
4,000번의 도전
드디어 찾아낸 반응 단백질
2012 인텔 국제과학경진대회 고든무어상 수상
(Intel ISEF 2012, Gordon E. Moore Award)
Java 프로그래밍
수업내용 정리
및 실습 과제
책을 보거나 인터넷을 찾아보면서 배우고 거기에 저만의 방식을 추가했어요.
TIME
JACK ANDRAKA
IS CREATING A REVOLUTION
AN ANTONIO SPURS
처음에는 막막한 느낌이지만 시간이 지나면 감이 오거든요.
더하기 빼기... 그게 거의 전부죠.
저는 여러분에게 컴퓨터를 통해 어떻게
세상을 바꿀 수 있는 지 얘기하고 싶습니다
그러나 사실, 구글이나 위키피디아 등을 이용해
알고 싶은 모든 것을 찾을 수 있고
ANESSA
D GIRL DEVELOP IT
처음 배울때 겁이 안나는게 뭐가 있을까요?
프로그래밍은 천재들만 할 수 있는것이 아니에요. 의지력이 필요할 뿐입니다.
15세 소년의 유일한 발명 도구는
인터넷(internet)
"더 좋은 진단 키트를 만들고 싶었다."
– 잭 안드라카
GOOGLE, WIKIPEDIA
AND STACKOVERFLOW
2020.04.21.화
B반 송명훈

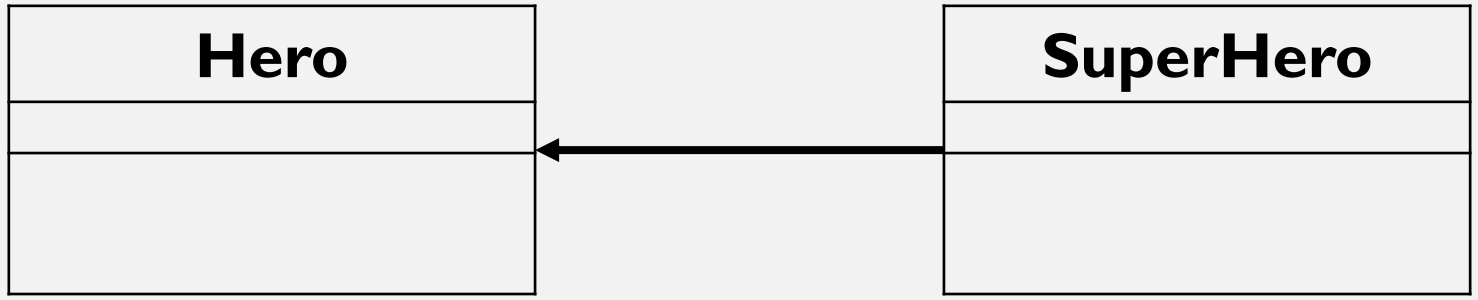
목 차

1. 수업 내용 정리

1. 수업 내용 정리 [Inheritance]

상속 (Inheritance) : 기존의 클래스에 기능을 추가하거나 재정의하여 새로운 클래스를 정의하는 것

1. 말 그대로 자식이 부모로부터 무언가를 물려받는 것, 캡슐화, 추상화와 더불어 객체 지향 프로그래밍을 구성하는 중요한 특징 중 하나
    1. "이전에 만든 클래스와 닮았지만, 일부 다른 클래스"를 만들 필요가 있을 경우 이용
    2. 기존에 정의되어 있는 클래스의 모든 필드와 메소드를 물려받아, 새로운 클래스를 생성
2. 클래스 상속을 위해서는 **extends** 라는 키워드 사용
   Ex] 자식클래스(child class=sub class=derived class) extends 부모클래스(parent class=super class=base class)
3. 상속관계의 표현방법

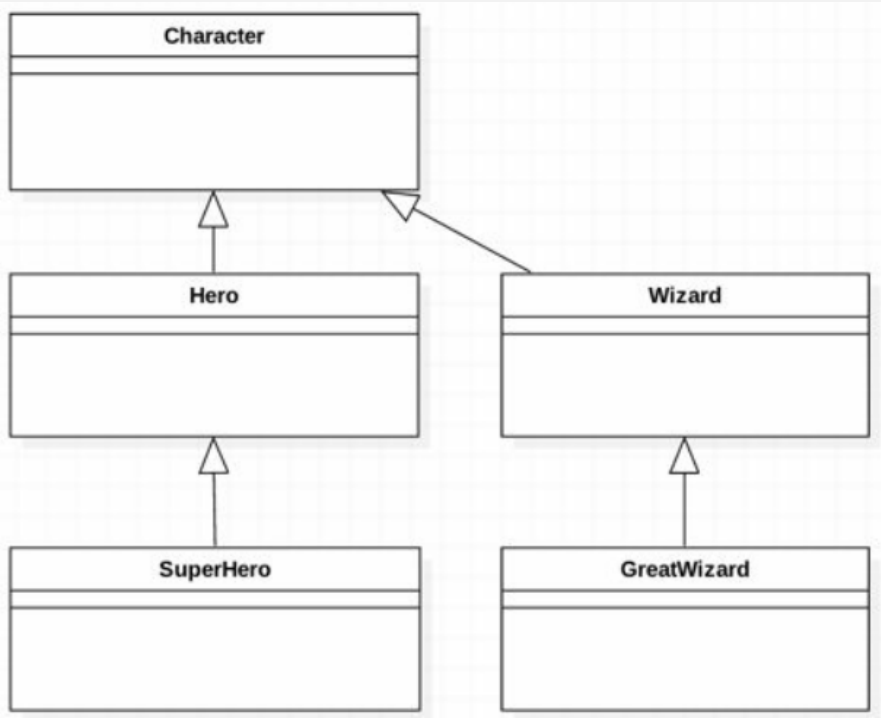


4. Java에서 다중상속은 금지!

1. 수업 내용 정리 [Inheritance]

올바른 상속: "is-a 원칙"

1. 올바른 상속이란, "is-a 원칙"에 따르는 상속을 말한다.
   - SuperHero is a Hero (SuperHero는 Hero의 한 종류이다)
   - Hero is a Person (Hero 는 Person의 한 종류이다)
2. 잘못된 상속
   - 현실 세계의 등장 인물 사이에 개념적으로 is-a 관계가 되지 못함에도 불구하고 상속을 사용한 경우가 "잘못된 상속"
     - Ex] House extends Item (House is a Item ??)
   - 잘못된 상속을 하는 경우 클래스 확장 시 현실세계와 모순 발생
   - 객체 지향의 3대 특징 중 1가지 "다형성"을 이용할 수 없게 된다.
   - House가 Item을 상속받으면? Item은 던질 수 있으나
     현실세계에서 House를 던질 수 있는가?
3. 구체화와 일반화
   - 자식클래스 일수록 구체화 되고,
     부모클래스 일수록 추상적인 것으로
     일반화 된다.



1. 수업 내용 정리 [Inheritance]

1. 상속의 기초
   - extends 키워드 사용해 기존 클래스를 기초로 하는 새로운 클래스 정의 가능
   - 부모클래스의 멤버는 자동으로 자식클래스에 상속되므로, 자식클래스에는 추가된 부분만 기술
   - 오버라이드(Override): 부모클래스의 메소드를 자식클래스에서 재작성 하는 것
   - 클래스에 final 붙이면 클래스 상속 불가, 메소드에 final 붙이면 메소드 오버라이드 불가
   - 상속은 클래스 간 "추상적, 구체적" 관계를 정의하는 역할도 존재
2. 인스턴스
   - 인스턴스는 내부에 부모클래스의 인스턴스를 가지는 다중 구조 가짐
   - 외측의 인스턴스(자식클래스의 인스턴스)에 속하는 메소드가 우선적으로 동작
   - 외측 인스턴스(자식클래스의 인스턴스) 소속 메소드는 super 키워드를 사용해 내측 인스턴스(부모클래스의 인스턴스)의 멤버에 접근 가능
3. 생성자 동작
   - 다중구조의 인스턴스 생성되는데, JVM는 자동적으로 가장 외측 인스턴스(자식클래스의 인스턴스)의 생성자 호출
   - 모든 생성자는, "부모 인스턴스의 생성자"를 호출
   - 생성자의 선두에 super()가 없으면, 암묵적으로 "super();"가 추가됨

2. 실습화면 캡쳐

2. 실습 화면 캡쳐

11-1. "싸우기"와 "도망"만 되는 Hero 클래스

```java
public class Hero {
    private String name = "김영웅";
    private int hp = 100;

    // 싸우기
    public void attack(Kinoko enemy) {
        System.out.println(name + "의 공격!");
        enemy.hp -= 5;
        System.out.println("5포인트의 데미지를 주었다!");
    }

    // 도망
    public void run() {
        System.out.println(name + "는 도망쳤다!");
    }
}
```

11-2. SuperHero 클래스

```java
public class SuperHero {
    private String name = "김영웅";
    private int hp = 100;
    private boolean flying;     // 필드 추가

    // 싸우기
    public void attack(Kinoko enemy) {
        System.out.println(name + "의 공격!");
        enemy.hp -= 5;
        System.out.println("5포인트의 데미지를 주었다!");
    }
    // 도망
    public void run() {
        System.out.println(name + "는 도망쳤다!");
    }
    // 날기
    public void fly() {
        flying = true;
        System.out.println("날았다");
    }
    // 착지
    public void land() {
        flying = false;
        System.out.println("착지했다!");
    }
}
```

2. 실습 화면 캡쳐

11-3. Hero 클래스를 상속한 SuperHero

```java
public class SuperHero extends Hero {
    private boolean flying;     // 추가한 필드

    // 추가한 메소드
    public void fly() {
        flying = true;
        System.out.println("날았다");
    }
    // 추가한 메소드
    public void land() {
        flying = false;
        System.out.println("착지했다!");
    }
}
```

11-4.

```java
public class GameMain {
    public static void main(String[] args) {
        SuperHero superHero = new SuperHero();
        superHero.run();
    }
}
```

GameMain

```
"C:\Program Files\Java\jdk-13.0.2\bin\java.exe"
김영웅는 도망쳤다!

Process finished with exit code 0
```

2. 실습 화면 캡쳐

11-5. 오버라이드 (Override)

```java
public class SuperHero extends Hero {
    private boolean flying;     // 추가한 필드

    public void fly() {
        flying = true;
        System.out.println("날았다");
    }

    public void land() {
        flying = false;
        System.out.println("착지했다!");
    }

    @Override
    public void run() {
        System.out.println("퇴각했다");
    }
}
```

11-6.

```java
public class GameMain {
    public static void main(String[] args) {
        Hero hero = new Hero();
        hero.run();
        SuperHero superHero = new SuperHero();
        superHero.run();
    }
}
```

GameMain

```
"C:\Program Files\Java\jdk-13.0.2\bin\java.exe"
김영웅는 도망쳤다!
퇴각했다

Process finished with exit code 0
```

11-7.

```java
public class Hero {
    private String name = "김영웅";
    private int hp = 100;

    public final void slip() {
        hp -= 5;
        System.out.println(name + "는 미끄러졌다!");
        System.out.println("5의 데미지!");
    }

    public void run() {
        System.out.println(name + "는 도망쳤다!");
    }
}
```

2. 실습 화면 캡쳐

11-8. SuperHero의 추가 사양

```java
public class SuperHero extends Hero {
    private boolean flying;

    @Override
    public void attack(Kinoko enemy) {
        System.out.println(this.name + "의 공격!");
        enemy.hp -= 2;
        System.out.println("5포인트의 데미지를 주었다!");

        if(this.flying) {
            System.out.println(this.name + "의 공격!");
            enemy.hp -= 5;
            System.out.println("5포인트의 데미지를 주었다!");
        }
    }
}
```

11-9. 부모 객체를 참조하는 super 키워드

```java
public class SuperHero extends Hero {
    private boolean flying;

    @Override
    public void attack(Kinoko enemy) {
        super.attack(enemy);

        if(this.flying) {
            System.out.println(this.name + "의 공격!");
            enemy.hp -= 5;
            System.out.println("5포인트의 데미지를 주었다!");
        }
    }
}
```

2. 실습 화면 캡쳐

11-10. 상속과 생성자

```java
public class Hero {
    public Hero() {
        System.out.println("Hero 생성자");
    }
}
```

```java
public class SuperHero extends Hero {
    public SuperHero() {
        System.out.println("SuperHero의 생성자");
    }
}
```

```java
public class GameMain {
    public static void main(String[] args) {
        SuperHero superHero = new SuperHero();
    }
}
```

GameMain

```
"C:\Program Files\Java\jdk-13.0.2\bin\java.exe"
Hero 생성자
SuperHero의 생성자

Process finished with exit code 0
```

11-11. 기본 생성자가 없을 때의 에러 해결

```java
public class Item {
    private String name;
    private int price;

    public Item(String name) {
        this.name = name;
        this.price = 0;
    }

    public Item(String name, int price) {
        this.name = name;
        this.price = price;
    }
}
```

```java
public class Weapon extends Item {
    public Weapon(String name) {
        super(name);
    }

    public Weapon(String name, int price) {
        super(name, price);
    }
}
```

3. 연습문제 11-1

3. 연습문제 11-1

11-1. 문제

- 다음 중에서 "잘못 된 상속"인 것을 모두 구하시오

	슈퍼클래스	서브클래스
1	Person	Student
2	Car	Engine
3	Father	Child
4	Food	Susi
5	SuperMan	Man

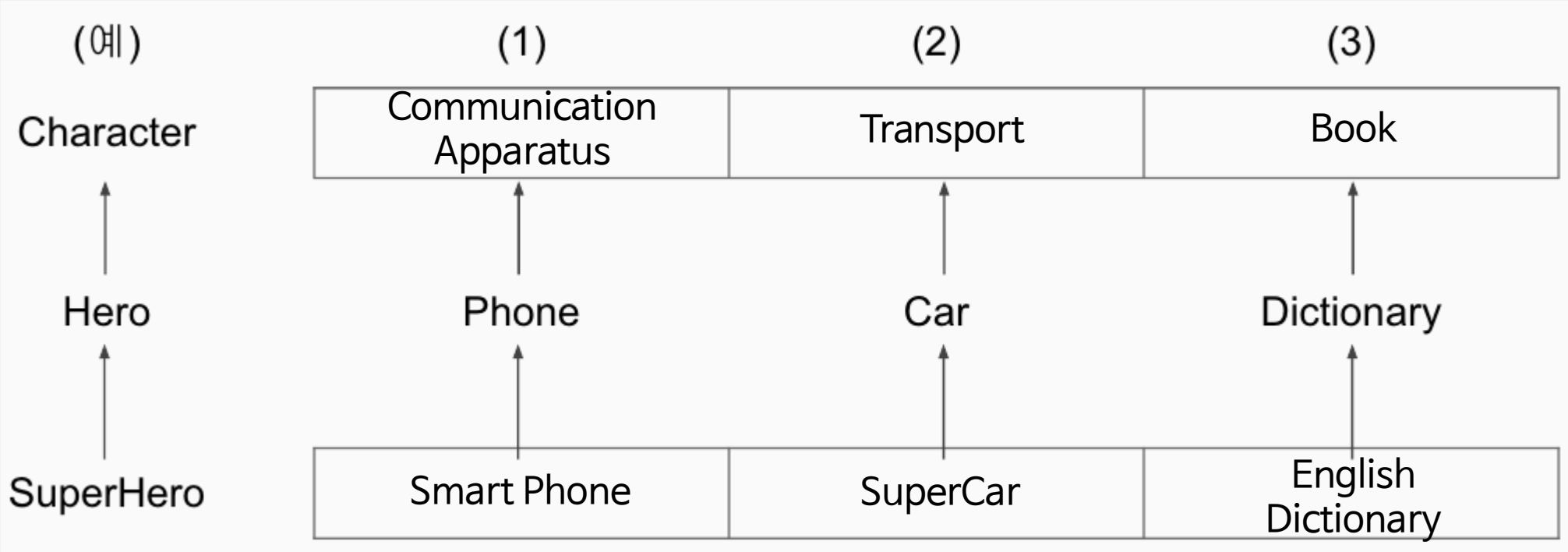
- 올바른 상속: Subclass is a Superclass

1. Student is a Person [O]
2. Engine is a Car [X]
3. Child is a Father [X]
4. Susi is a Food [O]
5. Man is a SuperMan [X]

4. 연습문제 11-2

4. 연습문제 11-2

11-2. 문제 : 다음 클래스에 대해 "부모 클래스"와 "자식 클래스"를 1개씩 생각해 보시오



5. 연습문제 11-3

5. 연습문제 11-3

11-3. 문제 : 이 클래스를 이용해, 다음 사양을 따르는 PoisonKinoko 클래스를 작성하시오.

1. 괴물 독버섯(PoisonKinoko)는, 괴물버섯 (Kinoko) 중에서도 특히 "독 공격"이 되는 것
2. PoisonKinoko 는 아래의 코드로 인스턴스화 되는 클래스임
   PoisonKinoko poisonKinoko = new PoisonKinoko('A');
3. PoisonKinoko는 독 공격이 가능한 남은 횟수를 int 형 필드를 가지고 있고 초기값은 5 이다.
4. PoisonKinoko는 attack() 메소드가 호출되면 다음 내용의 공격을 한다.
   A. 우선, "보통 괴물버섯과 같은 공격"을 한다.
   B. "독 공격의 남은 횟수"가 0이 아니면 다음을 추가로 수행한다
   C. 화면에 "추가로, 독 포자를 살포했다!"를 표시
   D. 용사의 HP의 1/5에 해당하는 포인트를 용사의 HP로부터 감소시키고, "~포인트의 데미지"라고 표시
   E. "독 공격의 남은 횟수"를 1 감소 시킨다

```java
public class Kinoko {
    int hp = 50;
    private char suffix;

    public Kinoko(char suffix) {
        this.suffix = suffix;
    }

    public void attack(Hero hero) {
        System.out.println("키노코 " + this.suffix + " 의 공격");
        System.out.println("10의 데미지");
        hero.setHp(hero.getHp() - 10);
    }
}
```

5. 연습문제 11-3 : Kinoko & PoisonKinoko class

Kinoko class

```java
public class Kinoko {
    private int hp;
    private char suffix;

    public Kinoko(char suffix) {
        this.suffix = suffix;
    }

    public char getSuffix() {...}

    public int getHp() {...}

    public void setHp(int hp) {...}

    public void setSuffix(char suffix) {...}

    public void attack(Hero hero) {
        System.out.println("키노코 " + this.suffix +" 의 공격");
        System.out.println("10의 데미지");
        hero.setHp(hero.getHp() - 10);
    }
}
```

PoisonKinoko class

독 공격 횟수 초기값 = 5

```java
public class PoisonKinoko extends Kinoko {
    private int poisonAttackCount = 5;

    public PoisonKinoko(char suffix) {
        super(suffix);
    }

    public int getPoisonAttackCount() {
        return poisonAttackCount;
    }

    @Override
    public void attack(Hero hero) {
        super.attack(hero);
        if(poisonAttackCount > 0) {
            System.out.println("추가로, 독 포자를 살포했다!");
            int poisonAttack = (int) (hero.getHp() / 5);
            hero.setHp(hero.getHp() - poisonAttack);
            System.out.println(poisonAttack + "포인트의 데미지");
            poisonAttackCount--;
        }
    }
}
```

보통괴물버섯과 같은 공격

독 공격 남은 횟수가 0이 아니면

5. 연습문제 11-3 : Main class

```java
public class GameMain {
    static void printStatus(Hero hero) {
        System.out.println(hero.getName() + "의 상태");
        System.out.println("HP/maxHP: " + hero.getHp() + "/" + Hero.MAX_HP);
        System.out.println("====================");
    }
    static void printStatus(PoisonKinoko poisonKinoko) {
        System.out.println(poisonKinoko.getSuffix() + "의 상태");
        System.out.println("남은 Poison Attack 수: " + poisonKinoko.getPoisonAttackCount());
        System.out.println("====================");
    }

    public static void main(String[] args) {
        Hero hero = new Hero();
        PoisonKinoko poisonKinoko = new PoisonKinoko( suffix: 'A');
        System.out.println("게임 시작");
        System.out.println("====================");
        for (int i = 0; i < 6 ; i++) {
            poisonKinoko.attack(hero);
            System.out.println("====================");
            printStatus(hero);
            printStatus(poisonKinoko);
        }
    }
}
```

5. 연습문제 11-3 : 출력 결과