

JAVA Programming

제네릭

제네릭

- 제네릭(Generic)이란?

데이터의 타입을 일반화 한다는 것을 의미

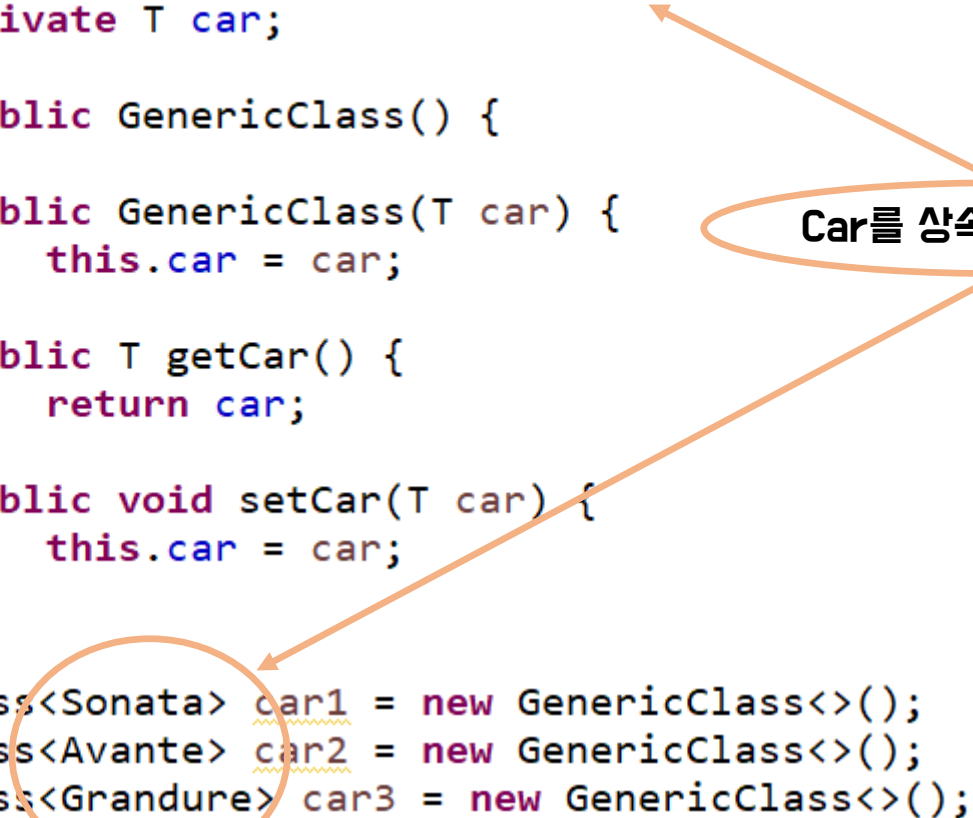
- 제네릭을 활용하면 **타입 변환 및 타입 검사에 들어가는 코드 생략이 가능**

```
public class GenericClass<T> {  
    private T car;  
  
    public GenericClass() {  
    }  
    public GenericClass(T car) {  
        this.car = car;  
    }  
    public T getCar() {  
        return car;  
    }  
    public void setCar(T car) {  
        this.car = car;  
    }  
}
```

제네릭

- 제네릭 클래스에 extends 키워드를 사용해 타입 제한 가능

```
public class GenericClass<T extends Car> {  
    private T car;  
  
    public GenericClass() {  
    }  
    public GenericClass(T car) {  
        this.car = car;  
    }  
    public T getCar() {  
        return car;  
    }  
    public void setCar(T car) {  
        this.car = car;  
    }  
}  
  
GenericClass<Sonata> car1 = new GenericClass<>();  
GenericClass<Avante> car2 = new GenericClass<>();  
GenericClass<Grandure> car3 = new GenericClass<>();
```



와일드카드

- 제네릭 클래스 타입의 객체를 메소드의 매개변수로 받을 때 그 객체의 타입을 제한 가능
 - 〈?〉 : 제한 없음
 - 〈? Extends Type〉 : 와일드카드의 상한 제한 (Type과 Type의 후손을 이용해 생성한 객체만 매개변수로 사용 가능)
 - 〈? super Type〉 : 와일드카드 하한 제한 (Type과 Type의 부모를 이용해 생성한 객체만 매개변수로 사용 가능)

ex) `GenericClass<? extends NewAvante> generic`
//NewAvante이거나 그 후손 타입으로 만들어진 자동차만 매개변수로 사용 가능

`GenericClass<? super NewAvante> generic`
//NewAvante이거나 그 부모 타입으로 만들어진 자동차만 매개변수로 사용 가능

학습점검

- ✓ 제네릭에 대해 이해할 수 있다.
- ✓ 제네릭의 목적에 대해 이해할 수 있다.
- ✓ 제네릭 클래스에 대해 이해할 수 있다.
- ✓ 제네릭 클래스를 적용하여 사용할 수 있다.
- ✓ 와일드 카드에 대해 이해할 수 있다.
- ✓ 와일드 카드를 적용하여 사용할 수 있다.