'this'는 '이것', 즉 가까운 어떤 것을 가리키는 대명사죠? 자바에서도 **this** 는 무언가를 가리키는 역할을 합니다!

this 는 메소드에서 현재 인스턴스를 가리키는 역할을 합니다. 이 코드를 살펴보세요.

```
class Person {
    private String name;

    public Person(String pName) {
        name = pName;
    }

    public void sayHello() {
        System.out.println("Hi. My name is " + this.name);
    }
}


class Main {
    public static void main(String[] args) {
        Person p1 = new Person("김신의");
        p1.sayHello();

        Person p2 = new Person("강영훈");
        p2.sayHello();
    }
}
```

```
Hi. My name is 김신의
Hi. My name is 강영훈
```

p1.sayHello(); 는 **sayHello** 메소드를 호출하죠? **sayHello** 메소드가 실행될 때 **this** 는 현재 인스턴스인 **p1** 입니다. **p1.name** 은 **"김신의"** 이기 때문에 위와 같은 출력값이 나옵니다.

마찬가지로 **p2.sayHello();** 를 하면 **this** 는 현재 인스턴스인 **p2** 를 가리킵니다.

그럼 이제 **this** 를 다양하게 활용해봅시다.

일반 메소드에서의 사용

우리가 작성한 **Person** 클래스의 **setName** 메소드 입니다.

```
public class Person {
    private String name;

    ...

    public void setName(String pName) {
        name = pName;
    }

    ...
}
```

이 메소드는 **Person** 클래스 내부에 있습니다. 그런데 만약 파라미터의 이름을 **name** 으로 바꾸면 어떻게 될까요?

```
public class Person {
    private String name;

    ...

    public void setName(String name) {
        name = name;
    }

    ...
}
```

name = name; 이라는 애매한 코드가 생기게 됩니다. **두 name 은 서로 같은 변수이며, 클래스의 인스턴스 변수가 아닌 파라미터를 가리키게 됩니다.** 해당 위치(**setName** 메소드)에서 인스턴스 변수 **name** 보다 파라미터 변수 **name** 이 더 가깝기 때문이죠!

이런 상황에서, 이 클래스로 생성된 인스턴스 변수 **name** 을 가리키고 싶다면? 아래처럼 **this** 을 사용하면 됩니다!

```
public void setName(String name) {
    this.name = name;
}
```

이제 클래스의 인스턴스 변수 **name** 과 파라미터 변수 **name** 이 확실하게 구분되죠?

생성자에서의 사용

생성자에서도 동일하게 적용 가능합니다.

저희가 과제로 만든 **BankAccount** 의 생성자를 보겠습니다.

```
public class BankAcount {
    private Person owner;

    ...

    public BankAccount(Person pOwner) {
        owner = pOwner;
    }

    ...
}
```

여기서도 **this** 를 사용할 수 있겠죠!

```
public BankAccount(Person owner) {
    this.owner = owner;
}
```

파라미터로 **owner** 인스턴스를 받고, 바로 **BankAccount** 의 인스턴스 변수에 넣어주었습니다.

그렇다면 **BankAccount** 의 인스턴스를 **Person** 의 인스턴스에 넣어주는 것은 어떻게 할 수 있을까요?

원래는 이렇게 했었는데,

```
// 은행 계좌 생성
BankAccount a1 = new BankAccount(p1);
a1.setBalance(10000);

p1.setAccount(a1); // 이 부분은 어떻게 처리할까요?
// a1.setOwner(p1); 생성자에서 처리하였음.
```

this 키워드를 사용하면 이렇게 바꿀 수 있습니다.

```
public class BankAccount {
    // 생성자
    public BankAccount(Person owner) {
        this.owner = owner;
        owner.setAccount(this);
    }
}
```

생성자로서의 사용

생성자에서 **this** 의 중요한 역할이 또 있습니다! **this** 는 현재 클래스의 생성자로 사용할 수도 있습니다. 예시를 봐야 이해가 쉽겠죠?

이렇게 생성자가 여러 개 있다고 가정합시다.

이렇게 생성자가 여러 개 있더라도 가능합니다.

```
public Person(String name) {
    this.name = name
    age = 12;
    cashAmount = 0;
}

public Person(String name, int age) {
    this.name = name;
    this.age = age;
    cashAmount = 0;
}

public Person(String name, int age, int cashAmount) {
    if (age < 0) {
        this.age = 12;
    } else {
        this.age = age;
    }

    if (cashAmount < 0) {
        this.cashAmount = 0;
    } else {
        this.cashAmount = cashAmount;
    }
    this.name = name;
}
```

뭔가 반복적인 게 많은데, **this** 를 사용하면 훨씬 깔끔하게 작성할 수 있습니다.

```
public Person(String name) {
    this(name, 12, 0); // 12살을 기본 나이로 설정, 초기 현금 보유액은 0원.
}

public Person(String name, int age) {
    this(name, age, 0); // 초기 현금 보유액은 0원.
}

public Person(String name, int age, int cashAmount) {
    if (age < 0) {
        this.age = 12;
    } else {
        this.age = age;
    }

    if (cashAmount < 0) {
        this.cashAmount = 0;
    } else {
        this.cashAmount = cashAmount;
    }
    this.name = name;
}
```

첫 번째 경우의 **`this(name, 12, 0);`** 은 세 번째 생성자 **`Person(name, 12, 0);`** 에 각각의 파라미터를 넘겨 호출하는 것과 같다고 생각할 수 있고, 두 번째 경우의 **`this(name, age, 0);`** 은 **`Person(name, age, 0);`** 에 파라미터를 넘겨 호출하는것과 같다고 생각할 수 있습니다.

이렇게 파라미터가 가장 많은 생성자를 파라미터가 적은 쪽에서 호출 하는 것이 가장 효율적입니다!

☑ 수업을 완료하셨으면 체크해주세요.

수강생 Q&A 보기

(/questions?assignment_id=439&sort_by=popular)

질문하기

(/questions/new?assignment_id=439&op1=%EA%B0%9D%EC%B2%B4+%EC%A7

이전 강의 this (/assignments/342)

다음 강의 특가 할인 매장 (/assignments/441)