

# INSTANT UML

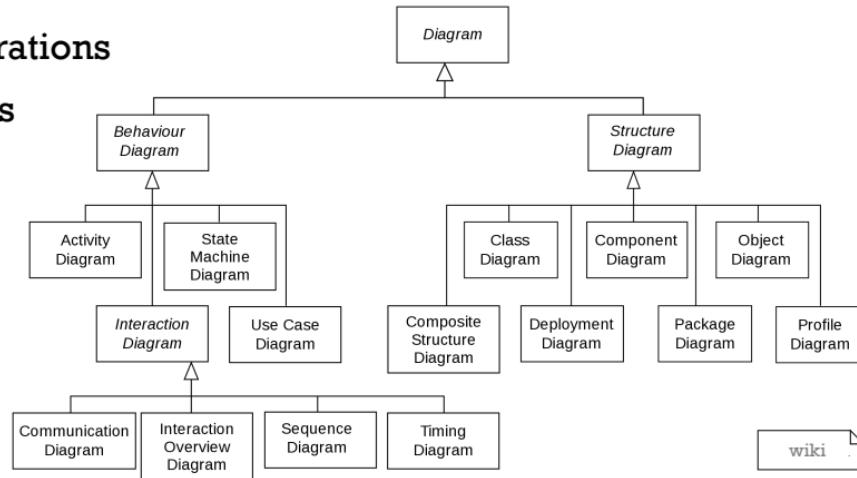
Class Diagram

Sequence Diagram



# CLASS DIAGRAM

- Type of **static structure diagram** that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects.
- Shows Class with attributes and operations
- Shows Relationships between classes



# CLASS

- Represented as square box contains three components
  - Class Name
  - Attribute
  - Operation
- Prefix (Access privileges)
  - +: public
  - - : private
  - #: protected
  - ~: package

«Stereotype» ClassName
-attribute #hart: Hart {composite} +id: int
+operation(i: int): int + <i>abstractOperation()</i> -noShow()



# PRACTICE: CLASS

- 사람

```
사람
```

```
public class 자동차 {  
    private SteeringWheel 핸들;  
    private Wheek 바퀴;  
    private Engine 엔진;
```

```
public void 전진(){  
}  
public Boolean 멈춤(){  
    return True;  
}
```

- 제목을 가진 책

```
책  
제목
```

```
public class Book {  
    private String title;
```

attribute들

나중에  
handle  
할인할 부분

- 바퀴와 엔진, 핸들을 가지고 앞으로 가거나 멈출 수 있는 자동차

자동차
핸들
바퀴
엔진
전진()
멈춤()

```
public class Person {  
}
```



# 액체 찾기

=대출

② 책을 6권까지 빌릴 수 있는 권한

- 도서관 시스템을 만들려고 한다. 이 도서관에서는 독서회원을 가지고 있으며 독서회원은 책을 6권까지 빌릴 수 있는 권한을 가진다. 도서관이 소장하고 있는 책은 빌릴 수 있는 일반도서와 빌릴 수 없는 지정도서 두 종류로 나뉘며 같은 책이 여러 카페 있을 수 있다. 독서회원은 책이 모두 대출되어 없으면 예약을 할 수 있다.

③ 책을 통해 copy 가능

액체

① 독서회원을 알아야 함

책의 종류 두 가지로 나뉨

- 책임의 존재 (주어진 문제 해결에 공헌하는 책임)
- 알아야 할 책임
- 행해야 할 책임

예약

③ 책의 정보를  
갖고 있어야 할 책임.

책임으로 액체 찾기

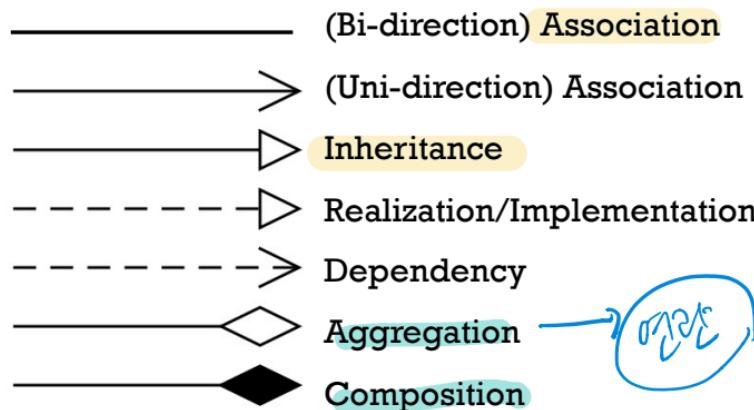


ERPK class

sy dass

schief.

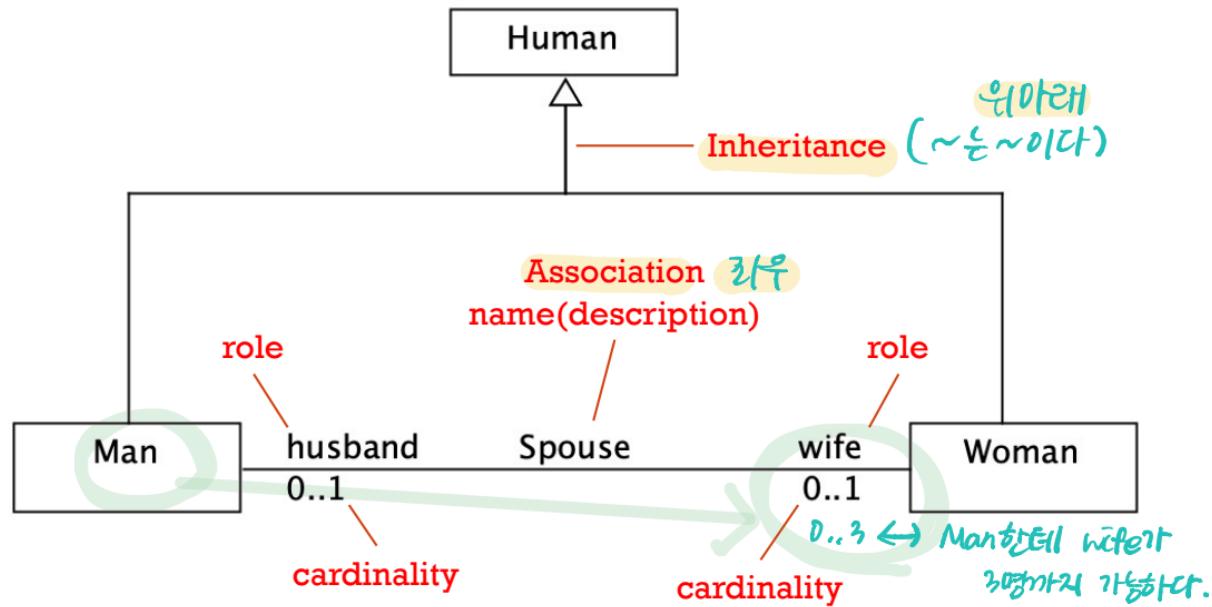
# RELATIONSHIPS



- Name (Description)
- Role name
- Multiplicity(Cardinality)
  - 1
  - 1,3,5
  - 0..5
  - \*
  - 3..\*
  - 0..1



# REPRESENTING RELATIONSHIP



# PRACTICE: ASSOCIATION

- 도서관에는 많은 책이 있다

도서관에는 책이 많이 있다.



- 도서관에는 독서회원들이 있다



```
public class Library {  
    private Human[] member;  
}
```

role 이름

```
public class Human {  
    private Library myLibrary = null;  
}
```



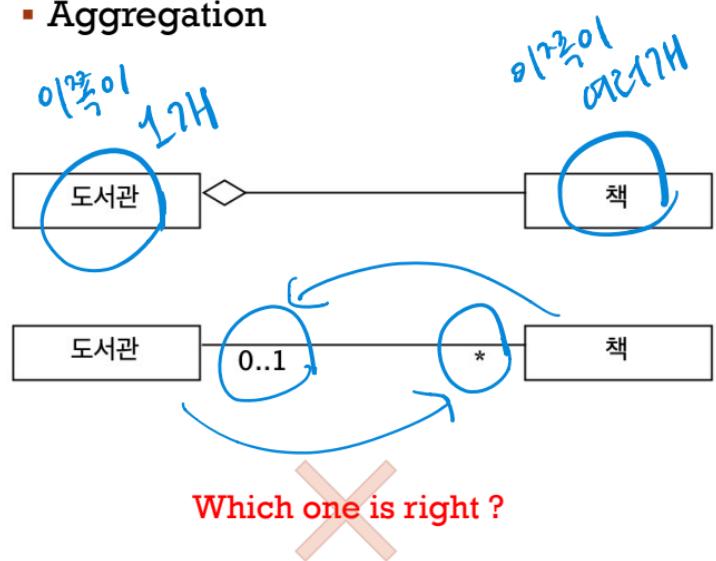
# BUILDING ASSOCIATION

- 아래 문장들을 하나의 Class Diagram로 그려주세요
- 도서관에는 많은 책이 있다
- 도서관에는 독서 회원들이 있다
- 독서 회원은 도서관에서 책을 6권까지 빌릴 수 있다

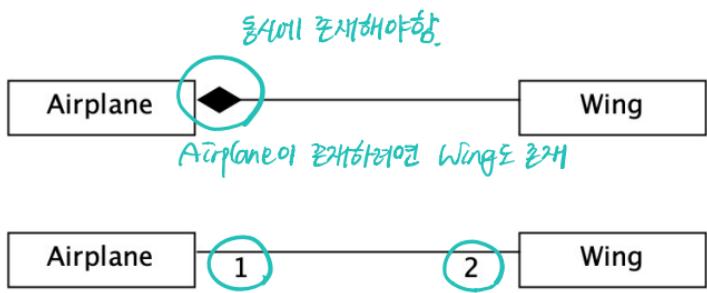


# AGGREGATION, COMPOSITION

- Aggregation



- Composition



Which one is right ?

Which one do you prefer ?



# DEPENDENCY

```
public class Waitress {  
    private Cook sam;  
}
```

```
public class Cook {  
}
```



Uni-directional association



Dependency

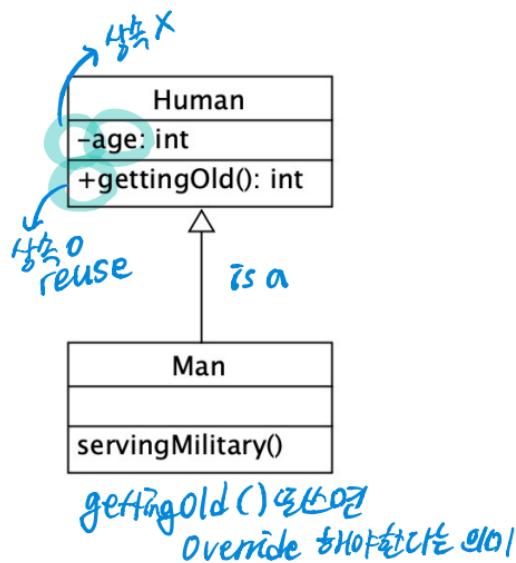
```
Public class Waitress {  
    public void placedOrder() {  
        Cook sam = new Cook();  
    }  
}  
  
public class Cook {  
}
```

지속적X  
필요할 때 사용한다는 의미

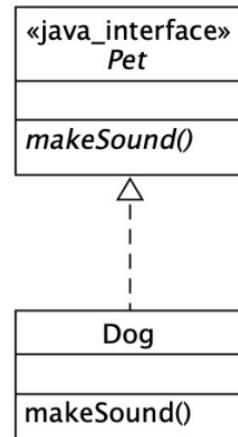


# INHERITANCE, REALIZATION

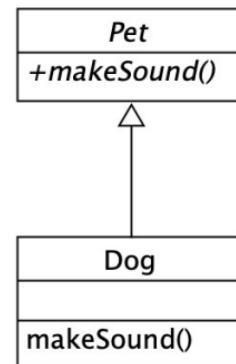
- Inheritance



- Realization



Abstract Method.



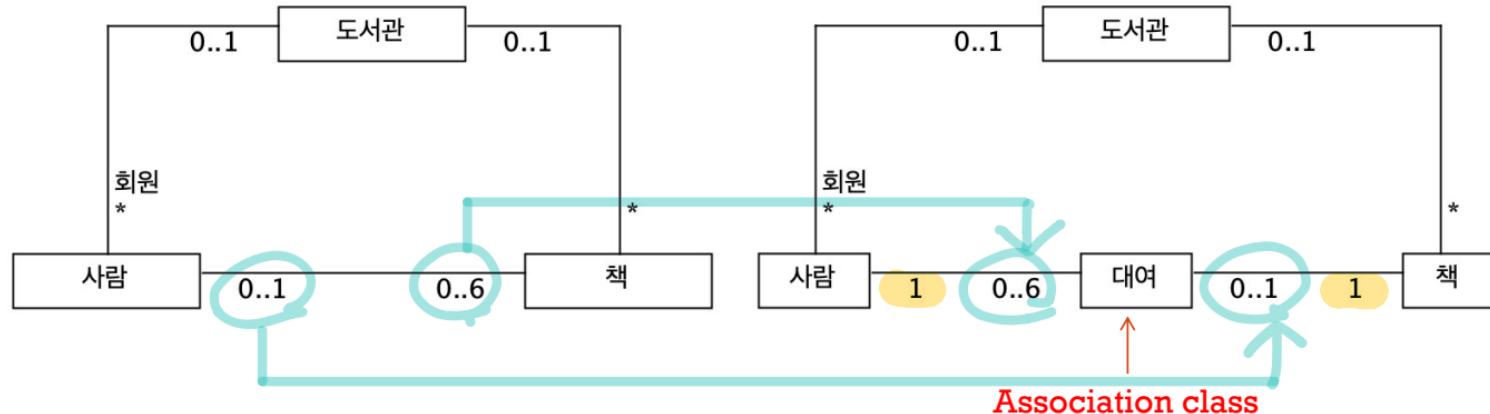
Good enough!



# ASSOCIATION CLASS

- 하나의 Class Diagram로 그려주세요

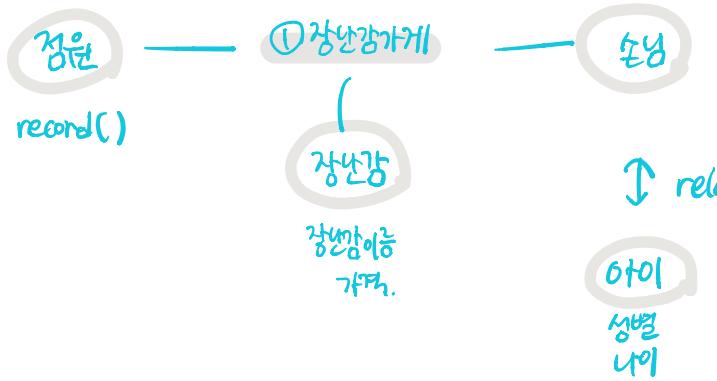
- 도서관에는 많은 책이 있다. 도서관에는 독서 회원들이 있다. 독서 회원은 도서관에서 책을 6권까지 빌릴 수 있다.



# PRACTICE: CLASS DIAGRAM

- 도서관 시스템을 만들려고 한다. 이 도서관에서는 독서회원을 가지고 있으며 독서회원은 책을 6권까지 빌릴 수 있는 권한을 가진다. 도서관이 소장하고 있는 책은 빌릴 수 있는 일반도서와 빌릴 수 없는 지정도서 두 종류로 나뉘며 같은 책이 여러 카페 있을 수 있다. 독서회원은 책이 모두 대출되어 없으면 예약을 할 수 있다.
- 장난감 가게 고객 시스템을 만들고 있다. 손님이 아이에게 적합한 장난감을 추천 받기 위해서 아이의 성별과 나이를 알려주면 점원은 아이에게 적당한 장난감을 추천해 준다. 손님은 추천 받은 장난감의 가격을 알아 보고 적당한 가격이면 구입을 한다. 점원은 판매 기록을 남긴다.

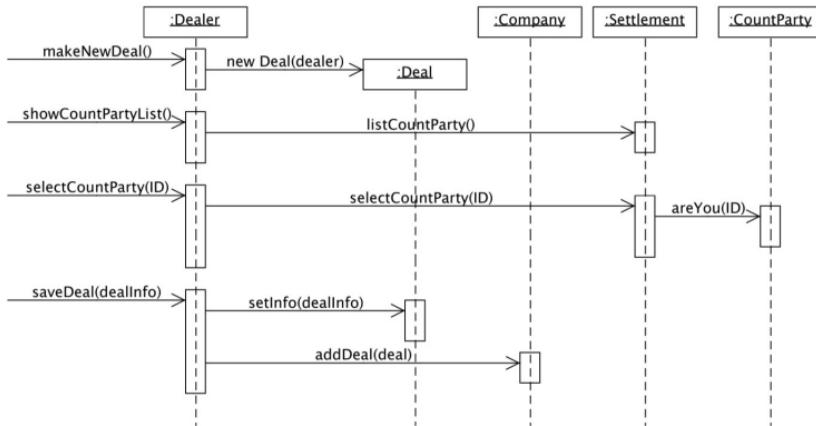




# SEQUENCE DIAGRAM

- Type of **dynamic behavior diagram** shows object interactions arranged in time sequence.
- Aka **event diagrams** or **event scenarios**.

- Interaction Diagram
  - Sequency Diagram
  - Communication Diagram (Collaboration)



# OBJECT INSTANCE

aObject:**ClassName**

:Student
age = 17

noName:Man

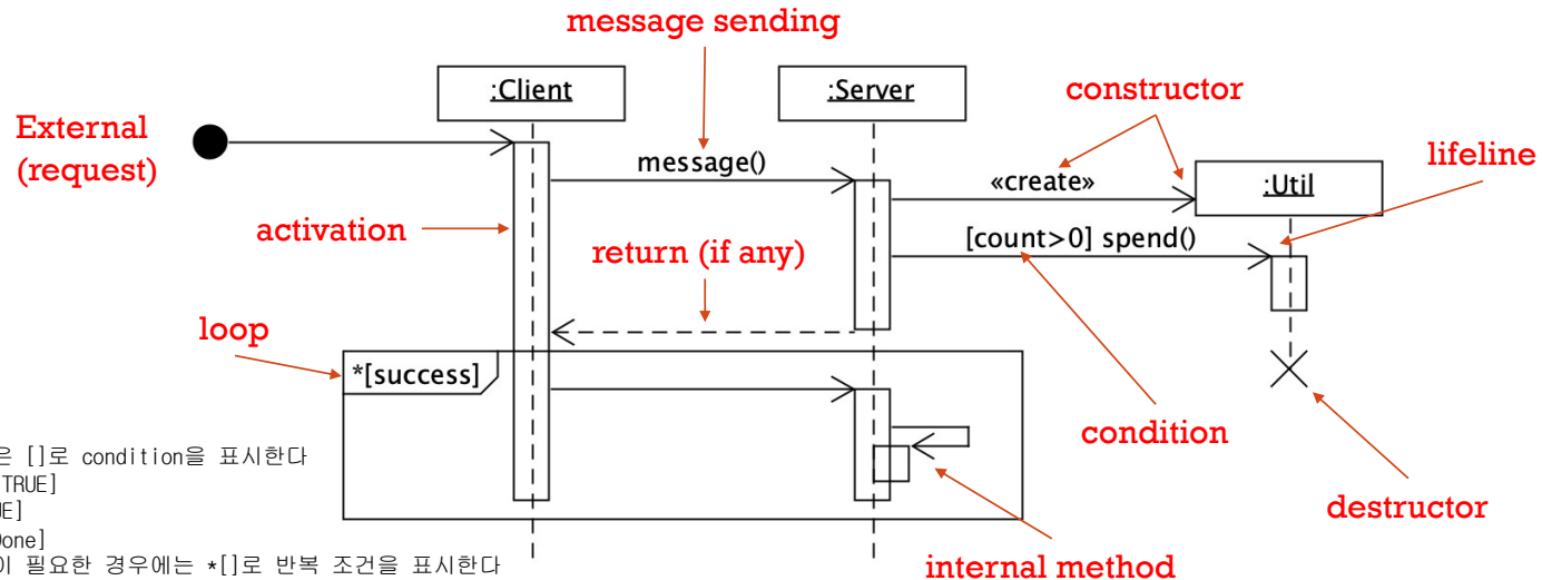
- **Class** 표기와 거의 동일하게 사각형으로 표시하지만,
  - **Instance**의 이름: **Class**의 이름으로 표기
  - 이름에 밑줄

또는

- 모서리가 둥근 사각형으로 표기하기도 함

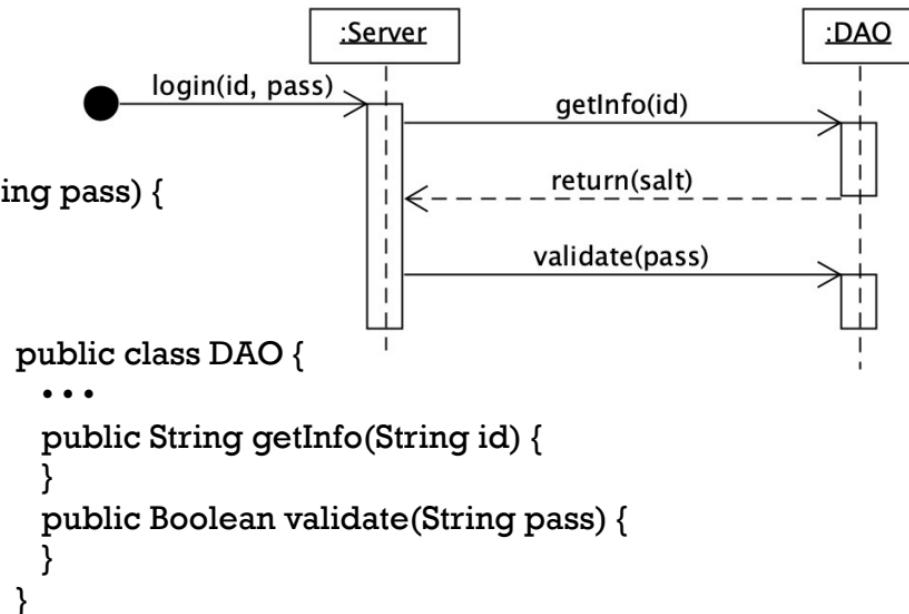


# KEY ELEMENTS



# TRANSLATE SEQUENCE DIAGRAM

```
public class Server {  
    private DAO myDAO;  
    ...  
    public Boolean login(String id, String pass) {  
        salt = myDAO.getInfo(id);  
        // modify pass with salt  
        return myDAO.validate(pass);  
    }  
    ...  
}
```



# PRACTICE: SEQUENCE DIAGRAM

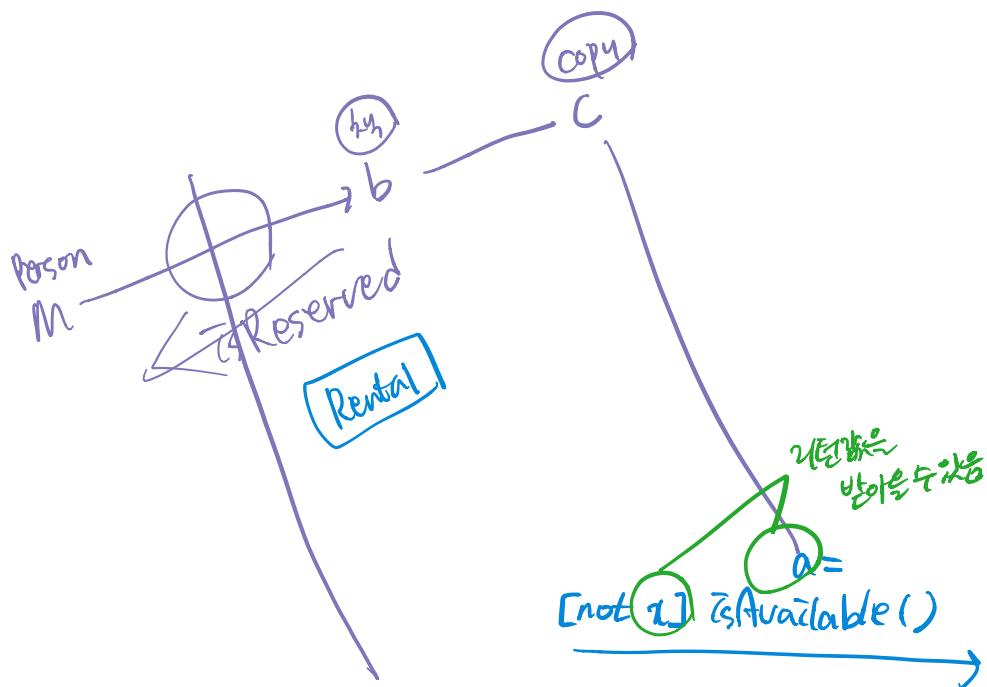
- 앞에서 만든 도서관 시스템 Class Diagram을 보고 아래 사건의 Sequence Diagram을 만드시오.

- 독서회원이 서가에 있는 책을 대출해 간다
- 독서회원이 서가에 없는 책을 예약한다
- 독서회원이 책을 반납한다

이름:	
의 특별한 경우	를 포함
필요지식	
처리임무	협력사항
결정사항/제약사항	



- 고객 → 회사
- Dealer → Deal → Company
- 고객 → 상품 → 장바구니



도서관  
책  
거치대

Rental

Rental 인스턴스  
(책, 빌려다섯)  
인증번호로 찾기

People  
Library  
Reservation  
Rental  
Book  
BookCopy

$x := \text{isReserved}()$   
 $[\text{not } x] a := \text{isAvailable}()$   
 $[\alpha = \text{true}] \ll \text{create} \gg$   
**checkout(r)**  
 r parameter를 통해  
 $\text{BookCopy} \rightarrow \text{Rental}$   
 접근 가능  
 $\text{recordCheckout}(m, c) \rightarrow \text{Rental}$

↓  
BookCopy에 있는  
HTTP://91.129.129.12  
만화책을 판권정보