

자식 클래스에서 **super** 키워드를 사용하는 경우는 두 가지입니다:

1. 자식 클래스가 부모 클래스의 변수, 메소드를 사용할 때
2. 자식 클래스가 부모 클래스의 생성자를 사용할 때

부모의 변수, 메소드를 사용할 때

```
public class BankAccount {  
    .  
    .  
    .  
    boolean withdraw(int amount) {  
        ...  
    }  
}  
  
public class TransferLimitAccount extends BankAccount {  
    private int transferLimit;  
  
    @Override  
    boolean withdraw(int amount) {  
        if (amount > transferLimit) {  
            return false;  
        }  
  
        return super.withdraw(amount);  
    }  
}
```

자식 클래스의 **withdraw** 는 부모 클래스의 **withdraw** 메소드를 오버라이드 했는데, 정의 내에서 부모 클래스의 **withdraw** 메소드를 사용합니다. 이렇게 부모 클래스의 메소드(또는 변수)를 사용할 때는 **super** 를 사용하면 됩니다.

부모의 생성자를 사용할 때

자식 클래스의 인스턴스가 생성되기 위해서는 부모 클래스와 자식 클래스의 초기 설정을 모두 해주어야 합니다. 따라서 부모 클래스의 생성자도 불러야하는 것이죠.

부모 클래스의 생성자는 **super** 키워드를 사용해서 부를 수 있습니다. 예전에 **this** 를 사용해서 생성자를 부르는 것과 같은 개념입니다.

```

public class BankAccount {
    ...
    public BankAccount(int balance) {
        this.balance = balance;
    }
}

public class TransferLimitAccount extends BankAccount {
    ...
    public TransferLimitAccount(int balance, int transferLimit) {
        super(balance);
        this.transferLimit = transferLimit;
    }
}

```

super 생성자 사용 규칙

```

public class Parent {
    ...
}

public class Child extends Parent {
    ...
}

```

부모 클래스의 생성자를 사용하는 데에는 몇 가지 규칙이 있습니다.

(1) 먼저, 자식 클래스의 인스턴스 생성시 부모 클래스의 생성자는 반드시 불립니다.

```
Child c = new Child(); // Child 인스턴스 생성시 Parent의 생성자도 불림
```

인스턴스의 생성을 위해선 생성자가 반드시 필요하다는 것을 배웠습니다. 자식 클래스의 인스턴스를 만들기 위해선 자식 클래스의 생성자는 물론이고, 부모 클래스의 생성자도 불러야 합니다. 상속이 여러 단계로 되면 맨 위쪽까지 다 불리게 됩니다.

만약 부모 클래스에 명시된 생성자가 없으면 자동으로 제공되는 부모 클래스의 기본 생성자가 불립니다.

(2) 또, 부모 클래스에 기본 생성자가 없는 경우, 즉, 파라미터가 없는 생성자가 없는 경우, 자식 클래스에서 반드시 직접 (코드로 써서) 부모 클래스의 생성자 호출을 시켜주어야 합니다.

```

public class Parent {
    // 별도로 생성자가 지정되어있는 경우 (파라미터 없는 기본 생성자가 없는 경우)
    public Parent(int a, int b) {
        ...
    }
    ...
}

public class Child extends Parent {
    public Child() {
        super(0, 0); // 자식 클래스에 생성자를 만들어 'super(int, int)'를 호출해 주어야 함
        ...
    }
}

```

(3) 그리고 그렇게 부모 클래스의 생성자를 호출할 때는 자식 클래스의 생성자 맨 윗줄에 적어야 합니다. 마치 부모 클래스의 생성자가 불린 다음 자식 클래스의 생성자가 불린다고 생각할 수 있습니다.

아래 코드를 실행시키면 오류가 발생합니다.


```


public class Child extends Parent {
    public Child() {
        int a = 0;
        ...
        super(0, 0); // 이 곳에서 호출할 수 없음. 반드시 첫 번째 줄에 있어야 함.
    }
}

```



수업을 완료하셨으면 체크해주세요.

 수강생 Q&A 보기

 질문하기

assignment_id=420&sort_by=popular)
(/questions/new?

assignment_id=420&op1=%EA%B0%9D%EC%B2%B4+%EC%A7

< 이전 강의 (/assignments/415)
super

다음 강의 (/assignments/488)
super 퀴즈 >