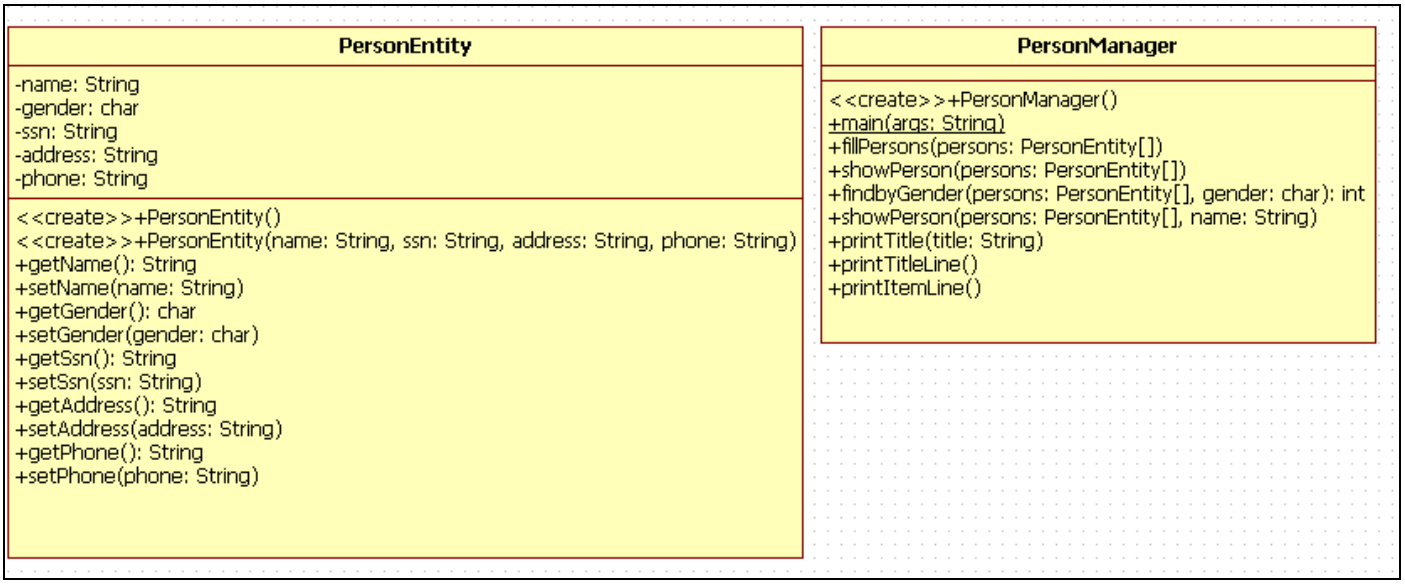
[실습1] Encapsulation 개념 연습하기

1. 클래스 이름은 Account로 합니다.
2. package명은 workshop.account.entity입니다.
3. 고객번호(custId), 계좌번호(acctId), 잔액(balance)에 해당하는 변수를 선언합니다.
   모든 변수들을 다른 클래스에서 직접 사용하지 못하도록 private으로 선언하며, 고객번호, 계좌번호는 String type으로 , 잔액은 int type으로 합니다.
4. 멤버변수 balance는 private으로 선언하였기 때문에 다른 클래스에서 이 멤버변수에 대해 직접 값을 읽거나 쓸 수 없습니다. 다른 클래스에서 잔액의 값을 할당하고, 읽을 수 있도록 메서드를 작성합니다.
   1) 값 할당 메서드
      메서드 이름 : setBalance, 아규먼트: int newBalance , 리턴타입: void
   2) 값 얻기 메서드
      메서드 이름 : getBalance, 아규먼트: 없음 , 리턴타입: int type
5. 멤버변수 custId와 acctId에 대해서도 값을 할당하고, 읽을 수 있는 메서드를 작성합니다.
   -setCustId, getCustId, setAcctId, getAcctId
6. 계좌잔액을 입력한 금액만큼 증가(입금) 시키는 메서드를 작성합니다.
   메서드이름 : deposit, 아규먼트: int amount , 리턴타입: void
7. 계좌잔액을 입력한 금액만큼 감소(출금) 시키는 메서드를 작성합니다.
   메서드이름 : withdraw, 아규먼트: int amount , 리턴타입: void
8. Account클래스를 객체 생성하여 이용하는 클래스인 TestAccount 클래스를 작성합니다.
   1) package는 workshop.account.test입니다.
   2) Account 클래스가 다른 package에 있기 때문에 import 해야 합니다.
   3) 실행 가능한 메서드인 main 메서드를 작성해야 합니다.
      a. Account 클래스의 객체를 생성합니다 (new 연산자를 사용함)
      b. 생성한 Account 객체의 멤버변수의 값을 아래와 같이 할당합니다.
         고객번호 : "A1100", 계좌번호 : "221-22-3477", 잔액 : 100000
      c. 고객번호와 계좌번호를 화면에 출력합니다.
      d. 잔액을 화면에 출력합니다.
      e. 잔액을 10000원 증가시킵니다.
      f. 잔액을 화면에 출력합니다.
      g. 잔액을 20000원 감소시킵니다.
      h. 잔액을 화면에 출력합니다.

작성자 : 백명숙(vega2k@naver.com)

[실습2] Encapsulation 개념, 배열 연습하기

1. 클래스 다이어그램



2. Sample Run

```
@@@ 인물 정보 조회 시스템 @@@
=============================================================
[이름] 이성호        [성별] 남       [전화번호] 032-392-2932
-------------------------------------------------------------
[이름] 김하늘        [성별] 여       [전화번호] 02-362-1932
-------------------------------------------------------------
[이름] 박영수        [성별] 남       [전화번호] 02-887-1542
-------------------------------------------------------------
[이름] 나인수        [성별] 남       [전화번호] 032-384-2223
-------------------------------------------------------------
[이름] 홍정수        [성별] 남       [전화번호] 02-158-7333
-------------------------------------------------------------
[이름] 이미숙        [성별] 여       [전화번호] 02-323-1934
-------------------------------------------------------------
[이름] 박성구        [성별] 남       [전화번호] 02-308-0932
-------------------------------------------------------------
[이름] 유성미        [성별] 여       [전화번호] 02-452-0939
-------------------------------------------------------------
[이름] 황재현        [성별] 남       [전화번호] 032-327-2202
-------------------------------------------------------------
[이름] 최철수        [성별] 남       [전화번호] 032-122-7832
-------------------------------------------------------------
성별 : '여 '(은)는 3명 입니다.
=============================================================
-- 이름 : '김하늘'(으)로 찾기 결과입니다. --
-------------------------------------------------------------
[이름] 김하늘
[성별] 여
[전화번호] 02-362-1932
[주소] 서울 강동구
```


작성자 : 백명숙(vega2k@naver.com)

3. PersonEntity.java : Person의 정보를 담는 클래스
   1) package명은 workshop.person.entity 입니다.
   2) 클래스 다이어그램을 참조하여 PersonEntity 클래스가 가져야 할 멤버변수들을 정의한다.
   3) PersonEntity() : Default Constructor를 정의한다.
   4) PersonEntity(String name, String ssn, String address, String phone)
      : 이름, 주민등록번호, 거주지주소, 거주지 전화번호에 해당하는 멤버변수들을 초기화 하는데 , 이때 5)번에서 작성하게 될, 각각의 멤버변수들과 관련있는 setter 메서드를 사용한다.
   5) 멤버변수들에 대한 getXXX() 메서드와 setXXX() 메서드를 작성한다.
   6) setSSN(String ssn) : 멤버변수 ssn의 setter 메서드
      : 전달받은 아규먼트 값으로 멤버 변수 ssn을 변경하고, 주민등록번호(ssn)의 6번째 글자가 '1' 또는 '3'이면 멤버변수 gender를 '남'으로 변경하고, '2' 또는 '4'이면 '여'로 변경한다.(조건 분기문(if) 를 사용하며, API에서 String 클래스의 charAt() 메서드를 참조)
4. PersonManager.java : Person의 정보를 담는 클래스
   1) package는 workshop.person.control 이다.
   2) PersonEntity 클래스를 import 한다.
   3) PersonManager() : Default Constructor 를 작성한다.
   4) main() : PersonEntity[ ] 객체들을 생성하여 화면에 출력한다.
      a. main() 메서드는 static 메서드 이므로 static이 아닌 메서드를 사용하기 위해 PersonManager pManager = new PersonManager() 객체를 생성한다.
      b. printTitle(String Title) , printTitleLine()을 이용해서 타이틀을 출력한다.
      c. PersonEntity[ ] persons = new PersonEntity[10]; 으로 PersonEntity 배열을 선언한다.
      d. fillPersons(PersonEntity[] persons)을 호출하여 persons에 정보를 set 한다.
      e. showPerson(PersonEntity[] persons)을 호출하여 전체 persons 정보를 display 한다.
      f. findByGender(PesonEntity[] persons, char gender)를 호출하여 persons 객체 정보 중 해당 gender의 인원수를 return 받는다. 이때 조건은 '여' 라는 성별로 검색한다.
      g. showPerson(PersonEntity[] persons, String name)을 호출하여 특정 person의 상세정보를 display 한다. 이때 이름 정보는 "김하늘" 로 한다.
   5) fillPersons(PersonEntity[] persons) : PersonEntity[]의 정보를 set 하는 메서드

```
new PersonEntity("이성호","7212121028102", "인천 계양구", "032-392-2932");
new PersonEntity("김하늘","7302132363217", "서울 강동구", "02-362-1932");
new PersonEntity("박영수","7503111233201", "서울 성북구", "02-887-1542");
new PersonEntity("나인수","7312041038988", "대전 유성구", "032-384-2223");
new PersonEntity("홍정수","7606221021341", "서울 양천구", "02-158-7333");
new PersonEntity("이미숙","7502142021321", "서울 강서구", "02-323-1934");
new PersonEntity("박성구","7402061023101", "서울 종로구", "02-308-0932");
new PersonEntity("유성미","7103282025101", "서울 은평구", "02-452-0939");
new PersonEntity("황재현","7806231031101", "인천 중구", "032-327-2202");
new PersonEntity("최철수","7601211025101", "인천 계양구", "032-122-7832");
```

   6) showPerson(PersonEntity[] persons) : PersonEntity[]의 정보를 출력하는 메서드. 아규먼트로 넘겨 받은

작성자 : 백명숙(vega2k@naver.com)

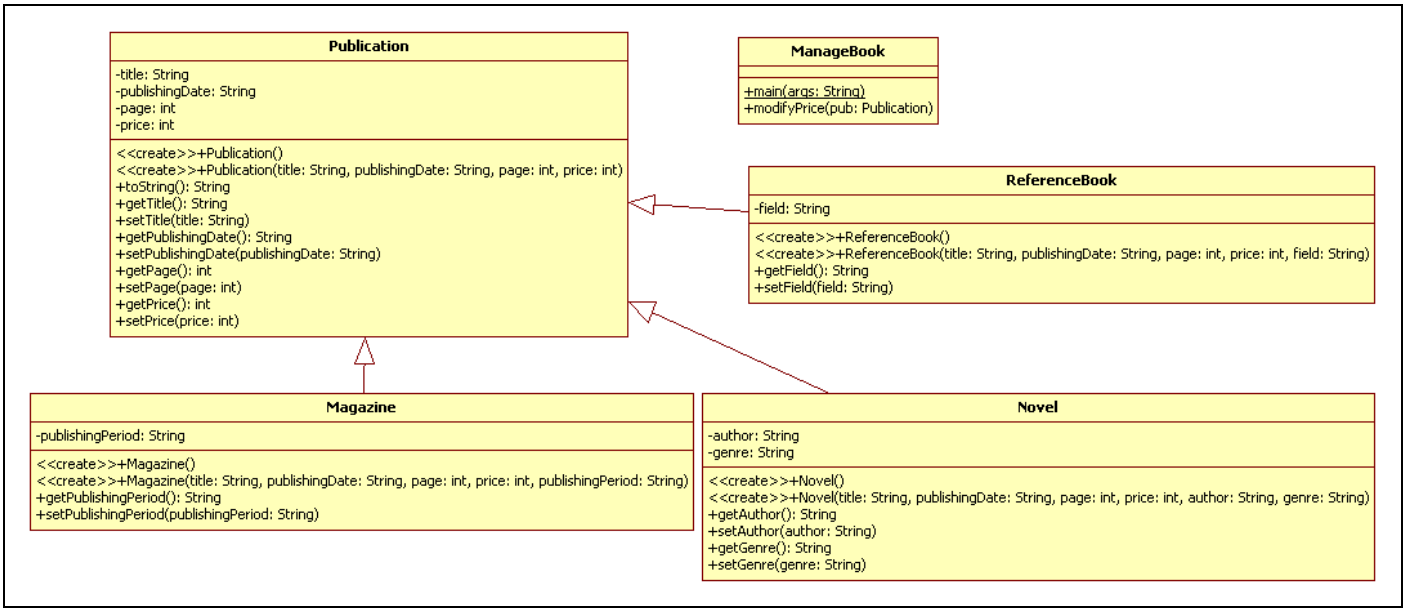
PersonEntity[] 객체를 이용하여, Sample Run을 참고하여 이름, 성별, 전화번호를 출력한다.

7) findByGender(PersonEntity[] persons, char gender) : PersonEntity[]의 정보 중 아규먼트로 넘어온 gender에 해당하는 인원수를 return 해준다.
8) showPerson(PersonEntity[]persons, String name) : PersonEntity[]의 정보 중 아규먼트의 name과 일치하는 person의 상세정보를 보여주는 메서드. 아규먼트로 넘겨받은 PersonEntity[] 객체 중 name과 일치하는(String 클래스의 equals() 메서드 사용) 객체의 이름, 성별, 전화번호, 주소를 Sample Run을 참고하여 display한다.
9) printTitle(String Title) : Title을 출력하는 메세드
   포맷은 "₩n" + title + "₩n" 이며 println()을 사용한다.
10) printTitleLine() : Title의 밑줄을 출력한다.
    Equal('=') 기호를 60 개 출력한다.
11) printItemLine() : Title의 밑줄을 출력한다.
    Equal('-') 기호를 60 개 출력한다.

작성자 : 백명숙(vega2k@naver.com)

[실습3] Encapsulation,Inheritance, Polymorphism 연습하기

1. 클래스 다이어그램



2. Sample Run

```
==== Book 정보 출력 ====
마이크로 소프트
경영과 컴퓨터
빠삐용
남한산성
실용주의 프로그래머
==== 가격정보 변경 전====
빠삐용 : 9800
==== 가격정보 변경 후====
빠삐용 : 7840
```

3. Publication.java : 출판물에 대한 클래스 (Super)

① package명은 workshop.book.entity 입니다.

② 생성자 : default 및 모든 멤버변수를 인자로 갖는 생성자를 작성한다.

③ getters / setters : 모든 멤버변수들에 대해 가각 getter와 setter 메서드를 작성한다.

④ toString() : 멤버변수 title을 return 하도록 한다.

4. Novel.java : 소설에 대한 클래스(Sub)

① package명은 workshop.book.entity 입니다.

② 생성자 : default 및 모든 멤버변수를 인자로 갖는 생성자를 작성한다.

③ getters / setters : 모든 멤버변수들에 대해 가각 getter와 setter 메서드를 작성한다.

5. Magazine.java : 잡지에 대한 클래스(Sub)

① package명은 workshop.book.entity 입니다.

② 생성자 : default 및 모든 멤버변수를 인자로 갖는 생성자를 작성한다.

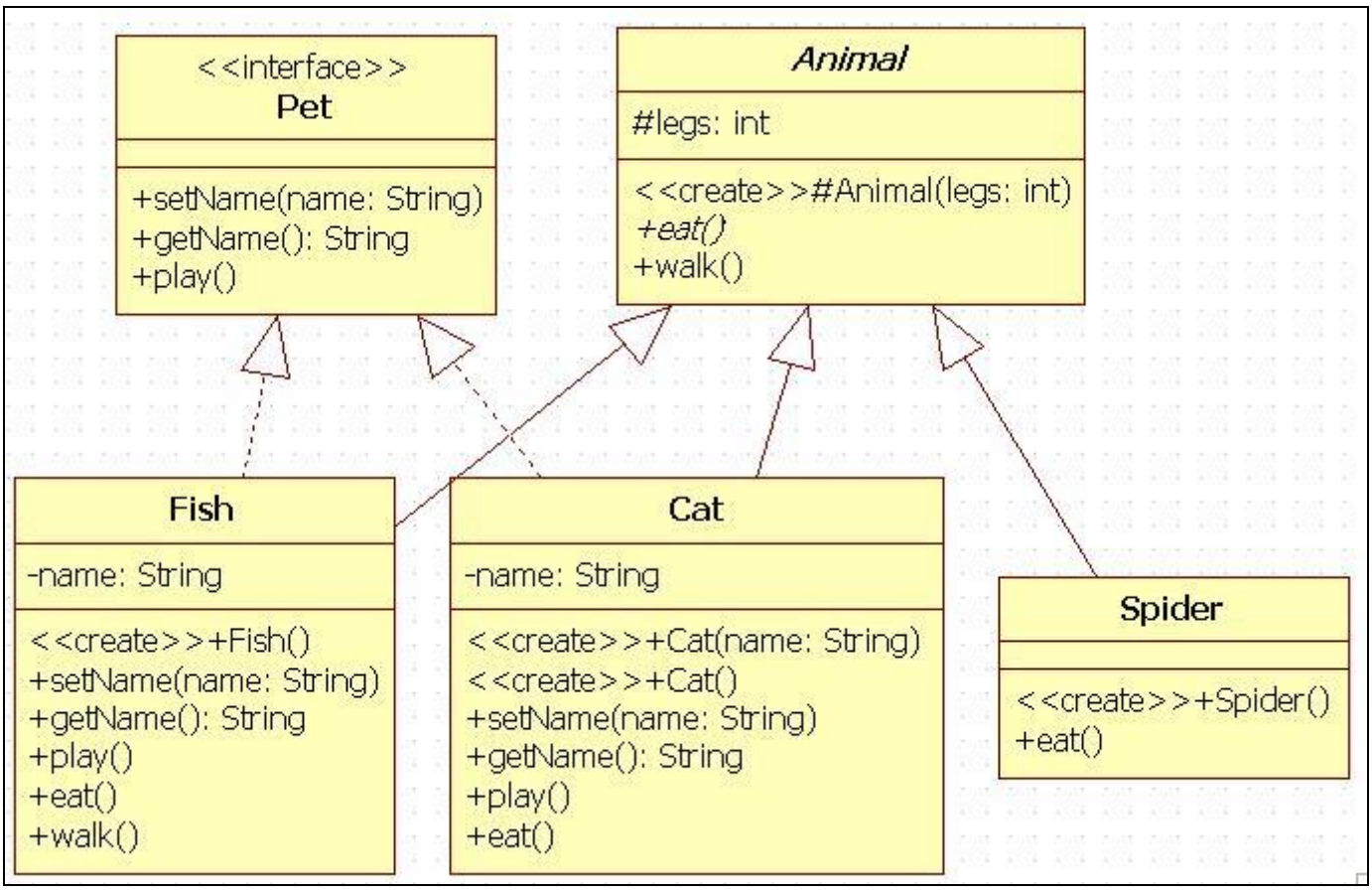
작성자 : 백명숙(vega2k@naver.com)

③ getters / setters : 모든 멤버변수들에 대해 가각 getter와 setter 메서드를 작성한다.

6. ReferenceBook.java : 소설에 대한 클래스(Sub)
   ① package명은 workshop.book.entity 입니다.
   ② 생성자 : default 및 모든 멤버변수를 인자로 갖는 생성자를 작성한다.
   ③ getters / setters : 모든 멤버변수들에 대해 가각 getter와 setter 메서드를 작성한다.
7. ManageBook.java : 도서정보를 관리하는 클래스
   ① package명은 workshop.book.control 입니다.
   ② main() 메서드
      a. 도서정보를 갖는 Publiction[] 배열을 생성한다. 이때 배열 타입은 Publication으로 하고 ,각 배열생성은 아래를 참고한다.
         new Magazine("마이크로소프트","2007-10-01",328,9900,"매월");
         new Magazine("경영과컴퓨터","2007-10-03",316,9000,"매월");
         new Novel("빠삐용","2007-07-01",396,9800,"베르나르베르베르","현대소설");
         new Novel("남한산성","2007-04-14",383,11000,"김훈","대하소설");
         new ReferenceBook("실용주의프로그래머","2007-01-14",496,25000,"소프트웨어공학");
      b. 반복문 (for loop)를 이용하여 각 도서에 대한 객체정보를 출력한다.(toString() 이용)
      c. 3번째 도서에 대한 가격을 가격변경 메서드를 이용하여 변경한다. 이때 변경 전 가격과 변경 후 가격을 화면에 출력한다.
      d. 모든 출력형태는 Sample Run을 참고한다.
   ③ modifyPrice() 메서드
      a. 아규먼트가 실제로 어떤 객체인지를 판별하여 (instanceof 연산자 사용), 가격정보를 Magazine이면 40%, Novel이면 20%, ReferenceBook 이면 10% 가격을 할인하여 저장한다.

작성자 : 백명숙(vega2k@naver.com)

[실습4] Abstract 클래스, Interface 연습하기



1. Create the Animal class, which is the abstract superclass of all animals.
    1. Declare a protected integer attribute called legs, which records the number of legs for this animal.
    2. Define a protected constructor that initializes the legs attribute.
    3. Declare an abstract method eat.
    4. Declare a concrete method walk that prints out something about how the animals walks (include the number of legs).
2. Create the Spider class.
    1. The Spider class extends the Animal class.
    2. Define a default constructor that calls the superclass constructor to specify that all spiders have eight legs.
    3. Implement the eat method.
3. Create the Pet interface specified by the UML diagram.
4. Create the Cat class that extends Animal and implements Pet.
    1. This class must include a String attribute to store the name of the pet.
    2. Define a constructor that takes one String parameter that specifies the cat's name. This constructor must also call the superclass constructor to specify that all cats have four legs.
    3. Define another constructor that takes no parameters. Have this constructor call the previous constructor (using the this keyword) and pass an empty string as the argument.
    4. Implement the Pet interface methods.

작성자 : 백명숙(vega2k@naver.com)

5. Implement the eat method.

5. Create the Fish class. Override the Animal methods to specify that fish can't walk and don't have legs.
6. Create an TestAnimals program. Have the main method create and manipulate instances of the classes you created above. Start with:

```
Fish d = new Fish();
Cat c = new Cat( “Fluffy” );
Animal a = new Fish();
Animal e = new Spider();
Pet p = new Cat();
```

Experiment by: a) calling the methods in each object, b) casting objects, c) using polymorphism, and d) using super to call super class methods.

작성자 : 백명숙(vega2k@naver.com)