

# 제8장 인스턴스와 클래스

# 오브젝트를 도출하는 순서

1. 각 오브젝트가 가지고 있을만한 “속성” 이나 “동작” 을 생각해 종류와 내용을 정의
2. 각 오브젝트를 가상세계에 도출, 동작시켜 보기

# 용어 정리

오브젝트 (object) : 현실 세계의 모든 객체

클래스 (class) : 오브젝트를 가상세계 용으로 구체화 한 것 (붕어빵 틀)

인스턴스 (instance) : 클래스를 활용 해 메모리 상에 만들어 낸 것 (붕어빵)

```
1 public class Main {  
2     public static void main(String[] args) {  
3         // 용사여, 가상 세계에 탄생하라  
4         // 괴물 버섯이여, 가상 세계에 탄생하라  
5         // 용사여, 싸워라  
6         // 괴물 버섯이여, 도망가라  
7     }  
8 }
```

이 프로그램을 작성하기 위해 필요한 클래스를 생각해 보자

1. main 메소드를 가지는 1개의 “신(GOD) 클래스”
2. “등장 인물 클래스”



용사

이름  
HP

싸우기  
도망  
앞기  
넘어지기  
잠자기

```
1 public class Hero {  
2     String name;  
3     int hp;  
4  
5     void attack() {}  
6     void run() {}  
7     void sit(int sec) {}  
8     void slip() {}  
9     void sleep() {}  
10 }
```



버섯

HP : 35

레벨 : 10

싸우기  
도망  
수면가스

```
1 public class Kinoko {  
2     int hp;  
3     final int LEVEL = 10;  
4 }
```

```
1  public class Hero {
2      String name;
3      int hp;
4
5      void attack() {}
6      void run() {}
7      void sit(int sec) {}
8      void slip() {}
9      void sleep() {
10         this.hp = 100;
11         System.out.println(this.name + "는 잠을 자고 회복했다!");
12     }
13 }
```

# 클래스명과 멤버변수명의 명명 규칙

클래스명	명사	단어의 맨 처음은 대문자	Hero, MonsterInfo
필드 명	명사	최초 이외의 단어의 맨 처음은 대문자	mLevel, mList, level, items, itemList
메소드 명	동사	최초 이외의 단어의 맨 처음은 대문자	attack, findWeakPoint



```
1  public class Hero {
2      String name;
3      int hp;
4
5      void attack() {}
6      void run() {
7          System.out.println(this.name + "는 도망쳤다!");
8          System.out.println("GAME OVER");
9          System.out.println("최종 HP는" + this.hp + " 입니다");
10     }
11     void sit(int sec) {
12         this.hp += sec; // 앉은 초 수만큼 HP 가 증가
13         System.out.println(this.name + "는 " + sec + "초 앉았다");
14         System.out.println("HP가 " + sec + "포인트 회복되었다");
15     }
16     void slip() {
17         this.hp -= 5;
18         System.out.println(this.name + "는 넘어졌다!");
19         System.out.println("5의 데미지!");
20     }
21     void sleep() {
22         this.hp = 100;
23         System.out.println(this.name + "는 잠을 자고 회복했다!");
24     }
25 }
```

# 클래스 정의에 따른 효과

1. 정의한 클래스로 **인스턴스를 생성** 할 수 있다
2. 이 클래스로 생성한 인스턴스를 넣을 수 있는 **새로운 변수의 타입이 이용 가능** 해 진다

Hero 클래스를 정의하면 Hero 타입의 변수가 이용 가능

```
Hero hero;
```

클래스를 정의하면 **Java** 에서 이용가능한 타입의 종류가 점점 늘어 남

```
1 public class Main {  
2     public static void main(String[] args) {  
3         // 가상 세계에 용사를 생성  
4  
5         // 생성된 용사에게 최초의 HP 와 이름을 설정  
6  
7         // 용사에게 '5초 앓기', '넘어지기', '25초 앓기', '도망' 을 지시  
8     }  
9 }
```

```
1 public class Main {  
2     public static void main(String[] args) {  
3         // 가상 세계에 용사를 생성  
4         Hero hero = new Hero();  
5  
6         // 생성된 용사에게 최초의 HP 와 이름을 설정  
7  
8         // 용사에게 '5초 앞기', '넘어지기', '25초 앞기', '도망' 을 지시  
9     }  
10 }
```

```
1 public class Main {  
2     public static void main(String[] args) {  
3         // 가상 세계에 용사를 생성  
4         Hero hero = new Hero();  
5  
6         // 생성된 용사에게 최초의 HP 와 이름을 설정  
7         hero.name = "준석";  
8         hero.hp = 100;  
9         System.out.println("용사 " + hero.name + " 를 생성했습니다!");  
10  
11         // 용사에게 '5초 앞기', '넘어지기', '25초 앞기', '도망' 을 지시  
12     }  
13 }
```

```
1 public class Main {  
2     public static void main(String[] args) {  
3         // 가상 세계에 용사를 생성  
4         Hero hero = new Hero();  
5  
6         // 생성된 용사에게 최초의 HP 와 이름을 설정  
7         hero.name = "준석";  
8         hero.hp = 100;  
9         System.out.println("용사 " + hero.name + " 를 생성했습니다!");  
10  
11        // 용사에게 '5초 앞기', '넘어지기', '25초 앞기', '도망' 을 지시  
12        hero.sit(5);  
13        hero.slip();  
14        hero.sit(25);  
15        hero.run();  
16    }  
17 }
```

```
// 가상 세계에 용사를 생성
int yongsa_hp = 100;
String yongsa_name = "준석";

// 생성된 용사에게 최초의 HP 와 이름을 설정
System.out.println("용사 " + yongsa_name + " 를 생성했습니다!");

// 용사에게 '5초 앓기', '넘어지기', '25초 앓기', '도망' 을 지시
yongsa_hp += 5;
System.out.println(yongsa_name + "는 5초 앓았다");
System.out.println("HP가 5포인트 회복되었다");
yongsa_hp -= 5;
System.out.println(yongsa_name + "는 넘어졌다!");
System.out.println("5의 데미지!");
yongsa_hp += 25;
System.out.println(yongsa_name + "는 25초 앓았다");
System.out.println("HP가 25포인트 회복되었다");
System.out.println(yongsa_name + "는 도망쳤다!");
System.out.println("GAME OVER");
System.out.println("최종 HP는" + yongsa_hp + " 입니다");
```

```
1  public class Kinoko {  
2      int hp;  
3      final int LEVEL = 10;  
4      char suffix;  
5  
6      void run() {  
7          System.out.println("괴물 버섯 " +  
8              this.suffix + "는 도망갔다!");  
9      }  
10 }
```



```
Hero hero = new Hero();  
hero.name = "준석";  
hero.hp = 100;  
  
Kinoko kinoko1 = new Kinoko();  
kinoko1.hp = 50;  
kinoko1.suffix = 'A';  
  
Kinoko kinoko2 = new Kinoko();  
kinoko2.hp = 48;  
kinoko2.suffix = 'B';  
  
// 모험의 시작  
hero.slip();  
kinoko1.run();  
kinoko2.run();  
hero.run();
```

### 인스턴스와 클래스

- 인스턴스와 클래스는 완전히 다른 것이다. 혼동 하지 말자
- 가상 세계에서 활동하는 것은 인스턴스 (오브젝트)
- 인스턴스를 생성하기 위한 틀이 클래스

### 필드와 메소드

- 클래스에는 속성을 필드로, 동작을 메소드로 선언한다
- **final** 이 붙은 필드는 상수 필드로서 값이 불변이다
- **this** 는 자기 자신의 인스턴스를 표시하는 키워드

### 클래스 타입

- 클래스를 정의하면, 그 클래스 타입의 변수를 선언 할 수 있다
- 어떤 클래스 타입 변수는 그 클래스의 인스턴스를 담을 수 있다

### 인스턴스화

- **new** 연산자를 사용하여 클래스로부터 인스턴스를 생성
- 어떤 클래스 타입 변수에 인스턴스가 담겨 있을 때 “변수명.필드명” 이나 “변수명.메소드명 ()” 으로 그 인스턴스의 필드나 메소드를 이용할 수 있다

## 연습문제 8-1

현실세계의 성직자 “클레릭” 를 표현하는 클래스 **Cleric** 를 작성 하시오.

속성이나 동작은 선언 할 필요 없습니다. (속은 아무것도 작성하지 않아도 됨)

## 연습문제 8-2

성직자는 용사 처럼 이름과 **HP**를 가지고 있고, 추가로 마법을 사용 하기 위한 **MP**를 가지고 있다. 거기에 연습 8-1에서 선언한 내용이 없는 **Cleric** 클래스에 “이름”, “**HP**”, “최대 **HP**”, “**MP**”, “최대 **MP**”를 속성으로 추가 하시오.

또한 **HP**와 최대 **HP**는 정수로 초기치 50, **MP**와 최대 **MP**는 정수로 초기치 10으로 하고,

최대 **HP**와 최대 **MP**는 상수 필드로 선언 하시오.

## 연습문제 8-3

성직자는 “셀프 에이드” 라는 마법을 사용할 수 있고, MP를 5소비하는 것으로 자신의 HP 를 최대 HP 까지 회복할 수 있다.

연습 8-2 에 선언한 Cleric 클래스에 “selfAid()” 메소드를 추가 하시오.

또한, 이 메소드는 인수가 없고, 리턴 값도 없다.

## 연습문제 8-4

성직자는 “기도하기” (**pray**) 라는 행동을 취할 수 있고, 자신의 **MP**를 회복한다.

회복량은 기도한 시간(초)에 랜덤하게 0 ~ 2포인트의 보정을 한 양이다 (3초 기도하면 회복량은 3 ~ 5 포인트). 단, 최대 **MP** 보다 더 회복하는 것은 불가능하다.

그리고, 연습 8-3에서 선언한 **Cleric** 클래스에 “**pray()**” 메소드를 추가하시오.

이 메소드는 인수에 “기도할 시간(초)”를 지정할 수 있고, 리턴 값은 “실제로 회복된 **MP** 양” 을 반환한다.