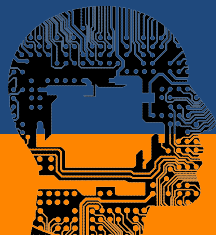




한국IT진흥부설

정보보호교육학원 아이섹



자바 프로그래밍 기초

키워드 this

By SoonGu Hong(Kokono)



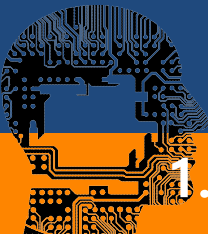
한국IT진흥부설

정보보호교육학원

아이섹

자바 프로그래밍 기초

1. 스스로의 객체참조 this



1-1. 인스턴스 멤버

- 클래스의 구성요소(필드, 생성자, 메서드)를 멤버라고 부릅니다.
- 그 중에서도 객체 생성을 통해서만 사용할 수 있는 멤버를 인스턴스(instance) 멤버라고 부릅니다.
- 인스턴스 멤버는 객체 생성 없이는 사용할 수 없습니다.
- 아래의 클래스 Player는 name필드와 attack메서드를 인스턴스 멤버로 가지고 있습니다.

```
public class Player {  
    String name;  
    void attack() { ... }  
}
```



```
Player kim = new Player();  
  
kim.name = "김빠빠";  
kim.attack();
```

1-2-1. 클래스 내부에서 인스턴스를 어떻게 구분?


```
public class Player {  
    String name;  
  
    void attack(Player target) {  
        출력: name이 name을 공격합니다.  
    }  
}
```

클래스 안에서
kim의 name과 park의 name을
어떻게 구분시킬 것인가?

```
Player kim = new Player();  
Player park = new Player();  
  
kim.name = "김빠빠";  
park.name = "박까까";  
kim.attack(park);
```

1-2-2. 클래스 내부에서 인스턴스를 어떻게 구분?

```
public class Player {  
    String name;  
  
    void attack(Player target) {  
        kim.name이 park.name을 공격합니다. (X)  
    }  
}
```




클래스는 설계도라고 했습니다.
설계도 안에서 kim, park과 같이
클래스 밖에서 만든 변수이름을
직접 쓴다면 변수이름이 바뀔 때
대처가 불가능하겠죠?

```
Player kim = new Player();  
Player park = new Player();  
  
kim.name = "김빠빠";  
park.name = "박까까";  
kim.attack(park);
```

1-2-3. 클래스 내부에서 인스턴스를 어떻게 구분?

```
public class Player {  
    String name;  
  
    void attack(Player target) {  
        this.name이 target.name을 공격합니다. (O)  
    }  
}
```



park에 대한 처리는 매개변수
target으로 정규화시키면 됩니다.
그렇다면 때린 주체인 kim은 어떻
게 할까요?
그 정답은 **this** 키워드에 있습니다.

```
Player kim = new Player();  
Player park = new Player();  
  
kim.name = "김빠빠";  
park.name = "박까까";  
kim.attack(park);
```

1-3. this는 자기 자신을 의미한다.

```
Player kim = new Player();  
Player park = new Player();
```

```
kim.name = "김빠빠";  
park.name = "박까까";
```

1. kim.attack(park);

2. park.attack(kim);

- 우리가 자기 자신을 1인칭으로 가리킬 때 "나"라고 부르듯, 클래스 내부에서 자기 자신의 객체를 가리킬 때 "this"라고 부릅니다.

- 왼쪽 코드에서 1번의 경우는 kim이 this가 되며 2번의 경우에선 park이 this가 됩니다.

```
void attack(Player target) {  
    this.name이 target.name을 공격합니다.  
}
```

1-4. 메서드나 생성자 내부에서 매개변수와 필드의 구분

```
public class Person {
```

```
    String name;  
    int age;
```

```
    Person(String name, int age) {  
        this.name = name;  
        this.age = age;  
    }
```

```
}
```

- 관례상 필드값을 초기화하는 매개변수는 필드명과 동일하게 선언합니다.

- 그러나 생성자나 메서드 내부에서 사용하는 매개변수의 이름이 필드이름과 같다면 2개를 구분할 수 없을 것입니다.

- 따라서 이럴 때 필드명 앞에 this를 표기함으로써 해당 데이터가 필드임을 명시합니다.

```
Person p = new Person("뽀로로", 10);
```




한국IT진흥부설

정보보호교육학원

아이섹

자바 프로그래밍 기초

2. 다른 생성자 호출 this()

2-1. 생성자 중복코드 제거하기

```
Phone(String model) {  
    this.model = model;  
    this.color = "코발트 블랙";  
    this.price = 500000;  
}  
  
Phone(String model, String color) {  
    this.model = model;  
    this.color = color;  
    this.price = 650000;  
}  
  
Phone(String model, String color, int price) {  
    this.model = model;  
    this.color = color;  
    this.price = price;  
}
```

- 왼쪽에 오버로딩된 생성자들을 보면 뭔가 중복된 코드들이 느껴지지 않나요??

- 3개의 생성자 모두 3가지 필드를 초기화하는 작업은 동일한데 말이죠.

- 이럴 때 `this()`를 사용하면 중복코드를 많이 없앨 수 있습니다.

2-2. this() - 자신의 다른 생성자 호출

```
Phone(String model) {  
    this(model, "코발트 블랙", 500000);  
}  
  
Phone(String model, String color) {  
    this(model, color, 650000);  
}  
  
Phone(String model, String color, int price) {  
    this.model = model;  
    this.color = color;  
    this.price = price;  
}
```

호출

호출

공통 코드

감사합니다
THANK YOU