JAVA Generic Method

프로그램은 사람이 이해하는 코드를 작성.

느려도 꾸준하면 경기에서 이긴다.

작성자 : 홍효상
이메일 : hyomee@naver.com
소스 : https://github.com/hyomee/JAVA_EDU

Content

6. Generic Method

1. Generic 이란

1. Generic 이란

" Data type을 특정한 type하나로 정하지 않고 사용할 때마다 바뀔 수 있게 범용적이고 포괄적으로 지정 "

Generic

- 포괄적인, 총칭의, 회사 이름이 붙지 않은, 일반 명칭으로 판매되는 사전적 의미로, 딱 하나를 정하지 않고 범용적이고 포괄적이라는 의미
- Data type을 특정한 type하나로 정하지 않고 사용할 때마다 바뀔 수 있게 범용적이고 포괄적으로 지정한다 라는 의미
- Object Class의 한계를 극복
- **호출되는 시점에 실제 Generic 타입을 지정**

01. 왜 Generic가 만들어 졌을까 ?

```java
public class GenericMethodMain {
    public static void main(String... args) {

        List list = new ArrayList<>();
        list.add("홍길동");
        list.add("홍당무");
        list.add("홍사과");

        for (int i = 0; i < list.size(); i++) {
            String str = list.get(i);
            System.out.println(str);
        }
    }
}
```

java: incompatible types: java.lang.Object
cannot be converted to java.lang.String

```java
public class GenericMethodMain {
    public static void main(String... args) {

        List list = new ArrayList<>();
        list.add("홍길동");
        list.add("홍당무");
        list.add("홍사과");

        for (int i = 0; i < list.size(); i++) {
            String str = (String) list.get(i);
            System.out.println(str);
        }
    }
}
```

Type Casting 이 빈번하게 일어나서 성능 저하 발생
- 수만개의 Data가 있다면

```java
public class GenericMethodMain {
    public static void main(String... args) {

        List<String> list = new ArrayList<>();
        list.add("홍길동");
        list.add("홍당무");
        list.add("홍사과");

        for (int i = 0; i < list.size(); i++) {
            String str = list.get(i);
            System.out.println(str);
        }
    }
}
```

Bast Code

```java
for (String str: list) {
    System.out.println(str);
}
```

2. Generic Method

Generic Method

- Method 만들 때 파라메터와 반환 값의 자료형을 Generic로 선언 한 Method
- 입력 매개 변수 값으로 Generic Type을 유추 할 수 있을 경우는 생략 가능
- Generic Method 내부에서는 매개변수로 한 참조 변수의 메서드로 Object Class의 Method만 가능

➢ 관례적 표기와 의미

Generic Type	의미
T	타입 (Type)
K	키 (Key)
V	값 (Value)
N	숫자 (Number)
E	원소 (Element)

➢ 문법 구조

입력 매개변수의 타입의 개수 만큼

선언 : 접근지정자 < T [, ..] > T 메서드명(T t [, V v, ...]);

호출 : 참조객체.<실제 제네릭 타입>메서드명(입력매개변수)

입력 매개변수의 실제 타입을 의미함

```
GenericMethod genericMethod = new GenericMethod();
int num01 =   genericMethod.<Integer> method(10);
System.out.println(num01);

// 입력 매개 변수가 Genetic Type가 유추 할 수 있는 경우 생략 가능
int num02 =   genericMethod.method(10);
System.out.println(num02);

String str02 = genericMethod.methodRnString("안녕" , 10);
System.out.println(str02);

genericMethod.methodPrint("name" , "홍길동");
```

```
10
I0
안녕10
name : 홍길동
```

```
<T> T method(T t) {
  return t;
}

// Generic Method 내부에서 매개변수의 산술 연산 되지 않음 ->  Object
<T> String methodRnString(T t1, T t2){
  // t1 +  t2 : Operator '+' cannot be applied to 'T', 'T'
  return t1 + "" + t2;
}

<K,V> void methodPrint(K t1, V v1){
  System.out.println(t1 + " : " + v1 );
}
```

2. Generic Method

Generic Method -

- extends : 매개 변수에 특정 타입만 받게 제한 할 때 사용
- 여러 개 사용시 임의로 타입 설정

```
String str03 =  genericMethod.methodPrintRnString("name" , "홍길동");
System.out.println(str03);

System.out.println(genericMethod.checkEquals(10 , 2.1));
System.out.println(genericMethod.checkEquals("사과" , "배"));
System.out.println(genericMethod.checkEquals("사과" , "사과"));

System.out.println(genericMethod.<String, Integer, Object>extednsMethod("사과" , 0, "첫문자 : "));
System.out.println(genericMethod.extednsMethod("사과" , 0, "첫문자 : "));
```

첫번째 파라메터 : 문자
두번째 매개변수 : 숫자
세번째 파라메터 : Object

```
<K,V> void methodPrint(K t1, V v1){
  System.out.println(t1 + " : " + v1 );
}

<K,V> V methodPrintRnString(K t1, V v1){
  // t1 +  t2 : Operator '+' cannot be applied to 'T', 'T'
  System.out.println(t1 + " : " + v1 );
  return (V) (t1 + " : " + v1);
}


<T, V> boolean checkEquals(V v1, V v2){
  return v1.equals(v2);
}
// Generic Method 내부에서는 매개변수로 한 참조 변수의 메서드로
// Object Class의 Method만 가능
<T extends String, I extends Integer, V> String extednsMethod(T t, I i, V v){
  char ch = t.charAt(i);
  return  v.toString() + ch ;
}
```