추상 클래스

- 추상 클래스(abstract class): 몸체가 구현되지 않은 메소드를 가지고 있는 클래스
- 추상 클래스는 추상적인 개념을 표현하는데 적당하다.

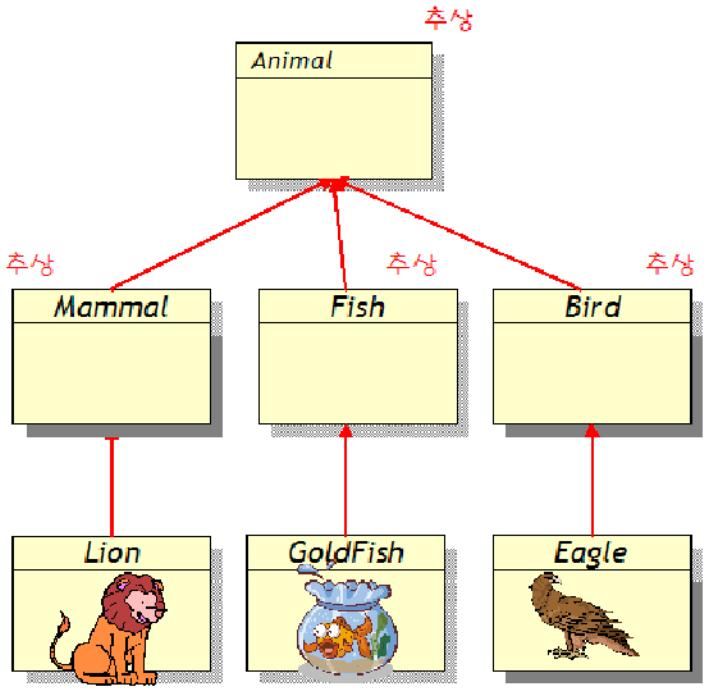


그림 12.1 추상 클래스의 개념

추상 클래스의 예

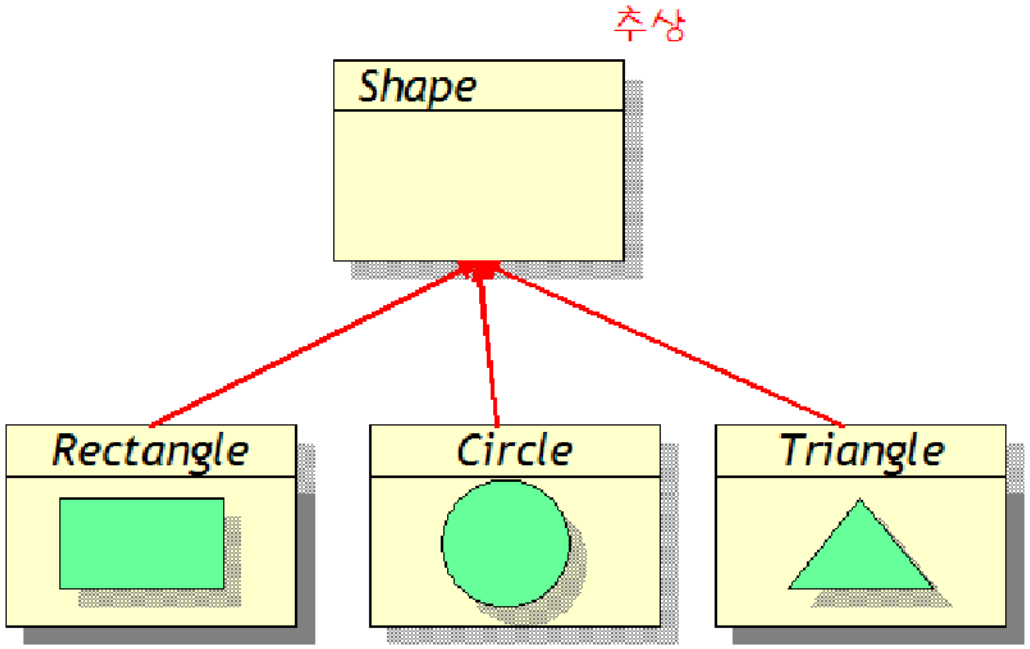


그림 12.2 도형을 나타내는 상속 계층도

추상 클래스의 예

```
abstract class Shape {
        int x, y;
        public void move(int x, int y)  {
                this.x = x;
                this.y = y;
    }
    public abstract void draw();
};
```

```
public class Rectangle extends Shape {
        int width, height;
        public void draw() {                    // 추상 메소드 구현
                System.out.println("사각형 그리기 메소드");
        }
};
```

인터페이스

- 인터페이스(interface): 추상 메소드들로만 이루어진다.

```
public interface 인터페이스_이름 {
        반환형    추상메소드1(...);
        반환형    추상메소드2(...);
        ...
}
```

```
public class 클래스_이름 implements 인터페이스_이름 {
        반환형    추상메소드1(...) {
                  .....
        }
        반환형    추상메소드2(...) {
                  .....
        }
}
```

인터페이스

- 인터페이스는 객체와 객체 사이의 상호 작용을 나타낸다.

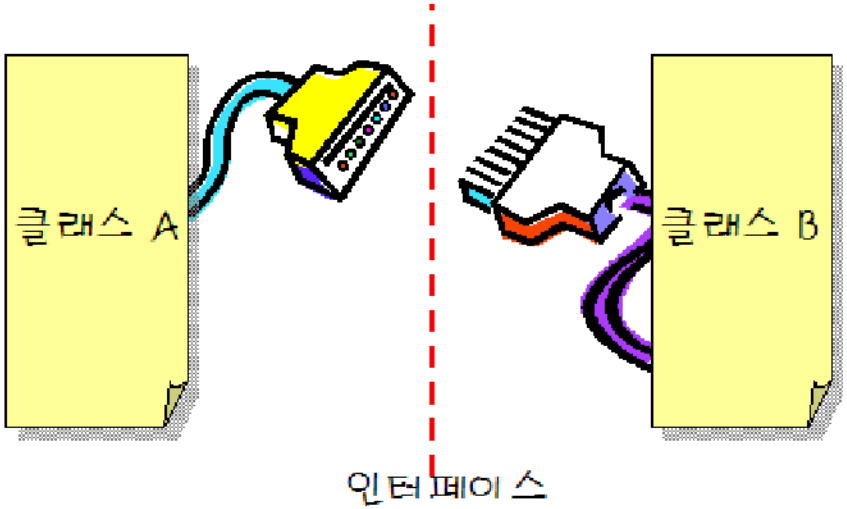


그림 12.3 인터페이스는 클래스 간의 상호 작용을 나타낸다.

인터페이스의 예

- 홈 네트워킹 예제

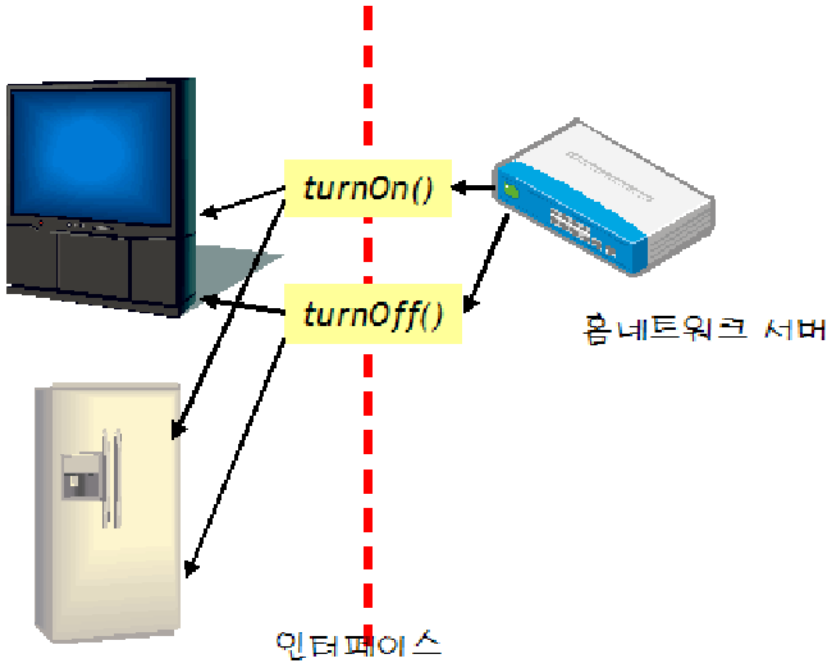


그림 12.4 홈네트워크 예제

홈네트워킹 예제

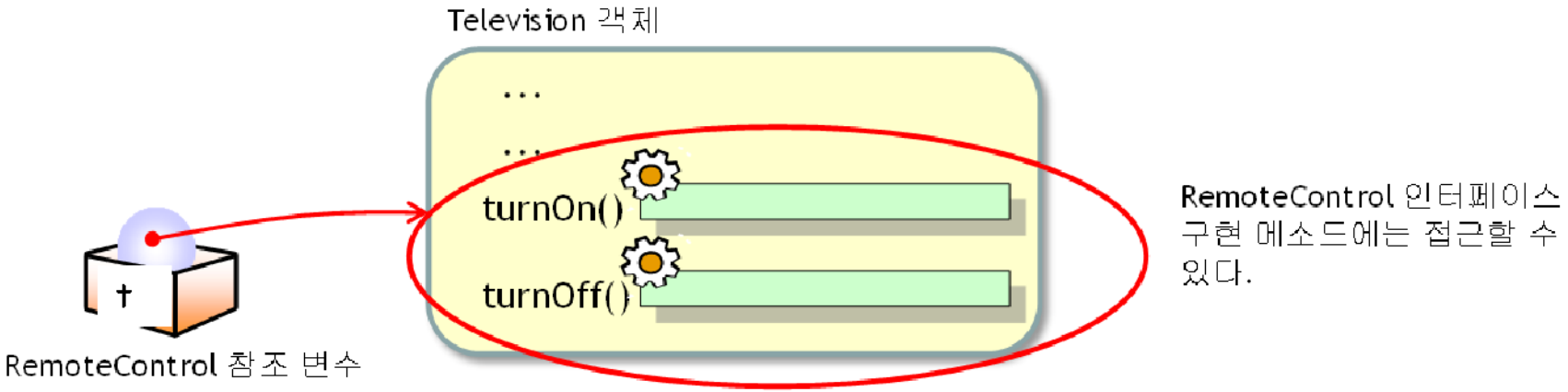
```
public interface RemoteControl {
        public void turnOn();          // 가전 제품을 켠다.
        public void turnOff();         // 가전 제품을 끈다.
}
```

인터페이스를 구현

```
public class Television implements RemoteControl {
        public void turnOn()
        {
        // 실제로 TV의 전원을 켜기 위한 코드가 들어 간다.
        }
        public void turnOff()
        {
        // 실제로 TV의 전원을 끄기 위한 코드가 들어 간다.
        }
}
```

홈네트워킹 예제

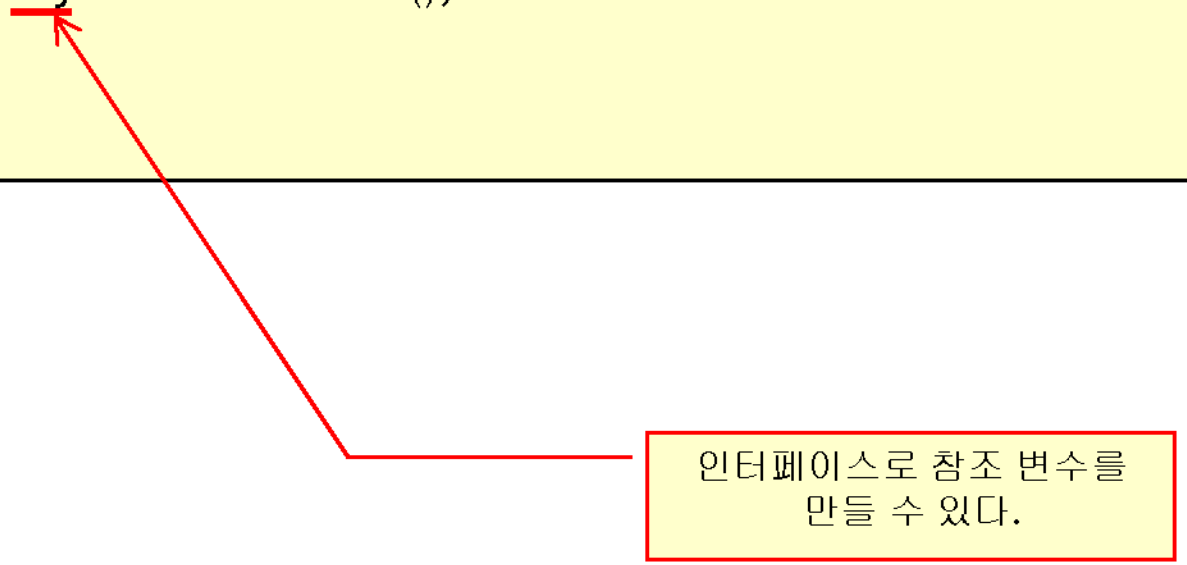
```
Television t = new Television();
t.turnOn();
t.turnOff();
```



인터페이스와 타입

- 인터페이스는 하나의 타입으로 간주된다.

```
RemoteControl obj = new Television();
obj.turnOn();
obj.turnOff();
```



예제

```
public interface Comparable {
    // 이 객체가 다른 객체보다 크면 1, 같으면 0, 작으면 -1을 반환한다.
    int compareTo(Object other);
}
```

```
public class Box implements Comparable
    private double volume = 0;
    public Box(double v) {
        volume = v;
    }
    public int compareTo(Object otherObject) {
        Box other = (Box) otherObject;
        if (this.volume < other.volume)  return -1;
        else if (this.volume > other.volume) return 1;
        else return 0;
    }
```

예제

```
        public static void main(String[] args) {
                Box b1 = new Box(100);
                Box b2 = new Box(85.0);
                if (b1.compareTo(b2) > 0)
                    System.out.println("b1이 b2보다 더 크다");
                else
                    System.out.println("b1가 b2와 같거나 작다“);
        }
}
```

여러 인터페이스를 동시에 구현

```
public class Television implements RemoteControl, SerialCommunication
{
        ...
        public void turnON()          {       ...       }
        public void turnOFF()         {       ...       }
        public void send(byte[] data) {                 ...       }
        public byte[] receive()       {       ...       }
}
```

인터페이스 상속하기

- 인터페이스가 인터페이스를 상속받는 것도 가능하다.

```
public interface AdvancedRemoteControl extends RemoteControl {
        public void volumeUp();        // 가전제품의 볼륨을 높인다.
        public void volumeDown();// 가전제품의 볼륨을 낮춘다.
}
```

다중 상속

- 다중 상속이란 여러 개의 수퍼 클래스로부터 상속하는 것
- 자바에서는 다중 상속을 지원하지 않는다.
- 다중 상속에는 어려운 문제가 발생한다.

```
class SuperA { int x; }
class SuperB { int x; }
class Sub extends SuperA, SuperB     // 만약에 다중 상속이 허용된다면
{
...
}
Sub obj = new Sub();
obj.x = 10;                          // obj.x는 어떤 수퍼 클래스의 x를 참조하는가?
```

다중 상속

- 인터페이스를 이용하면 다중 상속의 효과를 낼 수 있다.

```
class Shape {
        protected int x, y;
}
interface Drawable {
        void draw();
};
public class Rectangle extends Shape implements Drawable {
        int width, height;
        public void draw() {
                System.out.println("Rectangle Draw");
        }
};
```