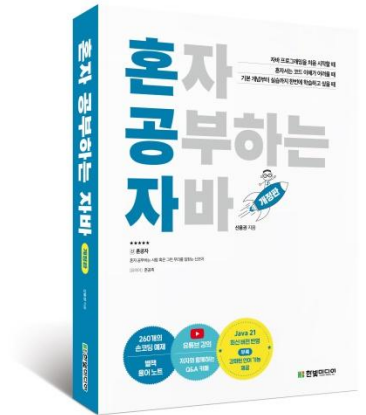


Chapter

# 02

## 변수와 타입



### 02-3. 타입 변환

혼자 공부하는 자바(개정판) (신용권 저)

- 0. 시작하기 전에
- 1. 타입 변환
- 2. 정수 연산에서의 자동 타입 변환
- 3. 실수 연산에서의 자동 타입 변환
- 4. + 연산에서의 문자열 자동 타입 변환
- 5. 문자열을 기본 타입으로 강제 타입 변환
- 6. 키워드로 끝내는 핵심 포인트



## 0. 시작하기 전에

[핵심 키워드] : 자동 타입 변환, 강제 타입 변환, 문자열 결합 연산,  
Integer.parseInt( ), Double.parseDouble( )

### [핵심 포인트]

- 타입 변환이란 데이터 타입을 다른 데이터 타입으로 변환하는 것을 말한다.
  - byte 타입 -> int 타입,
  - int 타입 -> byte 타입,
  - double 타입 -> int 타입
  - String 타입 -> int 타입

### ❖ 타입 변환

- 변수 값을 다른 타입의 변수에 저장할 때 타입 변환이 발생할 수 있다.

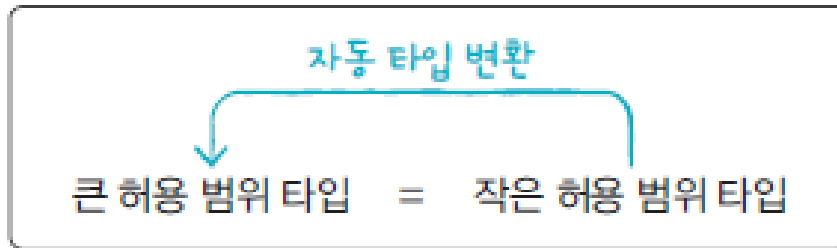
```
byte a = 10;    //byte 타입 변수 a에 10을 저장
int b = a;      //byte 타입 변수 a에 저장된 10을 int 타입 변수 b에 복사해서 저장
```



# 1. 타입 변환

## ❖ 자동 타입 변환 (promotion)

- 값의 허용 범위가 작은 타입이 큰 타입으로 저장될 경우



- 기본 타입의 허용 범위 순

```
byte < short < int < long < float < double
```

```
byte byteValue = 10;  
int intValue = byteValue;    //자동 타입 변환됨
```

```
long longValue = 5000000000L;  
float floatValue = longValue;    //5.0E9f로 저장됨  
double doubleValue = longValue;  //5.0E9로 저장됨
```



# 1. 타입 변환

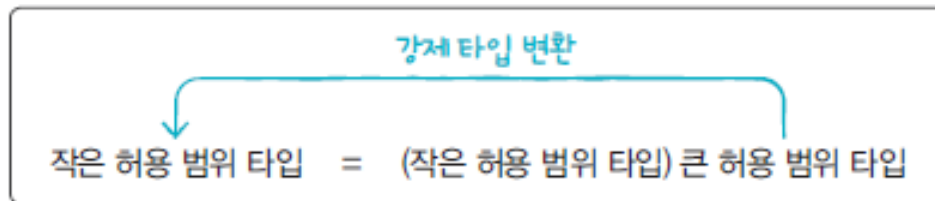
- char 타입의 경우 int 타입으로 자동변환되면 유니코드 값이 int 타입에 저장

```
char charValue = 'A';  
int intValue = charValue; //65가 저장됨
```

```
byte byteValue = 65; byte : -128 ~ 127 , char : ( )  
char charValue = byteValue; ← 컴파일 에러 char - byte 가
```

## ❖ 강제 타입 변환 (casting) ->

- 큰 허용 범위 타입을 작은 허용 범위 타입으로 강제로 나누어 한 조각만 저장



- 캐스팅 연산자 괄호 () 사용: 괄호 안이 나누는 단위

```
int intValue = 10;  
byte byteValue = (byte) intValue; //강제 타입 변환
```



# 1. 타입 변환

## ■ ex) int 타입을 char 타입으로 강제 변환

- 문자 출력 위함

```
int intValue = 65;  
char charValue = (char) intValue;  
System.out.println(charValue);    //"A"가 출력
```

## ■ ex) 실수 타입을 정수 타입으로 강제 변환

- 소수점 이하 부분 버려지고 정수 부분만 저장

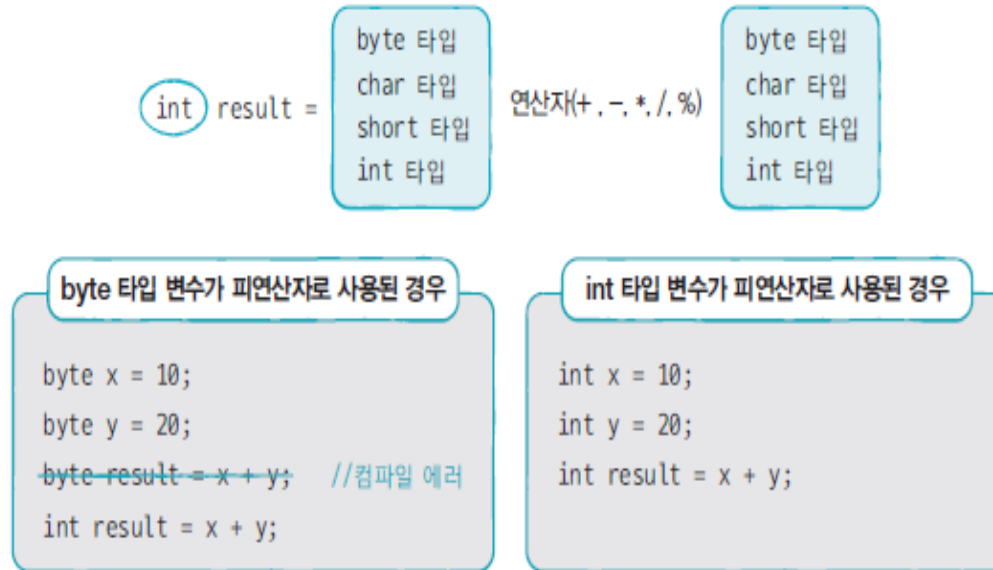
```
double doubleValue = 3.14;  
int intValue = (int) doubleValue;    //intValue는 정수 부분인 3만 저장
```



## 2. 정수 연산에서의 자동 타입 변환

❖ 정수 타입 변수가 산술 연산식에서 피연산자로 사용되는 경우

- byte, char, short 타입 변수는 int 타입으로 자동 변환



- 특별한 경우 아니라면 정수 연산에 사용하는 변수는 int 타입으로 선언하는 것이 효과적
- 피 연산자 중 하나가 long 타입이면 다른 피연산자는 long 타입으로 자동 변환

long result =

long 타입

연산자(+, -, \*, /, %)

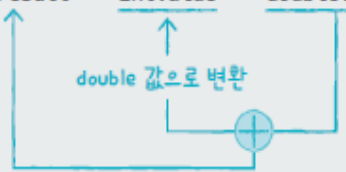
byte 타입  
char 타입  
short 타입  
int 타입



### 3. 실수 연산에서의 자동 타입 변환

#### ❖ 피연산자 중 하나가 double 타입일 경우 다른 피연산자도 double 타입으로 자동 변환

```
int intValue = 10;
double doubleValue = 5.5;
double result = intValue + doubleValue; //result에 15.5가 저장됨
```



#### ↳ 다른 E

```
int intValue = 10;
double doubleValue = 5.5;
int result = intValue + (int) doubleValue; //result에 15가 저장됨
```

#### ■ 실수 리터럴 연산

- `double result = 1.5 + 2.3;`
- `float result = 1.5 + 2.3`
- `float result = 1.5f + 2.3f;`





### 3. 실수 연산에서의 자동 타입 변환

#### ❖ 정수 연산의 결과를 실수로 저장할 때 주의할 점

- 정수 연산의 결과는 정수

```
int x = 1;
int y = 2;
double result = x / y;
System.out.println(result);
```

- 실수 결과를 얻으려면 실수 연산으로의 변환 필요

방법 1	<pre>int x = 1; int y = 2; double result = (double) x / y; System.out.println(result);</pre>
방법 2	<pre>int x = 1; int y = 2; double result = x / (double) y; System.out.println(result);</pre>
방법 3	<pre>int x = 1; int y = 2; double result = (double) x / (double) y; System.out.println(result);</pre>



## 4. + 연산에서의 문자열 자동 타입 변환

### ❖ + 연산

- 피연산자가 모두 숫자일 경우 덧셈 연산
- 피연산자 중 하나가 문자열일 경우 나머지 피연산자도 문자열로 자동 변환되고 문자열 결합 연산

```
int value = 3 + 7;    → int value = 10;  
String str = "3" + 7; → String str = "3" + "7"; → String str = "37";  
String str = 3 + "7"; → String str = "3" + "7"; → String str = "37";
```

- + 연산은 앞에서부터 순차적으로 수행
  - 먼저 스해되 여사이 결합 여사이 곱으 이흐 무드 여사이 결합 여사이 되

```
int value = 1 + 2 + 3;    → int value = 3 + 3;    → int value = 6;  
String str = 1 + 2 + "3"; → String str = 3 + "3"; → String str = "33";  
String str = 1 + "2" + 3; → String str = "12" + 3; → String str = "123";  
String str = "1" + 2 + 3; → String str = "12" + 3; → String str = "123";
```



## 5. 문자열을 기본 타입으로 강제 타입 변환

### ❖ 문자열을 기본 타입으로 강제 변환

변환 타입	사용 예
String → byte	<pre>String str = "10"; byte value = Byte.parseByte(str);</pre>
String → short	<pre>String str = "200"; short value = Short.parseShort(str);</pre>
String → int	<pre>String str = "300000"; int value = Integer.parseInt(str);</pre>
String → long	<pre>String str = "400000000000"; long value = Long.parseLong(str);</pre>
String → float	<pre>String str = "12.345"; float value = Float.parseFloat(str);</pre>
String → double	<pre>String str = "12.345"; double value = Double.parseDouble(str);</pre>
String → boolean	<pre>String str = "true"; boolean value = Boolean.parseBoolean(str);</pre>



## 5. 문자열을 기본 타입으로 강제 타입 변환

- 문자열이 숫자 외 요소를 포함할 경우 숫자 타입 변환 시도할 경우 숫자 형식 예외 발생

```
String str = "1a";  
int value = Integer.parseInt(str);    //NumberFormatException 발생
```

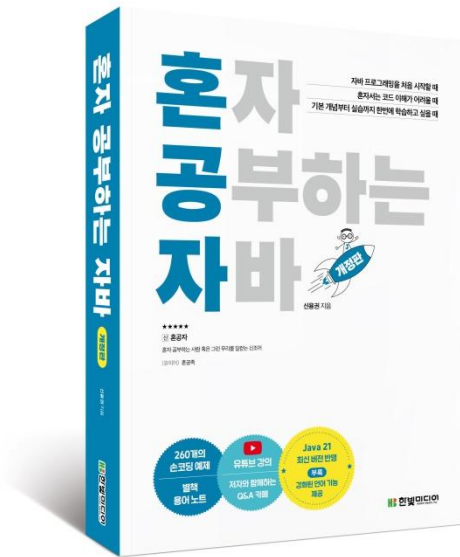
- `String.valueOf()` 메소드 사용하여 기본 타입을 문자열로 변환
  - `String str = String.valueOf(3);`



## 6. 키워드로 끝내는 핵심 포인트

- **자동 타입 변환**: 자동으로 타입이 변환되는 것. 값의 허용 범위가 작은 타입이 허용 범위가 큰 타입으로 저장될 때 발생함.
- **강제 타입 변환**: 강제로 타입을 변환하는 것. 값의 허용 범위가 큰 타입을 허용 범위가 작은 타입으로 쪼개어서 저장하는 것을 말함.
- **문자열 결합 연산**: 문자열과 + 연산을 하면 다른 피연산자도 문자열로 변환되어 문자열 결합이 발생함.
- **Integer.parseInt()** : 문자열을 정수 int 타입으로 변환
- **Double.parseDouble()** : 문자열을 실수 double 타입으로 변환





Thank You!