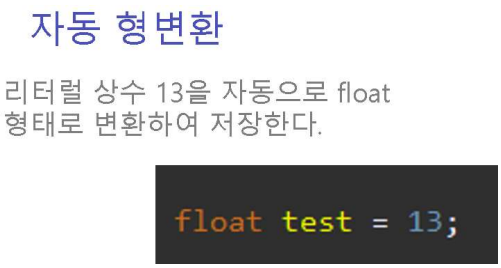
가. 명시적 형변환

다음과 같이 데이터 타입을 괄호를 통해 임의로 변환하는 것이다. 리터럴로 적혀있는 13은 정수이나 float형 변수에 저장하기 위해 명시적으로 float로 변환하고 있다.



나. 자동 형 변환

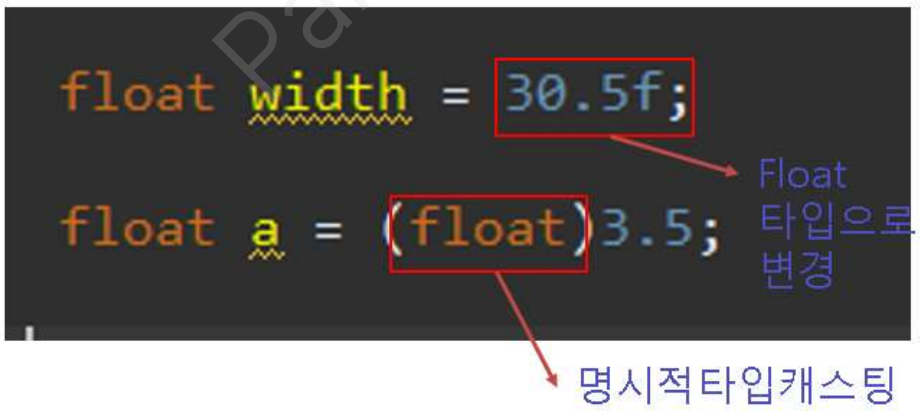
자동 형 변환은 개발자가 따로 작성을 해주지 않아도 자동으로 형 변환이 되어 에러를 일으키지 않는다.
자동 형 변환이 되는 조건은 작은 범위에서 큰 범위로 옮겨질 때 자동으로 변환된다. 생각해보면 당연한 것이다. 작은 물컵에서 큰 컵으로 옮겨 담을시 손실이 없듯이 int형에서 float로 이동될 때 데이터손실이 없기 때문에 바꾼다고 해서 문제가 될 것이 없다. 따라서 컴파일러가 자동으로 변환해주는 것이다. 반대로 float에서 int형으로 변환되면 소수점 부분의 데이터가 손실된다. 그래서 자동으로 해주지 않으며 이때 개발자가 명시적으로 데이터 손실을 감수하고서 변환은 가능하다.



다. 리터럴의 타입

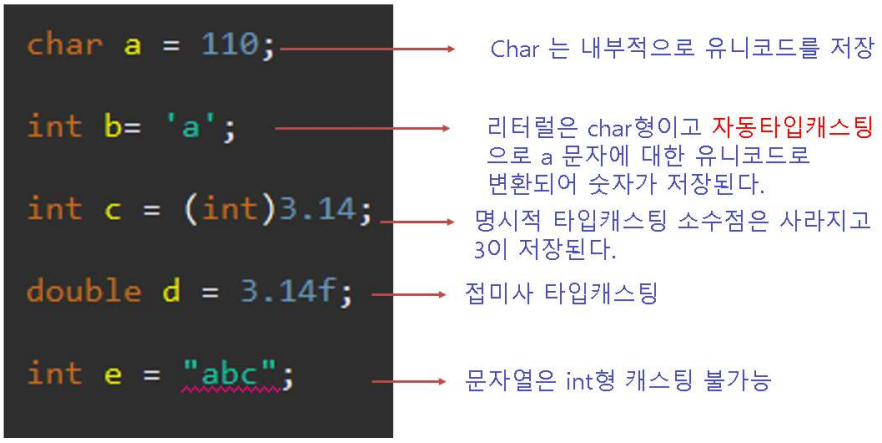
앞서 리터럴은 코드상에 직접 적어놓은 데이터를 말한다고 하였다. 이것역시 임시적으로 메모리에 저장이 되며 타입을 가진다. 아래의 코드가 오류가 발생하는 이유는 JAVA에서 기본적으로 실수를 리터럴로 작성하면 double로 취급하기 때문이다. double은 float보다 큰 범위이기에 자동 형 변환이 불가능하며 따라서 이때는 명시적 형 변환을 하여 저장하여야 한다.

명시적 형변환은 아래와 같이 괄호뿐만 아니라 접미사를 통해서도 가능하다.



다음은 자동 형 변환, 명시적형변환, 혹은 불가능한 타입 불일치 케이스를 다룬 것이다.

타입 불일치



4. 참조형

앞서 배운 기본형 8가지를 제외 하고 모두 참조형이다. C, C++을 배웠다면 참조형은 포인터라고 생각 하면 된다. 물론 JAVA에서는 포인터의 복잡함을 없애고자 포인터를 없애 버렸다. 하지만 내부적으로는 포인터와 똑같이 작동된다.

참조형은 앞으로 배울 클래스 와 객체 등을 다룰 때 쓰이는 타입이다. 현재는 객체를 저장하기 위해선 참조형 변수로 선언되어야 한다는것말 알아두자.