회의록

1. **회의 개요**

|  |  |  |  |
| --- | --- | --- | --- |
| 회 의 명 | 스터디 계획서 작성 | | |
| 일 시 | 2018.01.02 (15:00~16:00) | 장 소 | 도서관 5층 세미나실 |
| 참 석 자 | 심수현, 이연재, 정경환 | 작성자 | 정경환 |

**2. 회의 내용**

|  |  |
| --- | --- |
| **Title** | **Contents** |
| **회의내용**  **및**  **결과** | * 추상클래스와 인터페이스 * 추상클래스 (abstract, extends) * 하나 이상의 추상 메소드를 가지고 있는것(선언부만 있고 구현부가 없음) * 일반 메소드, 변수 포함 가능 , 생성자를 가질 수 있다.  “NormalClass ab = new NormalClass()” * ‘강제성’ / abstract 메소드 오버라이딩 필수 * 추상클래스를 상속받아 기능을 이용하고 확장 시킴 * 다중 상속 불가 (상속의 개념을 갖는다.) * 공통된 개념을 표현할 때   예를 들면 여러 클래스가 4개의 공통 기능과 1개의 다른 기능을 포함해야 한다. 1개의 다른 기능 때문에 모든 클래스에 똑같은 코드를 작성하는 것은 비효율적이다. 이 경우 5개의 기능을 추상클래스를 통해 작성하는 것이다.  public abstract class 추상클래스명 {  public void 공통기능1() { System.out.println("첫 번째 공통기능입니다."); }  public void 공통기능2() { System.out.println("두 번째 공통기능입니다."); }  public void 공통기능3() { System.out.println("세 번째 공통기능입니다."); }  public void 공통기능4() { System.out.println("네 번째 공통기능입니다."); }  public abstract void 다른기능();  }  4개의 공통 기능은 일반 메소드로 작성하고 1개의 다른 기능은 추상메소드로 구현한다. 그러면 이 추상클래스를 extends 하여 추상메소드만 다르게 구현함으로써 효율적인 코드를 작성할 수 있게 된다.   * 인터페이스 (interface , implements) * 메소드들의 형태만 선언한 것 * 인터페이스 구현시 추상 메소드를 알맞게 ‘구현’ 하여 사용 * 생성자를 가질 수 없다. / 구현한 클래스를 통해 인스턴스화 가능  “CommonAnimal ca = new Horse()” * 설계도의 개념으로 같은 인터페이스를 구현한 클래스들이 같은 기능을 동작하도록 보장한다. * 모든 메소드의 구현 필수 * 여러개의 인터페이스를 상속받을 수 있다. * 높은 응집도와 낮은 결합도  변경, 확장이 쉽다. |
| **기타사항** | * 2018.01.05(16:00~18:00)스터디 * 네트워크, JDBC * 16 강 실전문제 3번 * 17강 실전문제 1~6번 |