

제네릭 프로그래밍

제네릭 타입

■ 필요성

- 자바는 다양한 종류의 객체를 관리하는 컬렉션이라는 자료구조를 제공
- 초기에는 Object 타입의 컬렉션을 사용
- Object 타입의 컬렉션은 실행하기 전에는 어떤 객체인지?



- 예제(Object 타입)
 - [sec03/Beverage](#), [sec03/Beer](#), [sec03/Boricha](#), [sec03/object/Cup](#)
 - [sec03/GenericClass1Demo](#)



제네릭 타입

■ 소개

● 제네릭 타입의 의미

- 하나의 코드를 다양한 타입의 객체에 재사용하는 객체 지향 기법
- 클래스, 인터페이스, 메서드를 정의할 때 타입을 변수로 사용



● 제네릭 타입의 장점

- 컴파일할 때 타입을 점검하기 때문에 실행 도중 발생할 오류 사전 방지
- 불필요한 타입 변환이 없어 프로그램 성능 향상

제네릭 타입

■ 제네릭 타입 선언

```
class 클래스이름<타입매개변수> {  
    필드;  
    메서드;  
}
```

메서드나 필드에 필요한 타입을 타입 매개변수로 나타낸다.

- 타입 매개변수는 객체를 생성할 때 구체적인 타입으로 대체
- 전형적인 타입 매개변수

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

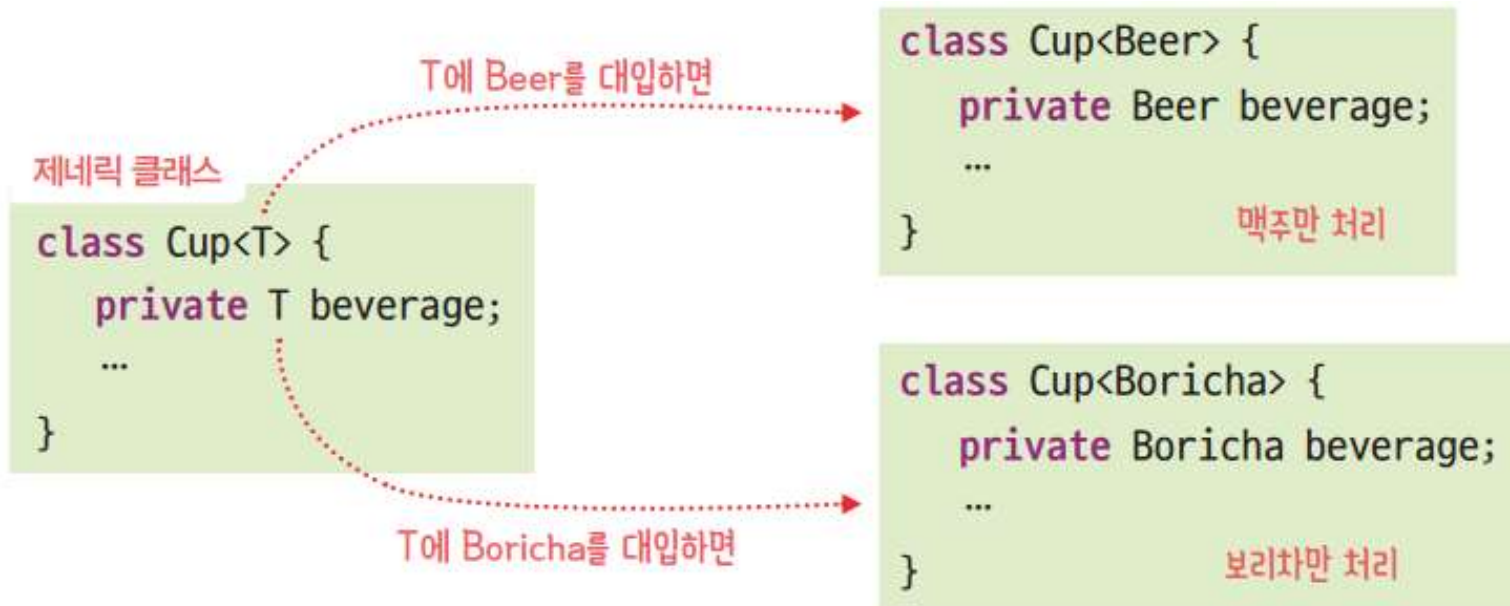
제네릭 타입

■ 제네릭 객체 생성

제네릭클래스 <적용할타입> 변수 = new 제네릭클래스<적용할타입>();

생략할 수 있다.

- <적용할타입>에서 적용할 타입을 생략할 경우 <>를 다이아몬드 연산자라고 함
- 제네릭 클래스의 적용



제네릭 타입

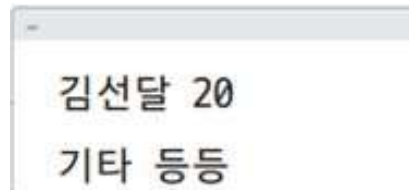
■ 제네릭 타입 응용

- 예제 : [sec03/generic/Cup](#), [sec03/GenericClass2Demo](#)



- 예제(2개 이상의 타입 매개변수)

- [sec03/Entry.java](#)
- [sec03/EntryDemo](#)



■ Raw 타입의 필요성 및 의미

- 이전 버전과 호환성을 유지하려고 Raw 타입을 지원
- 제네릭 클래스를 Raw 타입으로 사용하면 타입 매개변수를 쓰지 않기 때문에 Object 타입이 적용
- 예제 : [sec03/GenericClass3Demo](#)

제네릭 상속 및 타입 한정

■ 제네릭 타입의 상속 관계

- 예를 들어

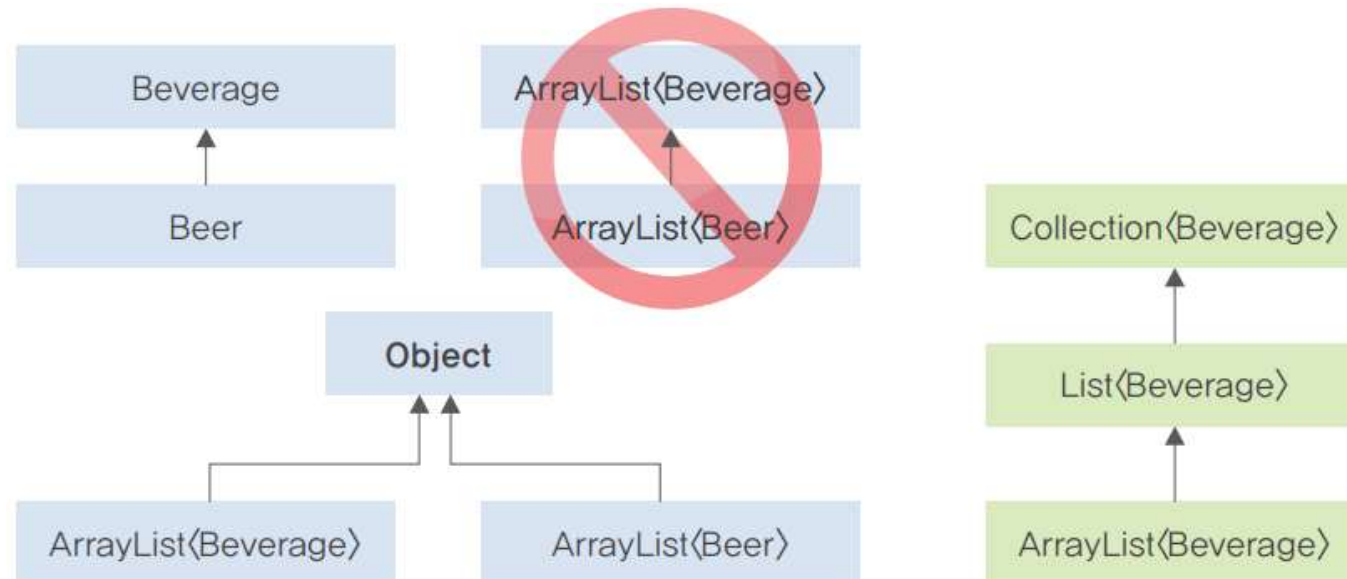
```
ArrayList<Beverage> list = new ArrayList<>();  
list.add(new Beer());           // OK  
list.add(new Boricha());        // OK
```

- 그러나 ArrayList<Beverage> 타입과 ArrayList<Beer>의 경우는 상속 관계가 없다.
- 예제 : [sec04/GenericInheritanceDemo](#)

제네릭 상속 및 타입 한정

■ 제네릭의 제약

- 기초 타입을 제네릭 인수로 사용 불가
- 정적 제네릭 타입 금지
- 제네릭 타입의 인스턴스화 금지. 즉, `new T()` 등 금지
- 제네릭 타입의 배열 생성 금지
- 실행 중에 제네릭 타입 점검 금지. 예를 들어, `a instanceof ArrayList<String>`
- 제네릭 클래스의 객체는 예외로 던지거나 잡을 수 없다
- 제네릭의 서브 타입 허용 않음



제네릭 상속 및 타입 한정

■ 타입 한정

```
<T extends 특정클래스> 반환타입 메서드이름(...) { ... }  
<T extends 인터페이스> 반환타입 메서드이름(...) { ... }
```

부모가 인터페이스라도 extends를 사용한다.

- 예제 : [sec04/bound/BoundedTypeDemo](#)

제네릭 메서드

■ 의미와 선언 방법

- 타입 매개변수를 사용하는 메서드
- 제네릭 클래스뿐만 아니라 일반 클래스의 멤버도 될 수 있음
- 제네릭 메서드를 정의할 때는 타입 매개변수를 반환 타입 앞에 위치

```
<타입매개변수> 반환타입 메서드이름(...) {  
    ...  
}
```

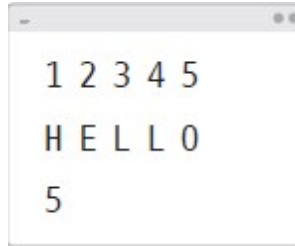
2개 이상의 타입 매개변수도 가능하다.

- 제네릭 메서드를 호출할 때는 구체적인 타입 생략 가능
- JDK 7과 JDK 8의 경우 익명 내부 클래스에서는 다이아몬드 연산자 사용 불허
- JDK 9부터는 익명 내부 클래스에서도 다이아몬드 연산자 사용 가능

제네릭 메서드

■ 예제

- 배열의 타입에 상관없이 모든 원소 출력
- [sec05/GenMethod1Demo](#)



```
1 2 3 4 5  
H E L L O  
5
```

제네릭 메서드

■ 제네릭 타입에 대한 범위 제한

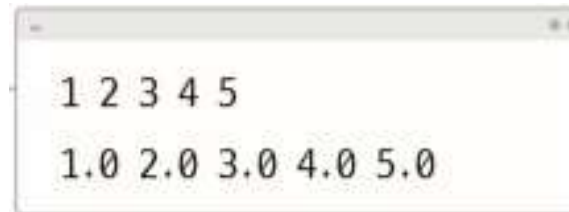
● 사용 방법

```
<T extends 특정클래스> 반환타입 메서드이름(...) { ... }  
<T extends 인터페이스> 반환타입 메서드이름(...) { ... }
```

부모가 인터페이스라도 extends를 사용한다.

● 예제

- [sec05/GenMethod2Demo](#)



```
1 2 3 4 5  
1.0 2.0 3.0 4.0 5.0
```

- [sec05/GenMethod3Demo](#)



```
3
```