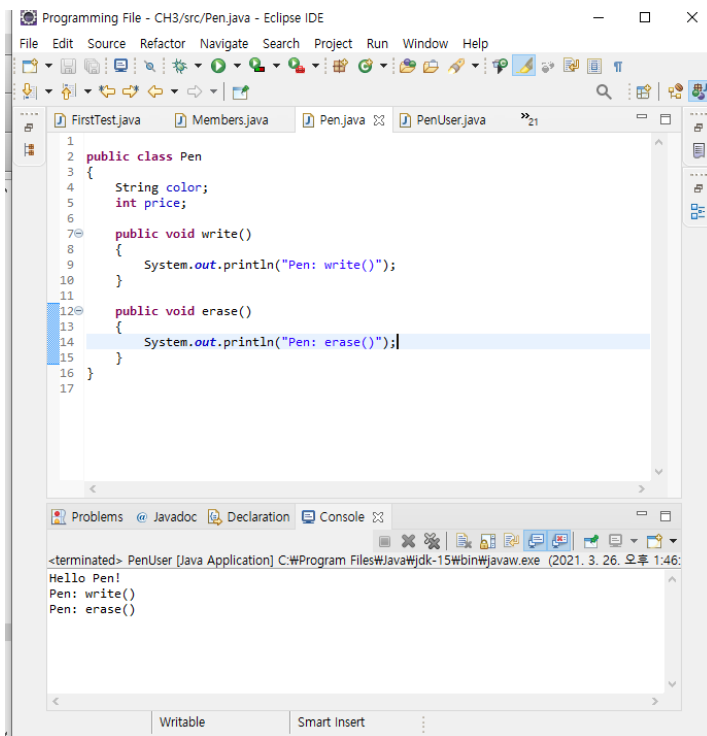


객체지향프로그래밍 4주차 정리

프로그래밍

Lab 1-1

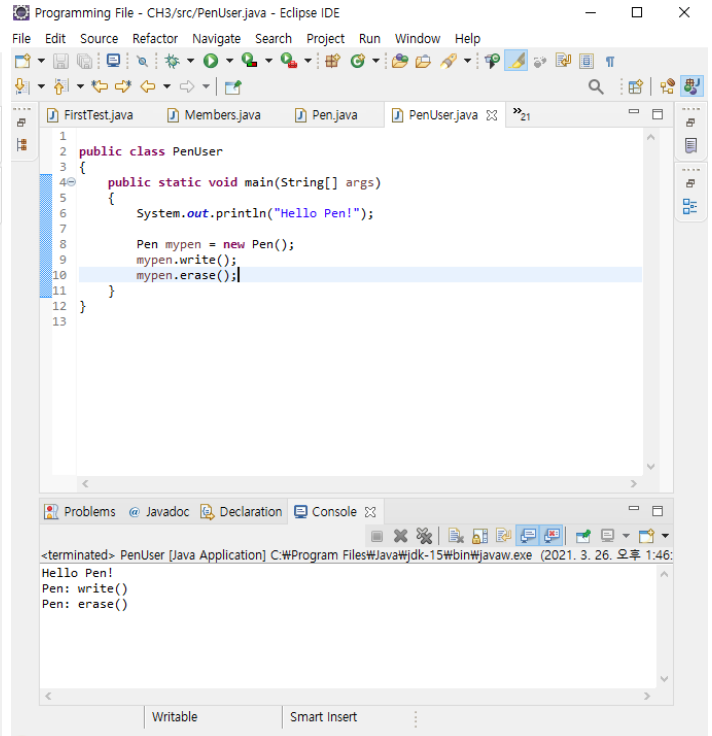


Programming File - CH3/src/Pen.java - Eclipse IDE

```
1 public class Pen
2 {
3     String color;
4     int price;
5
6     public void write()
7     {
8         System.out.println("Pen: write()");
9     }
10
11     public void erase()
12     {
13         System.out.println("Pen: erase()");
14     }
15 }
16
17
```

Console:

```
<terminated> PenUser [Java Application] C:\Program Files\Java\jdk-15\bin\javaw.exe (2021. 3. 26. 오후 1:46:
Hello Pen!
Pen: write()
Pen: erase()
```



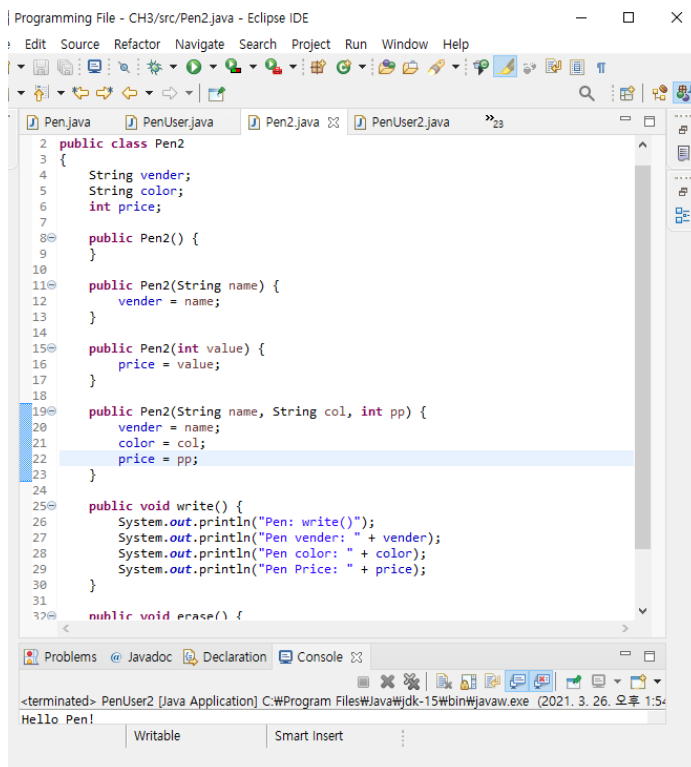
Programming File - CH3/src/PenUser.java - Eclipse IDE

```
1
2 public class PenUser
3 {
4     public static void main(String[] args)
5     {
6         System.out.println("Hello Pen!");
7
8         Pen mypen = new Pen();
9         mypen.write();
10        mypen.erase();
11    }
12 }
13
```

Console:

```
<terminated> PenUser [Java Application] C:\Program Files\Java\jdk-15\bin\javaw.exe (2021. 3. 26. 오후 1:46:
Hello Pen!
Pen: write()
Pen: erase()
```

Lab1-2

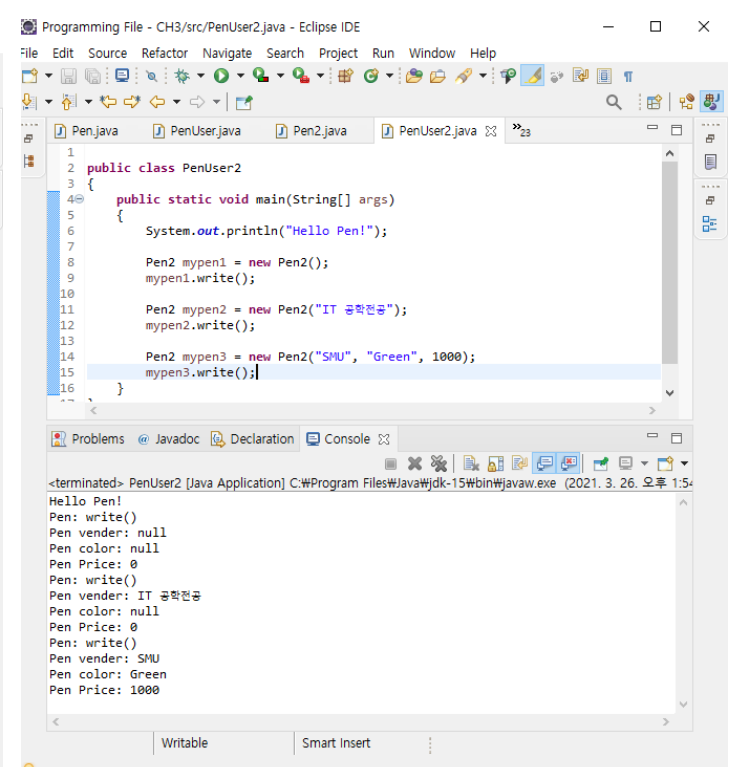


Programming File - CH3/src/Pen2.java - Eclipse IDE

```
1
2 public class Pen2
3 {
4     String vender;
5     String color;
6     int price;
7
8     public Pen2() {
9     }
10
11     public Pen2(String name) {
12         vender = name;
13     }
14
15     public Pen2(int value) {
16         price = value;
17     }
18
19     public Pen2(String name, String col, int pp) {
20         vender = name;
21         color = col;
22         price = pp;
23     }
24
25     public void write() {
26         System.out.println("Pen: write()");
27         System.out.println("Pen vender: " + vender);
28         System.out.println("Pen color: " + color);
29         System.out.println("Pen Price: " + price);
30     }
31
32     public void erase() {
33     }
34 }
```

Console:

```
<terminated> PenUser2 [Java Application] C:\Program Files\Java\jdk-15\bin\javaw.exe (2021. 3. 26. 오후 1:54:
Hello Pen!
Pen: write()
Pen vender: null
Pen color: null
Pen Price: 0
Pen: write()
Pen vender: IT 공학전공
Pen color: null
Pen Price: 0
Pen: write()
Pen vender: SMU
Pen color: Green
Pen Price: 1000
```



Programming File - CH3/src/PenUser2.java - Eclipse IDE

```
1
2 public class PenUser2
3 {
4     public static void main(String[] args)
5     {
6         System.out.println("Hello Pen!");
7
8         Pen2 mypen1 = new Pen2();
9         mypen1.write();
10
11         Pen2 mypen2 = new Pen2("IT 공학전공");
12         mypen2.write();
13
14         Pen2 mypen3 = new Pen2("SMU", "Green", 1000);
15         mypen3.write();
16     }
17 }
```

Console:

```
<terminated> PenUser2 [Java Application] C:\Program Files\Java\jdk-15\bin\javaw.exe (2021. 3. 26. 오후 1:54:
Hello Pen!
Pen: write()
Pen vender: null
Pen color: null
Pen Price: 0
Pen: write()
Pen vender: IT 공학전공
Pen color: null
Pen Price: 0
Pen: write()
Pen vender: SMU
Pen color: Green
Pen Price: 1000
```

Lab1-3

```

Programming File - CH3/src/Pen3.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help

Pen3.java Pen2.java PenUser2.java PenUser3.java »24
1 public Pen3(String name, String col, int pp) {
2     vender = name;
3     color = col;
4     price = pp;
5 }
6
7 void write() {
8     System.out.println("Pen: write()");
9     System.out.print("Pen Vender: " + vender);
10    System.out.print(", Pen Color: " + color);
11    System.out.print(", Price: " + price);
12 }
13
14 void write(int xx) {
15     System.out.println("Pen: write(int xx)");
16     System.out.print("Pen Vender: " + vender);
17     System.out.print(", Pen Color: " + color);
18     System.out.print(", Price: " + xx);
19 }
20
21 void write(int xx, String yy) {
22     System.out.println("Pen: write(int, String)");
23     System.out.print("Pen Vender: " + yy);
24     System.out.print(", Pen Color: " + color);
25     System.out.print(", Price: " + xx);
26 }
27
28 void write(int xx, String yy, String zz) {
29     System.out.println("Pen: write(int, String, String)");
30     System.out.print("Pen Vender: " + yy);
31     System.out.print(", Pen Color: " + zz);
32     System.out.print(", Price: " + xx);
33 }
34
35
36
37
38
39
40
41
42
43
44
45
46

```

```

Programming File - CH3/src/PenUser3.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help

Pen3.java Pen2.java PenUser2.java PenUser3.java »24
3 {
4     public static void main(String[] args)
5     {
6         System.out.println("Hello Pen!");
7
8         Pen3 mypen1 = new Pen3();
9         mypen1.write();
10        mypen1.write(10000);//price
11        mypen1.write(1000, "Red");//price, color
12
13        Pen3 mypen2 = new Pen3("IT공학과");//vender 지정
14        mypen2.write();
15        mypen2.write(20000);//price
16        mypen2.write(2000, "SMU2");//price vender
17
18        Pen3 mypen3 = new Pen3("IT공학과", "노란색", 30000);//3가지 지정
19        mypen3.write();
20        mypen3.write(20000);//price
21    }
22 }

```

Problems @ Javadoc Declaration Console

```

<terminated> PenUser3 [Java Application] C:\Program Files\Java\jdk-15\bin\javaw.exe (2021. 3. 26. 오후 2:11)
Hello Pen!
Pen: write()
Pen Vender: null, Pen Color: null, Price: 0Pen: write(int xx)
Pen Vender: null, Pen Color: null, Price: 10000Pen: write(int, String)
Pen Vender: Red, Pen Color: null, Price: 1000Pen: write()
Pen Vender: IT공학과, Pen Color: null, Price: 0Pen: write(int xx)
Pen Vender: IT공학과, Pen Color: null, Price: 20000Pen: write(int, String)
Pen Vender: SMU2, Pen Color: null, Price: 20000Pen: write()
Pen Vender: IT공학과, Pen Color: 노란색, Price: 30000Pen: write(int xx)
Pen Vender: IT공학과, Pen Color: 노란색, Price: 20000Pen: write(int, String)
Pen Vender: SMU3, Pen Color: 노란색, Price: 3000Pen: write(int, String, String)
Pen Vender: SMU3, Pen Color: Yello, Price: 3000

```

Lab1-4

```

Microsoft Windows [Version 10.0.19041.87]
(c) 2020 Microsoft Corporation. All rights reserved.

C:\Users\양원서>cd C:\Users\양원서\Desktop\제4회\Programming File\CH3\src
C:\Users\양원서\Desktop\제4회\Programming File\CH3\src>javac -d . Pen4.java
C:\Users\양원서\Desktop\제4회\Programming File\CH3\src>javac -d . Pen4.java
C:\Users\양원서\Desktop\제4회\Programming File\CH3\src>javac -d . Pen4.java
C:\Users\양원서\Desktop\제4회\Programming File\CH3\src>java Pen4
Error: Could not find or load main class Pen4
Caused by: java.lang.ClassNotFoundException: Pen4

C:\Users\양원서\Desktop\제4회\Programming File\CH3\src>javac PenUser4.java
PenUser4.java:1: error: package PenP does not exist
import PenP.Pen4;
^
1 error

C:\Users\양원서\Desktop\제4회\Programming File\CH3\src>java PenUser4
PenUser4.java:9: error: cannot access Pen4
    Pen4 mypen = new Pen4("SMU", "Green", 1000);
    ^
  bad source file: .\Pen4.java
    file does not contain class Pen4
  Please remove or make sure it appears in the correct subdirectory of the source path.
2 errors

C:\Users\양원서\Desktop\제4회\Programming File\CH3\src>javac PenUser4.java
PenUser4.java:1: error: package PenP does not exist
import PenP.Pen4;
^
1 error

C:\Users\양원서\Desktop\제4회\Programming File\CH3\src>java PenUser4
PenUser4.java:9: error: cannot access Pen4
    Pen4 mypen = new Pen4("SMU", "Green", 1000);
    ^
  bad source file: .\Pen4.java
    file does not contain class Pen4
  Please remove or make sure it appears in the correct subdirectory of the source path.
2 errors

C:\Users\양원서\Desktop\제4회\Programming File\CH3\src>javac PenUser4.java
PenUser4.java:1: error: package PenP does not exist
import PenP.Pen4;
^
1 error

C:\Users\양원서\Desktop\제4회\Programming File\CH3\src>java PenUser4
Hello Pen!
Pen: write()
Pen Vender: SMU
Pen Color: Green
Pen Price: 1000
Pen: write()
Pen Vender: SMU
Pen Color: Green
Pen Price: 2000
Pen: write()
Pen Vender: Dept. IT Engineering
Pen Color: Green
Pen Price: 3000

```

```

Programming File - CH3/src/PenUser4.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help

Pen3.java Pen4.java PenUser3.java PenUser4.java »25
1 import PenP.Pen4;
2
3 public class PenUser4
4 {
5     public static void main(String [] args)
6     {
7         System.out.println("Hello Pen!");
8
9         Pen4 mypen = new Pen4("SMU", "Green", 1000);
10
11        mypen.erase();
12        mypen.write();
13        mypen.write(2000);
14        mypen.write(3000, "Dept. IT Engineering");
15    }
16 }
17

```

```

Programming File - CH3/src/Pen4.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help

Pen3.java Pen4.java PenUser3.java PenUser4.java »25
1 package PenP;
2
3 public class Pen4
4 {
5     String vender;
6     String color;
7     int price;
8
9     public Pen4() {}
10
11
12     public Pen4(String name) {
13         vender = name;
14     }
15
16     public Pen4(String name, String col, int pp) {
17         vender = name;
18         color = col;
19         price = pp;
20     }
21
22     public void write() {
23         System.out.println("Pen: write()");
24         System.out.print("Pen Vender: " + vender);
25         System.out.print(", Pen Color: " + color);
26         System.out.print(", Pen Price: " + price);
27     }
28
29     public void write(int xx) {
30         System.out.println("Pen: write()");
31         System.out.print("Pen Vender: " + vender);
32         System.out.print(", Pen Color: " + color);
33     }
34 }

```

- PenP
- Members.java
- Pen.java
- Pen2.java
- Pen3.java
- Pen4.java
- PenUser.java
- PenUser2.java
- PenUser3.java
- PenUser4.class
- PenUser4.java

Lab 2

Programming File - CH3/src/School.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

```
Members.java Pen.java Teacher.java School.java 26
9      yiyoon = new Teacher("Yoon", "M10313", "IT공학전공");
10     kim = new Members("Kim", "IT200324");
11     song = new Members("song", "IT190324");
12     choi = new Members("choi", "IT180324");
13
14     students = new Members[4];
15     students[0] = yiyoon;
16     students[1] = kim;
17     students[2] = song;
18     students[3] = choi;
19
20
21     public void makeWork() {
22         int n = students.length;
23         for (int i = 0; i < n; i++)
24         {
25             students[i].work();
26         }
27     }
28
29     public static void main(String [] args) {
30         School mycom = new School();
31         mycom.makeWork();
32     }
```

Problems Javadoc Declaration Console

```
<terminated> School [Java Application] C:\Program Files\Java\jdk-15\bin\javaw.exe (2021. 3. 27. 오전 9:11:2)
Teacher: "Yoon" studies hard with is students is IT공학전공dept.
Student: "Kim" does his best.
Student: "song" does his best.
Student: "choi" does his best.
```

Writable

Smart Insert

Programming File - CH3/src/Teacher.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

```
Members.java Pen.java Teacher.java School.java 26
1 class Teacher extends Members
2 {
3     String dept;
4     Members students[];
5
6     public Teacher(String name, String idnum, String dept)
7     {
8         super(name, idnum);
9         this.dept = dept;
10    }
11
12    public void setStudents(Members sub[])
13    {
14        students = sub;
15    }
16
17    public void work()
18    {
19        System.out.println("Teacher: \"\" + name + \"\" studies hard with is stud
20        + dept + "dept.");
21    }
22 }
23
```

Problems Javadoc Declaration Console

```
<terminated> School [Java Application] C:\Program Files\Java\jdk-15\bin\javaw.exe (2021. 3. 27. 오전 9:11:2)
Teacher: "Yoon" studies hard with is students is IT공학전공dept.
Student: "Kim" does his best.
Student: "song" does his best.
Student: "choi" does his best.
```

Writable

Smart Insert

Programming File - CH3/src/Members.java - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

```
Members.java Pen.java Teacher.java School.java 26
1 public class Members
2 {
3     String name;
4     String dept;
5     String major;
6     int id;
7
8     public Members(String name, String dept) {
9         this.name = name;
10        this.dept = dept;
11    }
12
13
14    public void setId(int id) {
15        this.id = id;
16    }
17
18    public void setMajor(String major) {
19        this.major = major;
20    }
21
22    public void work() {
23        System.out.println("\tStudent: \"\" + name + \"\" does his best.");
24    }
25 }
```

Problems Javadoc Declaration Console

```
<terminated> School [Java Application] C:\Program Files\Java\jdk-15\bin\javaw.exe (2021. 3. 27. 오전 9:11:2)
Teacher: "Yoon" studies hard with is students is IT공학전공dept.
Student: "Kim" does his best.
Student: "song" does his best.
Student: "choi" does his best.
```

Writable

Smart Insert

Lab2_students class

Programming File - CH3/src/School_S.java - Eclipse IDE

```
File Edit Source Refactor Navigate Search Project Run Window Help
School.java Members_S.java Teacher_S.java School_S.java »30
1 public class School_S {
2     Teacher_S yiyoon;
3     Students kim, song, choi, lee;
4
5
6
7
8     public School_S() {
9         yiyoon = new Teacher_S("Yoon", "M10313", "IT공학전공");
10        kim = new Students("Kim", "IT200324");
11        song = new Students("song", "IT190324");
12        choi = new Students("choi", "IT180324");
13    }
14
15    public void makeWork() {
16        yiyoon.work();
17        kim.work();
18        song.work();
19        choi.work();
20    }
21
22    public static void main(String [] args) {
23        School mycom = new School_S();
24        mycom.makeWork();
25    }
26 }
```

Problems Javadoc Declaration Console

<terminated> School_S [Java Application] C:\Program Files\Java\jdk-15\bin\javaw.exe (2021. 3. 27. 오전 9:51)

Teacher: "Yoon" studies hard with is students is IT공학전공dept.
Student: "Kim" does his best.
Student: "song" does his best.
Student: "choi" does his best.

Programming File - CH3/src/Members_S.java - Eclipse IDE

```
File Edit Source Refactor Navigate Search Project Run Window Help
School.java Members_S.java Teacher_S.java School_S.java »30
1 public class Members_S {
2     String name; //이름
3     String dept; //과+학번
4     String major; //전공
5     int id;
6
7
8     public Members_S(String name, String dept) {
9         this.name = name;
10        this.dept = dept;
11    }
12
13    public void setId(int id) {
14        this.id = id;
15    }
16
17    public void setMajor(String major) {
18        this.major = major;
19    }
20
21    public void work() {
22        System.out.println("\tStudent: \"" + name + "\" does his best.");
23    }
24 }
25 }
```

Problems Javadoc Declaration Console

<terminated> School_S [Java Application] C:\Program Files\Java\jdk-15\bin\javaw.exe (2021. 3. 27. 오전 9:51)

Teacher: "Yoon" studies hard with is students is IT공학전공dept.
Student: "Kim" does his best.
Student: "song" does his best.
Student: "choi" does his best.

Programming File - CH3/src/Students.java - Eclipse IDE

```
File Edit Source Refactor Navigate Search Project Run Window Help
Students.java Members_S.java Teacher_S.java School_S.java »30
1 public class Students extends Members_S
2 {
3
4
5     public Students(String name, String dept)
6     {
7         super(name, dept);
8         this.name = name;
9         this.dept = dept;
10    }
11
12    public void work() {
13        System.out.println("\tStudent: \"" + name + "\" does his best.");
14    }
15 }
16 }
```

Problems Javadoc Declaration Console

<terminated> School_S [Java Application] C:\Program Files\Java\jdk-15\bin\javaw.exe (2021. 3. 27. 오전 9:51)

Teacher: "Yoon" studies hard with is students is IT공학전공dept.
Student: "Kim" does his best.
Student: "song" does his best.
Student: "choi" does his best.

Programming File - CH3/src/Teacher_S.java - Eclipse IDE

```
File Edit Source Refactor Navigate Search Project Run Window Help
School.java Members_S.java Teacher_S.java School_S.java »30
1 public class Teacher_S extends Members_S {
2     String dept;
3     Students student;
4
5
6     public Teacher_S(String name, String idnum, String dept)
7     {
8         super(name, idnum);
9         this.dept = dept;
10    }
11
12    public void setStudents(String name, String dept)
13    {
14        student = new Students(name, dept);
15    }
16
17    public void work()
18    {
19        System.out.println("Teacher:\"" + name + "\" studies hard with is stude
20        + dept +\"dept.\"");
21    }
22
23 }
```

Problems Javadoc Declaration Console

<terminated> School_S [Java Application] C:\Program Files\Java\jdk-15\bin\javaw.exe (2021. 3. 27. 오전 9:51)

Teacher: "Yoon" studies hard with is students is IT공학전공dept.
Student: "Kim" does his best.
Student: "song" does his best.
Student: "choi" does his best.

컴포넌트

= 하나의 클래스 with 서비스를 위한 메소드

메소드를 잘 짜는 것이 목표

HIPO

(Hierarchical Input Process Output model)

소프트웨어의 기본

시스템 분석 설계 지원 및 문서화 기법

시스템의 모듈을 계층으로 표현하고 각 모듈을 문서화하는데 사용되었다

요구사항을 개발, 설계를 구성, 자동화된 랑데부를 보여주는 전문 시스템의 완성을 돕기 위해 사용했고, 그 후에는 설계 및 구현 방법으로 인해 검증이 체계적으로 수행되었다.

시스템 전체 설계는 HIPO 차트 또는 구조차트를 사용하여 문서화

구조차트

시스템을 관리가능한 최저 수준으로 분류한 차트, 구조화된 프로그래밍에서 프로그램 모듈을 트리로 배열하는데 사용

트리 구조

모듈 간의 관계를 시각화한다

트리구조는 구조를 계층적으로 표현하는 방법이다.

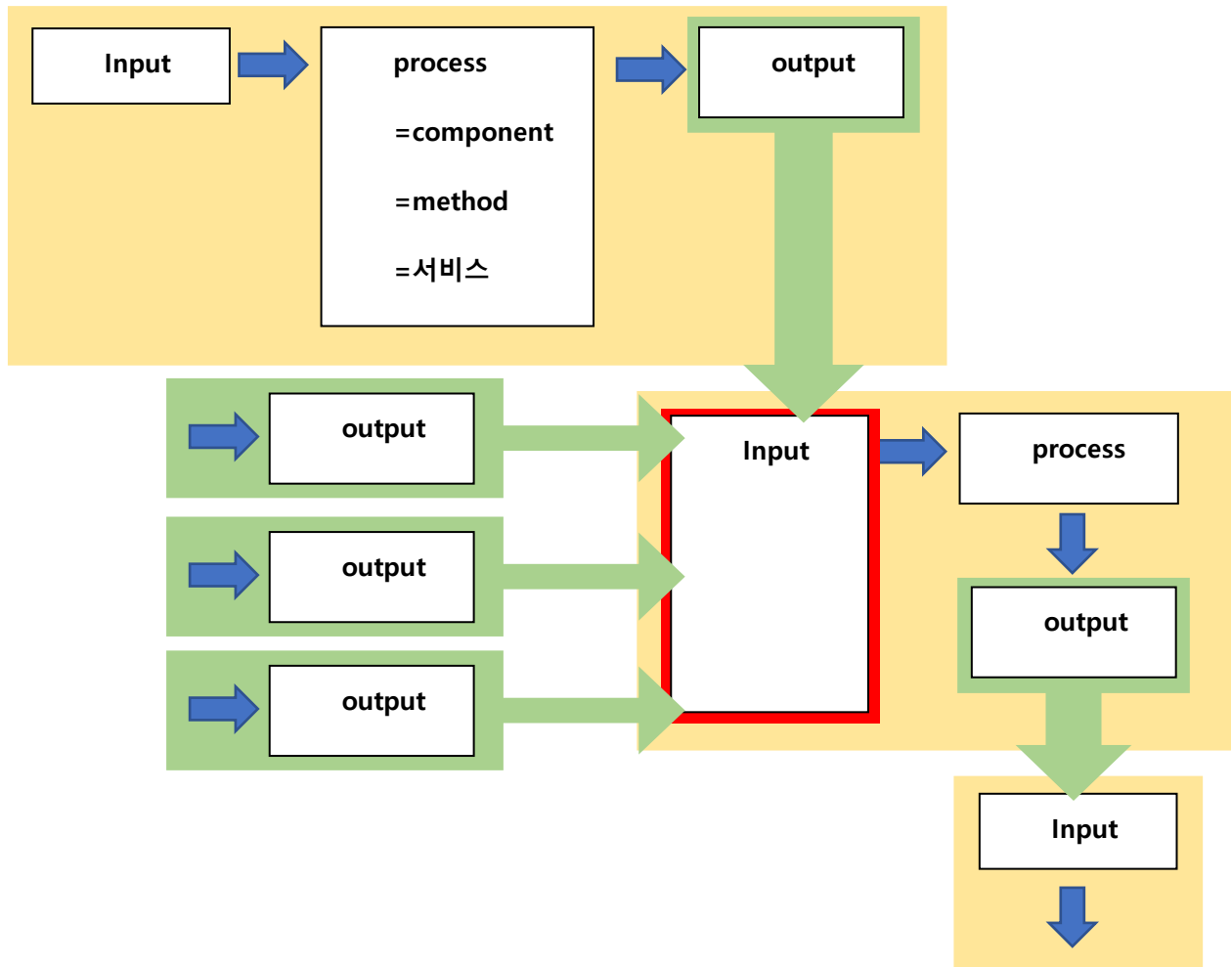
표현 방식이 실제 나무를 닮았다고 해서 트리 구조가 되었다.

(뿌리가 위 줄기가 아래에 위치)

Computer science 에서 트리는 다양하게 사용되는 계층적인 추상 데이터 타입이다

(출처 : [https://en.wikipedia.org/wiki/Tree_\(data_structure\)](https://en.wikipedia.org/wiki/Tree_(data_structure)),
https://en.wikipedia.org/wiki/Tree_structure)

(출처 : https://en.wikipedia.org/wiki/HIPO_model)



CBD

(Component Based Design / Development)

기존의 시스템이나 소프트웨어를 구성하는 컴포넌트를 조립해서 하나의 새로운 응용 프로그램을 만드는 소프트웨어 개발방법론

Ex) 기업들은 쇼핑 바구니, 사용자 인증, 검색엔진, 카탈로그 등 상업적으로 이용 가능한 컴포넌트를 결합하여 그들의 전자상거래 응용 프로그램을 개발하는 컴포넌트 기반 개발을 사용한다

(출처:

[https://ko.wikipedia.org/wiki/%EC%BB%B4%ED%8F%AC%EB%84%8C%ED%8A%B8_%EA%B8%B0%EB%B0%98_%EC%86%8C%ED%94%84%ED%8A%B8%EC%9B%A8%EC%96%B4_%EA%B3%B5%ED%95%99\)](https://ko.wikipedia.org/wiki/%EC%BB%B4%ED%8F%AC%EB%84%8C%ED%8A%B8_%EA%B8%B0%EB%B0%98_%EC%86%8C%ED%94%84%ED%8A%B8%EC%9B%A8%EC%96%B4_%EA%B3%B5%ED%95%99))

>>클래스를 어떻게 만드느냐가 중요(클래스 정의)

Object : 현실에 존재하는 객체



개념 설계 단계

Abstraction : 추상화, 공통된 특성 파악, 불필요한 특성 제거

굉장히 중요하고 어려운 단계

프로그램을 만들 때 코딩이 중요한 것이 아니라 무엇을 프로그래밍할 지를 결정하는 것이 더 중요한데 이를 Abstraction 단계에서 진행한다

공통된 특징을 뽑아 일반화하는 단계

Target 이 있어야 함

동일한 객체: 실제 세계의 사물을 일반화

동일한 행위: 적용할 함수를 일반화

동일한 속성: 사용할 데이터 일반화

객체 -(추상화)-> 일반화

개념 정의 단계

Class : 추상화를 통해 공통된 특성을 모은다

Member Field(attributes) : 객체들이 가진 공통적 특성

Member Function : 객체들의 공통된 행위, 상태

Class Instance

정의된 클래스를 생성자를 통해 만들어낸 객체

객체를 만들어야 클래스를 사용하는 것이라고 할 수 있다

클래스 정의 > 객체를 만들지 않는다면 의미가 없다

Method invocation

다른 클래스에 있는 메소드를 호출할 수 있다

★객체가 만들어져 있어야 한다

>>message passing 에 기반(ECA rule)

컴퓨터 과학에서 메시지 전달은 컴퓨터에서 동작(프로그램 실행)을 호출하는 기술

해당 프로세스와 지원 인프라(JVM)에 의존하여 적절한 코드를 선택하고 실행한다

Signature 을 전달한다

Message passing 은 객체지향프로그래밍 모델의 핵심

메시지가 발생한 다른 위치에서 다른 운영 체제와 프로그래밍 언어를 사용하여 프로세스를 찾는다

메시지를 처리하는데 적합한 개체가 현재 실행 중이 아닌 경우, 큐에 메시지를 저장한 다음 개체를 사용할 수 있을 때 메시지를 호출한다

필요한 경우 전송 객체가 수신할 준비가 될 때까지 결과를 저장한다

(출처 : https://en.wikipedia.org/wiki/Message_passing)

Encapsulation :

Class 만들 때
따라온다

객체지향프로그래밍에서 캡슐화(Encapsulation)은 데이터 상에서 method 등의 구성 요소들의 묶음을 의미한다.

이를 통해 외부에서 해당 클래스의 직접적인 접근을 막는다

클래스 내에서 객체의 값과 method 를 숨기는데 사용되어 숨겨진 세부 구현 정보를 숨기고 해당 클래스를 변화시키는 것을 막는다

공개적으로 액세스 가능한 메소드는 일반적으로 클래스에 제공되어 상태를 보다 추상적으로 액세스하거나 수정할 수 있다

소프트웨어 개체가 해당 서비스가 구현되는 방식을 알거나 신경 쓰지 않고 다른 개체에서 서비스를 호출할 수 있도록 한다

코딩의 logic 의 양을 줄이고 시스템을 유지 관리할 수 있도록 도와준다

(출처 : [https://en.wikipedia.org/wiki/Encapsulation_\(computer_programming\)](https://en.wikipedia.org/wiki/Encapsulation_(computer_programming)))

Class 만들 때
따라온다

Information Hiding : 정보 은닉

Class 만들 때
필요하면 사용

외부에서 member fields, methods 에 접근하는 것을 막는다

public, private, protected 키워드 사용

Inheritance : 상속

새로운 클래스를 만들 때 존재하는 다른 클래스의 속성을 활용하는 것

다른 클래스의 변수나 함수를 활용할 수 있도록 한다

불필요한 중복을 방지한다

클래스를 확장하고 싶을 때 사용한다

java 에서는 기본적으로 단일상속밖에 되지 않는다(single inheritance)>> logic 을 잘
짜야 한다

```
class ClassName extends ParentsClass
```

implement 를 통해 다중상속이 가능하다

```
class ClassName extends ParentsClass, implements InterfaceClass
```

상속을 통해 멤버 부모 클래스와 인터페이스의 변수와 메소드까지 확장할 수 있다

Is-a Relationship

이 객체는 그 객체의 타입이라는 의미

의미, 속성, 분류의 계층적인 개념

계층화될 수 있는 구조

Ex)

```
Public class Animal{
```

```
}
```

```
Public class Mammal extends Animal{
```

```
}
```

```
Public class Reptile extends Animal{
```

```
}
```

```
Public class Dog extends Mammal{
```

```
}
```

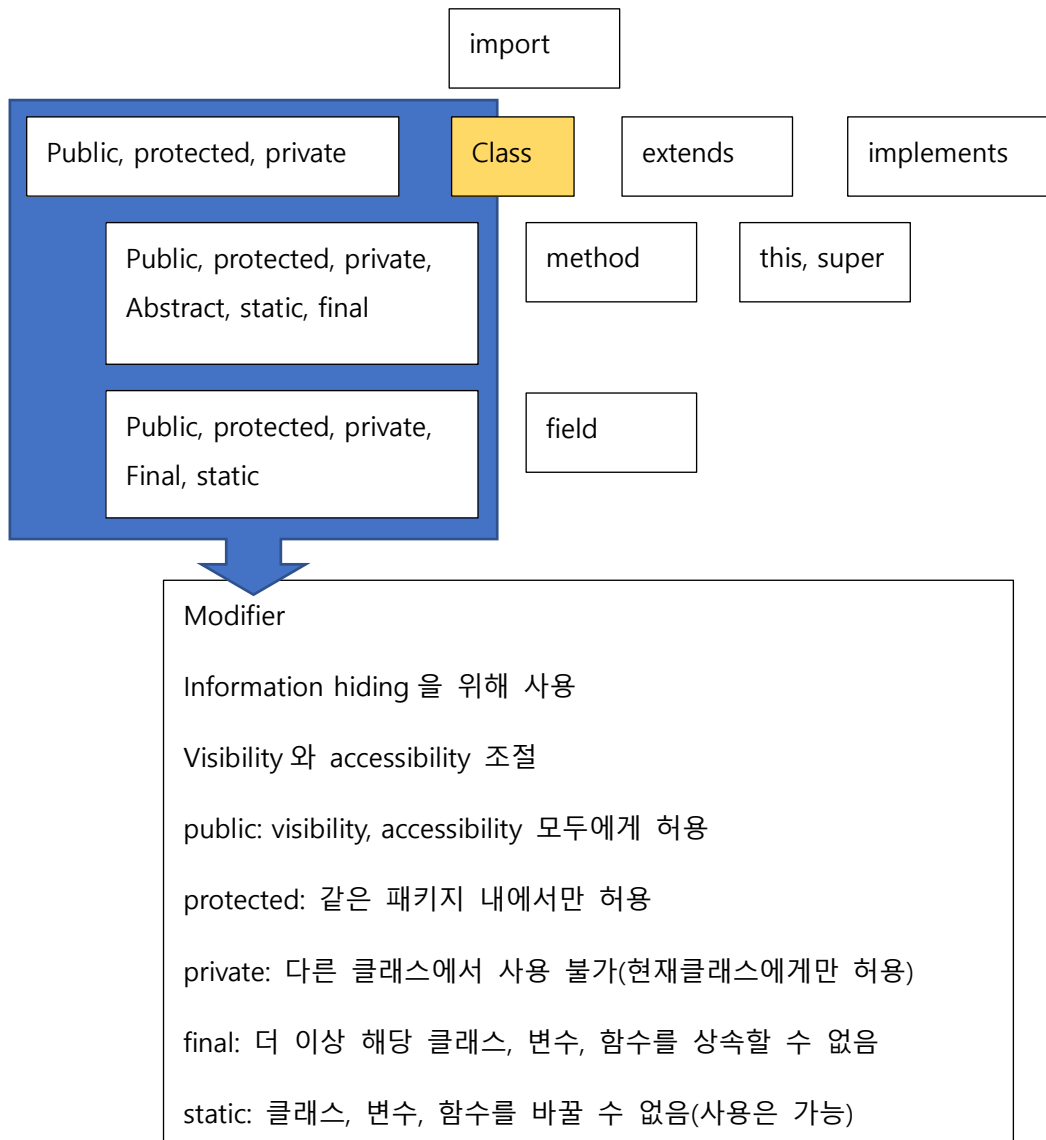
: Animal 클래스는 Mammal 클래스의 상위 클래스

Animal 클래스는 Reptile 클래스의 상위클래스

Mammal 과 Reptile 클래스는 Animal 클래스의 하위클래스

Dog 클래스는 Mammal 과 Animal 의 하위클래스

(출처 : <https://www.tutorialspoint.com/What-is-Is-a-relationship-in-Java>)



생성자(Constructor)

클래스 이름과 동일한 이름의 함수

인스턴스 생성 시 클래스의 변수를 초기화하는 기능

return X

생성자를 따로 선언하지 않으면 자동으로 기본 생성자가 생성되지만 이는 바람직하지 않다

★인스턴스 생성 시 입력값이 없으면(원하는 값으로 초기화하지 않으면) String 에는 null, int 에는 0 이 들어가지만 만약 0 을 넣고 싶다고 해도 초기화를 해야 함

>초기화를 하지 않을 경우 어떤 값이 들어갈지는 아무도 모름

오버로딩(Overloading)

=Polymorphism(현상은 같지만 성질이 다른 것)

이름이 같지만 signature 가 다른 경우

signature : 함수의 이름, 변수의 수, 변수의 타입 등

메소드, 생성자에게 해당

다른 메소드와 이름과 의미는 같지만 변수의 타입과 수가 다르다(실행 내용이 다른 것은 오버라이딩)

여러 개의 메소드를 구분하는 것 > ECA rule 의 C(Condition)가 하는 일(signature 체크, 매치, find)

ECA rule(Event, Condition, Action)

이벤트 기반 아키텍처 및 활성 데이터베이스 시스템에서 활성 규칙의 구조를 참조하기 위한 바로가기

이벤트가 감지되면 조건을 평가하고 조건이 충족되면 행동을 실행한다.

세부분으로 구성

1. 이벤트 부분: 함수 호출, 호출을 발동시키는 신호를 결정
2. 조건부분: 이벤트를 수행하기 위해 충족해야 하는 조건

조건은 지정된 이벤트 발생 시에만 확인된다

시그니처 체크, 매치, 탐색

3. 액션부분: 호출한 함수 실행

Event 가 발생하면 signature(함수 이름, 변수의 수, 변수의 타입 등)을 생성, 전달(message passing)

Condition 에서 어떤 함수(메소드)를 실행할지 결정

Action 에서 Condition 에서 받아온 메소드를 실행

Method Invocation > Making Signature(Message) > Message Passing > Condition Check > Method Execution

메소드를 호출(Method Invocation)하면(이벤트가 발생, ECA 의 A) ECA 의 C 로 signature 를 전달한다(message passing).
ECA 의 C 에서 받은 시그니처를 메소드들 중에서 비교하여 실행할 메소드를 찾는다.
그 후 ECA 의 A 에서 메소드를 실행한다.

오버라이딩(Overriding)

: 함수 재정의, 상속 시 사용할 수 있다

주용도 : 실행할 코드를 다시 구현하는 경우

Ex)인터페이스에 있는 메소드는 실행 코드만 있고 실행할 코드가 존재하지 않는다.

(인터페이스는 자주 쓸 함수 이름만 만들어 놓은 것)

이를 상속받아서 구현할 때 실행할 코드를 적어야 하는데 이 때 사용하는 것이 오버라이딩이다.

클래스마다 같은 함수를 실행할 코드가 달라야 하고, 비슷한 클래스를 여러 개 만들게 된다면 오버라이딩을 통해 중복을 줄이고 원하는 함수를 만들 수 있다.

```
Interface Name {  
  
    eat( ){ }  
  
}  
  
class MyStyle implements Name{  
  
    eat( ){ 먹는 방법 구현 코드 }  
  
}
```

오버라이딩을 금지하고 싶다면 static 을 사용하면 된다

Static 을 사용하면 상속은 가능하지만 오버라이딩은 불가능하게 된다

(하나의 메모리에 하나만 저장되기 때문 / 수정 불가)

final : 상속을 불가능하게 한다

IDL(Interface Definition Language)

소프트웨어 컴포넌트의 인터페이스를 묘사하기 위한 명세 언어이다

명세언어 : 특정한 법칙들에 따라 적절하게 구성된 문자열들의 집합

‘무엇을’을 기술하는 언어, 일반적으로 직접 실행되지 않는다

어느 한 언어에 국한되지 않는 언어중립적인 방법으로 인터페이스를 표현함으로써 같은 언어를 사용하지 않는 소프트웨어 컴포넌트 사이의 통신을 가능하게 한다

소프트웨어 컴포넌트 :

기존의 시스템이나 소프트웨어를 구성하는 컴포넌트를 조립해서 하나의 새로운 응용프로그램을 만드는 소프트웨어 개발론

(출처:

https://ko.wikipedia.org/wiki/%EC%BB%B4%ED%8F%AC%EB%84%8C%ED%8A%B8_%EA%B8%B0%EB%B0%98_%EC%86%8C%ED%94%84%ED%8A%B8%EC%9B%A8%EC%96%B4_%EA%B3%B5%ED%95%99)

IDL 은 다른 두 개의 시스템을 연결하는 다리 역할을 한다

(출처:

https://ko.wikipedia.org/wiki/%EC%9D%B8%ED%84%B0%ED%8E%98%EC%9D%B4%EC%8A%A4_%EC%A0%95%EC%9D%98_%EC%96%B8%EC%96%B4)

한 언어로 작성된 프로그램이나 객체가 알려지지 않은 언어로 작성된 다른 프로그램과 통신을 할 수 있도록 해주는 언어를 지칭하는 일반적인 용어

분산 객체 기술에서 새로운 객체들이 어떠한 플랫폼 환경에서도 보내질 수 있으며 그 환경에서 어떻게 실행되는지를 알아내는 것은 매우 중요하다

IDL 은 설명되어야 할 프로그램의 인터페이스들 등의 경미한 확장을 요구함으로써 동작한다

(출처: <https://soulduo.tistory.com/91>)

패키지

특정 같은 서비스를 위한 클래스의 집합

클래스의 첫번째 줄에 `package packageName;`이 들어감으로 패키지를 만들 수 있다

클래스의 기능에 따라 여러 패키지를 만들어서 관리하는 것이 중요하다

다른 패키지에 있는 클래스를 이용하기 위해서는 `import` 키워드를 사용해야 한다.

this

Heap 영역 안에 존재하는 자기 자신의 메모리 공간을 가리킨다.

super

메모리 공간에 있는 부모 클래스를 가리킨다

this()

자기 자신을 다시 만드는 것(값 초기화) > 새로운 인스턴스 생성

생성자인 경우 일부를 재초기화한다

그닥 좋지 않은 함수

super()

부모클래스의 속성을 만들 때 사용한다

부모클래스가 필요할 때 사용한다

유용한 함수

instanceof

어떤 타입의 객체인지 확인할 때 사용한다

프로그램을 짜다보면 길어지게 되는데, 어떤 객체인지 알 수 없을 때(찾고 싶을 때)사용한다