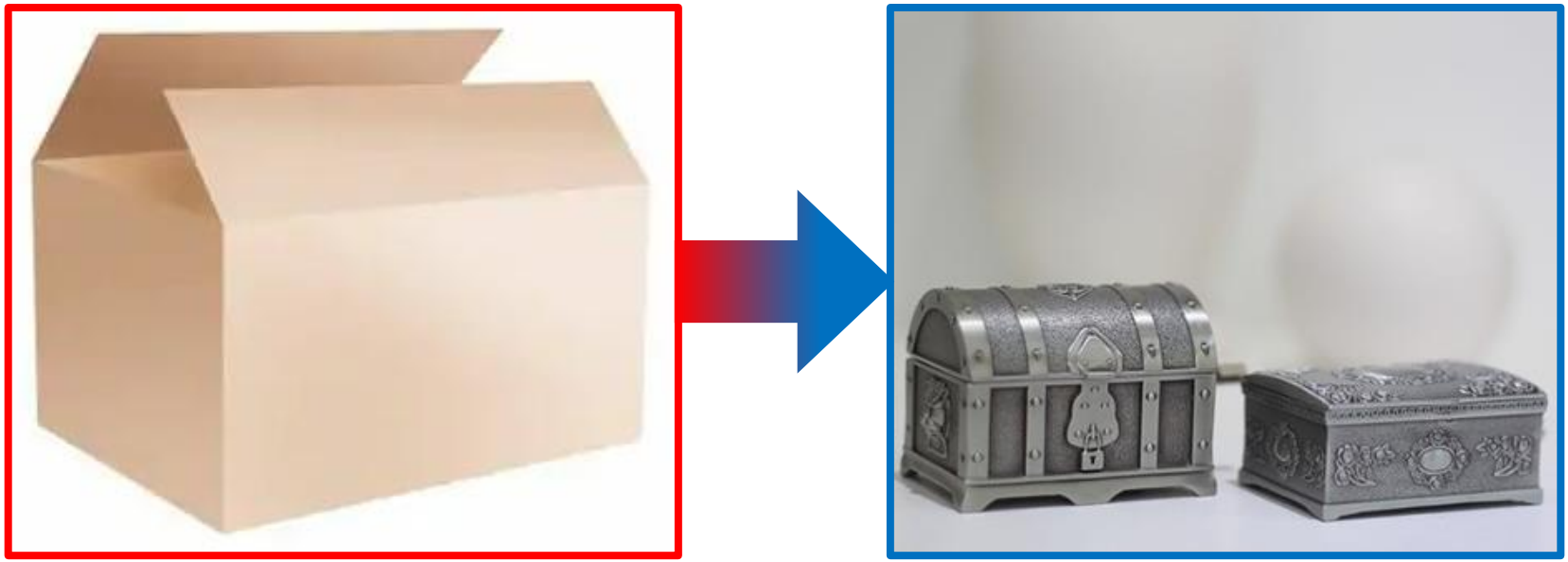

K G 아 이 티 뱅 크
JAVA
J A V A

상속

상속

- **상자는 다양한 것을 담을 수 있는 상자가 될 수 있음**
  - "상자" 라면 종류는 다양하지만 기본구성은 비슷함
  - "보물상자" 와 "음식상자" 는 다른 것을 보관하게 됨
- **상속은 개념(클래스)을 이어받아 확장시키는 기술**
  - "상자" 라는 개념에서 "보물상자" 로 개념으로 확장
  - "상자" 라는 개념을 "오두막" 이라는 개념으로 확장



상속

❖ 상속시 주의사항 : 비슷한 개념으로 확장해야 함

❖ 클래스의 상속

```
public class Super {
    public int n1;
    protected int n2;
    public Super( .. ) { .. }
    public void method1( .. ) { .. }
}
public class Sub extends Super {
    private int n3;
    public Sub( .. ) {
        super(); n3 = ..;
    }
    public void method2( .. ) { .. }
}
```

❖ 부모클래스

포괄적인 개념
필드는 private 대신 protected를 설정함

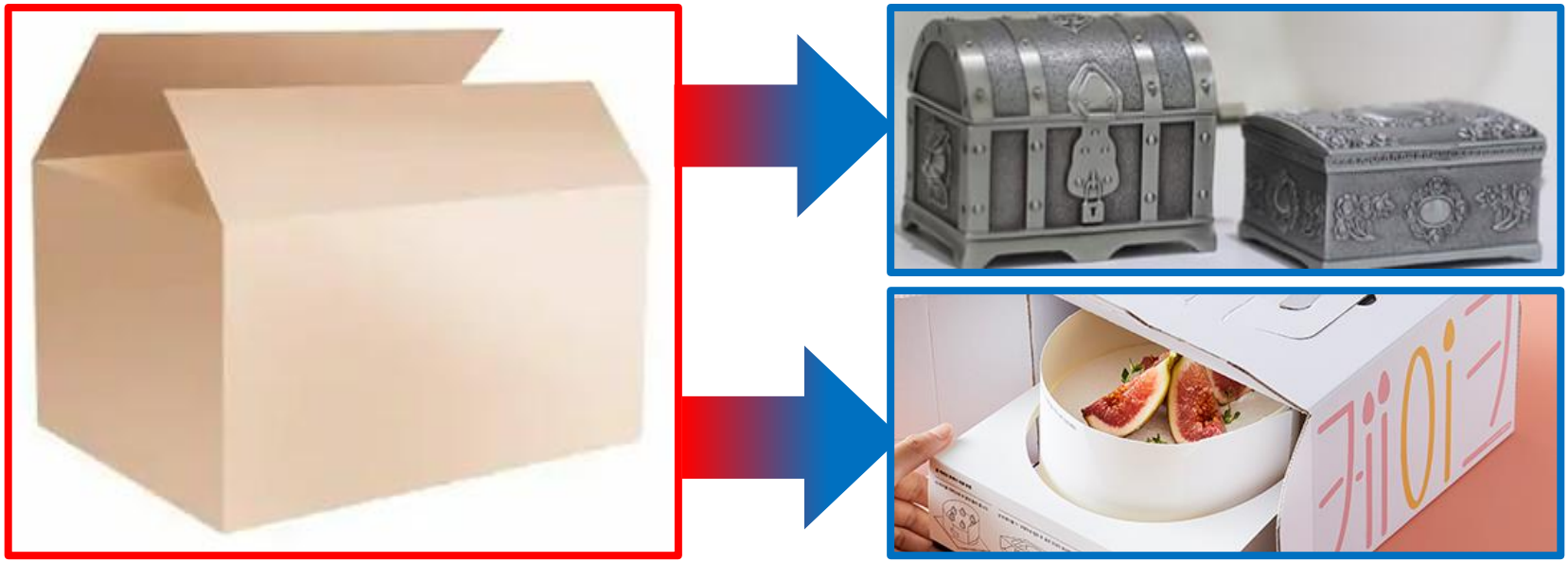
❖ 자식클래스

확장된 개념
super()와 super가 사용됨

제네릭 클래스

제네릭 클래스

- **상속을 통해서 다양한 상자를 만들지만 전부 불가능**
  - 상자에 담을 수 있는 물건의 종류가 너무 많음
  - 이름으로 구별하면 뭘 담은 상자인지 알 수 있음
- **제네릭 클래스 : 자료형에 자유롭게 대응하는 기술**
  - 같은 코드를 자료형만 달리할 경우 적용하게 됨



제네릭 클래스

❖ 제네릭 클래스는 동작이 동일한 것만 처리할 수 있음

❖ 제네릭 클래스의 정의 과정

```
public class Sample<Type> {
    private Type n1;
    private int n2;
    public Sample( Type n1, .. ) { .. }
    public Type method( .. ) { .. }
}
public class Sub<Type> extends Super {
    private Type n3;
    public Sub( Type n3 ) {
        super( ... ); this.n3 = n3;
    }
    public Type method2( .. ) { .. }
}
```

❖ 자료형에 적용됨

자료형이 연결되는 모든 곳에 **꼼꼼하게 적용해야 함**

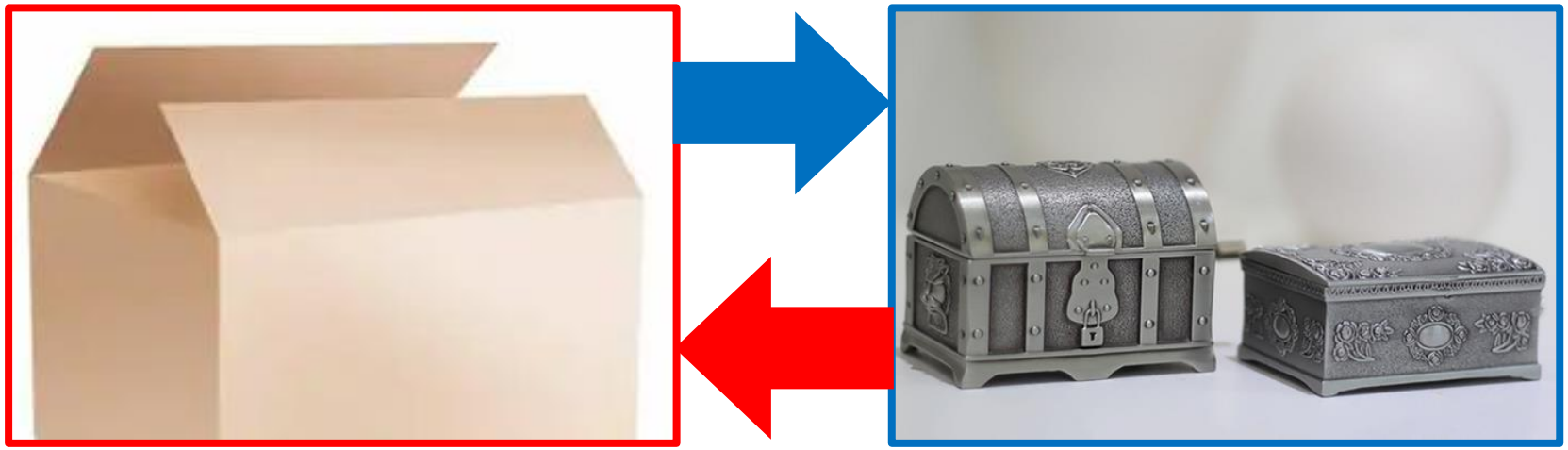
❖ 상속에도 가능

자식클래스와 연관된 곳에만 **꼼꼼하게 적용해야 함**

업캐스팅 / 다운캐스팅

업캐스팅 / 다운캐스팅

- ❖ **상자를 기반으로 하여 다양한 형태로 확장을 수행**
  - ➢ "상자" 를 기반으로 다양한 종류로 활용하게 됨
  - ➢ 만들어진 것들은 모두 서로 다른 객체로 취급
- ❖ **서로 다른 객체가 많아지면 코드의 관리가 어려움**
  - ➢ 객체가 많다 -> 변수가 많다 -> 관리가 어렵다
  - ➢ 이를 통합관리하기 위한 수단/방법이 필요하게 됨
  - ➢ **"보물상자" -> 보물을 다루는 "상자" = 업캐스팅**
  - ➢ **보물을 다루는 "상자" -> "보물상자" = 다운캐스팅**



업캐스팅 / 다운캐스팅

❖ 캐스팅시 주의사항 : 원본이 업되었다가 다운되는 과정

❖ 클래스의 상속

```
Sub sub1 = new Sub();
Super super1 = sub1;
Super super2 = new Sub();
super1.method1(); // 부모메서드만
super2.method2(); // 사용가능

sub1 = (Sub)super1;
Sub sub2 = (Sub)super2;
sub1.method3();  // 자식의 메서드를
sub2.method4();  // 쓸 수 있게 됨
```

❖ 업캐스팅

상속관계만 가능
상속관계가 아니면 불가능하니 주의

❖ 다운캐스팅

부모/자식관계만 가능
다른 자식으로는 다운캐스팅이 안됨