

First Java

유영진 · 이도연



XD 연두어디션

JAVA 객체지향 프로그래밍

First Java

first
coding

목차

1. 코딩해보기

2. 프로젝트

First JAVA

1. 코딩해보기

- | | |
|---------------------------|----------------|
| 1. 객체지향 언어인 JAVA | 9. API |
| 2. JAVA에서 데이터 다루기 | 10. 예외처리 |
| 3. JAVA에서 제공하는 연산자 | 11. 컬렉션 프레임워크 |
| 4. 프로그램 흐름제어를 위한 조건문과 반복문 | 12. JAVA I/O |
| 5. 객체지향 - 인스턴스와 클래스 | 13. 스레드 |
| 6. 배열 | 14. GUI |
| 7. 객체지향 - 상속 | 15. 네트워크 프로그래밍 |
| 8. 객체지향 - 추상클래스와 인터페이스 | |

1. MyProject라는 프로젝트를 생성해봅시다.
2. Member 클래스를 생성하고, main()메소드를 만들고 자기 자신의 이름을 출력하는 메소드를 만들어 출력해 봅시다.
3. Calculator 클래스를 정의해 봅시다.
 - ① 숫자 두 개를 매개변수의 인자로 전달받아 더하기 메소드를 추가합니다.
 - ② 숫자 두 개를 매개변수의 인자로 전달받아 빼기 메소드를 추가합니다.
 - ③ 숫자 두 개를 매개변수의 인자로 전달받아 곱하기 메소드를 추가합니다.
 - ④ 숫자 두 개를 매개변수의 인자로 전달받아 나누기 메소드를 추가합니다.

1. 앞서 생성 했던 Member 클래스에 main() 메소드 안에 아래 조건의 변수를 정의해봅시다.

- ① String 타입의 이름을 저장할 수 있는 변수 name을 정의해봅시다.
- ② int 타입의 나이를 저장할 수 있는 변수 age를 정의해봅시다.
- ③ double 타입의 키를 저장할 수 있는 변수 height를 정의해봅시다.
- ④ boolean 타입의 JAVA책의 보유 여부를 저장할 수 있는 변수 hasBook를 정의해봅시다.
- ⑤ 이름과 나이, 키, 책의 보유 여부를 출력해봅시다.

2. Person 클래스를 만들어 보고, 아래의 회원 정보를 저장하는 변수들을 선언해봅시다.

변수이름을 작성하는 규칙에 맞게 직접 변수 이름을 정의해 보세요.

- ① 회원이름을 저장하는 변수
- ② 회원 전화번호 (000-0000-0000)를 저장하는 변수
- ③ 회원 주민등록번호 (000000-0000000 또는 00000000000000)를 저장하는 변수

1. Calculator 클래스를 정의해봅시다.

- ① 정수 두 개를 매개변수의 인자로 전달받아 더하기연산 후 출력하는 메소드를 정의
- ② 정수 두 개를 매개변수의 인자로 전달받아 빼기연산 후 출력하는 메소드를 정의
- ③ 정수 두 개를 매개변수의 인자로 전달받아 곱하기연산 후 출력하는 메소드를 정의
- ④ 정수 두 개를 매개변수의 인자로 전달받아 나누기연산 후 출력하는 메소드를 정의
- ⑤ 실수 반지름 하나를 매개변수의 인자로 전달받아 원의 둘레를 구해 반환하는 메소드를 반환하는 메소드를 정의
- ⑥ 실수 반지름 하나를 매개변수의 인자로 전달받아 원의 넓이를 구해 반환하는 메소드를 반환하는 메소드를 정의
원의 둘레 : $2 \times \pi \times r$, 원의 넓이 : $\pi \times r \times r$
- ⑦ main() 메소드를 정의하고 각각의 메소드를 호출해서 결과를 콘솔에 출력해봅시다.
- ⑧ 콘솔에서 사용자에게 데이터를 받아 메소드를 호출할 때 사용자에게 받은 데이터를 메소드의 매개변수의 인자로 전달하는 코드를 main() 메소드에 추가해봅시다.

1. 앞에서 만든 Member 클래스에는 아래 요구사항에 맞는 메소드를 정의하고, main()메소드에 해당 메소드를 호출해서 결과를 확인해보세요. 메소드의 이름도 JAVA의 규칙에 맞게 정의해 봅시다.

- ① 독감예방 접종이 가능한지 여부를 확인하는 메소드를 정의합니다.
 - 매개변수로 태어난 해(년도)를 전달받습니다.
 - 15세 미만의 경우와 65세 이상의 경우 "무료예방접종이 가능합니다." 메시지가 콘솔에 출력하도록 합니다.
 - 위에서 정의한 조건의 범위가 아닌 나머지의 경우 "무료접종 대상이 아닙니다."라고 출력하도록 합니다.
- ② 건강검진 대상 여부를 판별하고 어떤 검진이 가능한지 확인하는 메소드를 정의합니다.
 - 매개변수로 태어난 해(년도)를 전달받습니다.
 - 대한민국 성인(20세)의 경우 무료로 2년마다 건강검진을 받을 수 있습니다.
 - 짝수 해에 태어난 사람은 올해가 짝수년이라면 검사 대상이 됩니다.
 - 40 이상의 경우는 암 검사도 무료로 검사를 할 수 있습니다.

1. Member 클래스에는 아래 요구사항에 맞는 변수와 메소드를 정의하고, main()메소드 에 해당 메소드를 호출해서 결과를 확인해보세요.
단 인스턴스의 생성은 생성자를 이용해서 초기화하는 코드를 작성해 봅시다.
 - ① 아래의 데이터를 저장 이름, 전화번호, 전공, 학년, email, 생일, 주소
 - ② 위에서 정의한 정보를 출력하는 메소드 정의
 - ③ 모든 정보를 저장하도록 하는 생성자와 생일과 주소는 저장하지 않을 수 있는 생성자를 정의
 - ④ main() 메소드에서 두 가지 생성자를 이용해서 인스턴스 생성하고 출력 메소드를 통해 저장된 데이터 출력

1. 국어, 영어, 수학 점수 10개씩을 저장하는 배열을 정의하고 점수를 모두 출력하고, 평균 점수를 출력하는 프로그램을 작성해봅시다.
2. Student 클래스를 정의해봅시다.
 - ① 학생이름, 국어점수, 영어점수, 수학점수를 저장하는 변수를 정의 합니다.
 - ② 변수는 캡슐화를 합니다. getter/setter 메소드를 정의합니다.
 - ③ 총점과 평균을 구해 결과를 반환하는 메소드를 정의합니다.
3. main()메소드에 아래 내용을 정의해봅시다.
 - ① Student 타입의 배열을 선언하고, 요소 10개를 저장할 수 있는 배열 인스턴스를 생성해 봅시다.
 - ② Student 타입의 인스턴스를 생성하고 배열에 저장하는 코드를 정의해봅시다.
 - ③ 배열에 저장된 Student 타입의 인스턴스의 메소드를 이용해서 모든 데이터를 출력해봅시다.

1. Person 이라는 클래스를 정의해봅시다.

- ① 이름을 저장하는 변수, 주민등록번호를 저장하는 변수를 정의해봅시다.
- ② 인사하는 메소드를 정의해봅시다.
 - “안녕하세요. 저는 000입니다. 00살 입니다.”라는 문자열이 출력하도록 정의합시다.

2. Person 클래스를 상속해서 확장하는 새로운 클래스 Male 클래스와 Female 클래스를 정의 해봅시다.

- ① 각 클래스는 상속 받은 멤버 외에 추가적인 변수와 메소드를 추가해서 새로운 클래스를 정의해봅시다.
- ② 각 클래스는 상속 받은 멤버 외에 추가적인 변수와 메소드를 추가해서 새로운 클래스를 정의해봅시다.
- ③ Person 클래스에서 정의된 인사하는 메소드를 오버라이딩 해봅시다.
- ④ Person 클래스에 생성자를 정의해서 인스턴스 변수들을 초기화 해봅시다.

3. main()메소드를 정의해봅시다.

- ① Person 클래스를 상속받은 Male클래스와 Female클래스를 이용해서 인스턴스를 생성해 봅시다.
- ② 생성된 인스턴스들을 이용해서 메소드를 호출해봅시다.

아래 코드는 계산기 클래스를 정의할 때 가이드 역할을 하도록 정의해놓은 인터페이스입니다.

```
interface Calculator {  
    long add(long n1, long n2);  
    long subtract(long n1, long n2 );  
    long multiply(long n1, long n2 );  
    double divide(double n1, double n2 );  
}
```

1. Calculator 인터페이스를 상속하는 추상 클래스를 정의해봅시다.
2. Calculator 인터페이스를 상속하는 인스턴스를 생성할 수 있는 클래스를 정의해봅시다.
3. 다형성의 특징으로 상위 타입인 인터페이스 타입의 참조변수에 인터페이스를 구현한 클래스 타입의 인스턴스를 참조하는 코드를 작성해 봅시다.

```
class Person {  
    String name;  
    String personNumber;  
}
```

1. 위 Person 클래스의 equals() 메소드를 오버라이딩해서 주민등록번호가 같으면 인스턴스로 판별하는 프로그램을 만들어봅시다.
2. 1~100,000,000까지 더하기하는 연산의 실행 시간을 측정하는 프로그램을 만들어봅시다.
3. 사용자에게 이름을 입력 받아 입력 받은 문자열을 정상적인 문자열의 이름으로 표현하는지 판별하고, 공백으로 입력되었는지도 판별하는 프로그램을 만들어봅시다.
4. 자신의 생일을 기준으로 오늘까지 몇 일을 살았는지 출력하는 프로그램을 만들어봅시다.

1. 콘솔에서 사용자 아이디를 입력 받아 정상적인 영문자와 숫자로만 이루어진 값을 입력했는지 확인하는 프로그램을 만들어봅시다.
 - ① 사용자 예외 클래스를 정의해서 예외를 발생 시켜 봅시다.
 - ② 예외 클래스 이름은 `BadIdInputException`이라고 정의합시다.
2. `Scanner` 클래스로 태어난 년도를 입력 받을 때 `int` 타입으로 받도록 합시다.
이때 `nextInt()` 메소드를 사용하게 되는데 이때 발생하는 예외처리를 하도록 합시다.

축구선수 클래스를 만들어 봅시다.

```
class FootballPlayer {  
    String name;  
    int number;  
    String team;  
    int age  
}
```

1. 축구선수 인스턴스를 저장할 수 있는 `List<E>` 컬렉션 인스턴스를 생성해서 인스턴스를 저장하고 출력하는 프로그램을 만들어 봅시다.
2. 축구선수의 인스턴스가 팀과 이름 그리고 나이가 같으면 같은 선수라 판단하고 입력이 되지 않도록 `Set<E>` 컬렉션을 이용해서 축구선수 인스턴스를 저장하고 출력하는 프로그램을 만들어 봅시다.
3. `TreeSet<E>`을 이용해서 팀 이름순으로 정렬하고, 같은 팀의 선수들은 이름 순으로 정렬하고, 같은 이름의 선수는 번호 순으로 저장하는 프로그램을 만들어 봅시다.
4. 축구선수의 번호를 key로 하고 축구선수 인스턴스를 저장하는 `Map<K,V>` 인스턴스를 이용해서 프로그램을 만들어봅시다.

1. 콘솔기반으로 메모장 기능을 만들어 봅시다.

- ① File 클래스를 이용해서 저장 폴더 생성
- ② 문자기반 스트림을 이용해서 날짜와 제목, 메모를 파일에 저장
- ③ 파일의 이름은 날짜와 메모의 제목을 이용해서 생성
- ④ 메모리스트와 파일 읽기 기능을 구현해봅시다.

2. 앞 Chapter에서 만들어본 축구선수 정보 파일로 저장하는 프로그램을 만들어 봅시다.

- ① 축구선수 정보 인스턴스를 List<E>에 저장하는 프로그램을 만들어 봅시다.
- ② 이 인스턴스 들을 파일로 저장하는 기능을 만들어 봅시다.
- ③ 저장된 파일을 객체로 만드는 기능을 만들어봅시다.

1. 10초 안에 맞추는 하이로우 게임을 만들어 봅시다.

- ① 1~100 사이의 랜덤 한 숫자를 추출합니다.
- ② 사용자에게 숫자를 입력 받고, 랜덤 숫자와 비교하고, 높은 숫자인지 낮은 숫자인지 출력
- ③ 10초 카운팅은 스레드를 이용해서 처리해봅시다.
- ④ 10초 이전에 맞추면 미션 성공, 10초가 지나면 프로그램 종료하는 흐름으로 만들어봅시다.

2. 복사하고자 하는 파일 경로와 복사할 파일의 디렉토리 주소를 받아 파일을 복사하는 프로그램을 만들어봅시다.

- ① 복사할 대상 파일의 경로와 복사해올 위치 경로를 받는 메인 스레드는 멈춤 없이 실행하고 데이터의 복사는 스레드로 처리하는 프로그램을 만들어 봅시다.
- ② 파일이 복사가 완료되면 완료 메시지를 콘솔에 출력합니다.

1. 간단한 계산기를 만들어봅시다.

- ① 사칙연산과 나머지를 구하는 기능을 구현
- ② 각 기능은 버튼으로 기능을 실행 하도록 구현해 봅시다.

2. 로또 번호 추출 하는 프로그램을 만들어 봅시다.

- ① 랜덤하게 번호를 추출하고, 배열이나 컬렉션에 저장하는 프로그램을 만들어봅시다.
- ② 저장된 번호는 윈도우 화면에 출력시키는 프로그램을 만들어 봅시다.

First JAVA

1. 프로젝트

Project 1. 객체지향 - 인스턴스와 클래스

Project 2. 배열

Project 3. 객체지향 - 상속

Project 4. 객체지향 - 추상클래스와 인터페이스

Project 5. API

Project 6. 예외처리

Project 7. 컬렉션 프레임워크

Project 8. JAVA I/O

Project 9. 스레드

Project 10. GUI

아래 이미지는 우리가 흔히 볼 수 있는 스마트폰에 이름, 전화번호, 이메일 등과 같은 연락처 정보를 저장하는 애플리케이션 화면입니다.

아래 데이터들을 저장하고, 데이터를 출력하는 메소드를 가지는 클래스 Contact를 정의해봅시다.

저장 정보

- 이름
- 전화번호
- 이메일
- 주소
- 생일
- 그룹

기능

- 위 데이터를 출력하는 기능

추가 요구 사항

- 변수들은 직접 참조를 막아 캡슐화 처리를 하도록 해봅시다.
변수의 직접 참조는 안되지만 변수의 값을 얻을 수 있는 메소드(getter)와 변수에 값을 저장할 수 있는 메소드(setter)를 정의합니다.
- 인스턴스 생성과 함께 데이터를 초기화 할 수 있도록 생성자를 정의해봅시다.
- 저장할 데이터를 콘솔에서 사용자의 입력 값으로 인스턴스를 생성해봅시다.

실행 과정

- main()메소드를 정의합니다.
- 연락처 데이터를 저장하는 인스턴스를 생성합니다.
- 변수 값을 반환하는 각각의 메소드를 호출해서 각 데이터를 출력문으로 출력합니다.
- 생성된 인스턴스의 정보 출력 메소드를 호출합니다.
- 인스턴스의 각 변수에 값을 바꾸는 메소드를 이용해서 데이터를 수정합니다.
- 인스턴스의 출력메소드를 다시 실행합니다.

프로젝트-1 에서 정의한 Contact 클래스를 기반으로 아래 요구사항을 추가해서 프로그램을 작성 합니다.

1. SmartPhone 클래스를 정의합니다. 이 클래스는 연락처 정보를 관리하는 클래스입니다.

- ① Contact 클래스의 인스턴스 10개를 저장 할 수 있는 배열을 정의합니다.
- ② 배열에 인스턴스를 저장하고, 수정하고, 삭제, 저장된 데이터의 리스트를 출력하는 메소드를 정의합니다.

2. main()메소드를 아래의 요구조건을 정의해봅니다.

- ① SmartPhone 클래스의 인스턴스를 생성합니다.
- ② 사용자로부터 입력을 받아 Contact 인스턴스를 생성해서 SmartPhone 클래스의 인스턴스가 가지고 있는 배열에 추가합니다.
- ③ 10번 반복해서 배열에 추가합니다.
- ④ 배열의 모든 요소를 출력합니다.
- ⑤ 배열의 모든 요소를 검색합니다.
- ⑥ 배열의 요소를 삭제해 봅니다.
- ⑦ 배열의 요소를 수정해 봅니다.

프로젝트-2 에서 정의한 Contact 클래스를 상속의 구조로 만들어 봅니다.

1. Contact 클래스는 기본정보를 저장하고 기본 정보를 출력하는 메소드를 정의합니다.
 - 생성자를 통해 기본 정보들을 초기화 합니다.
2. 그룹에 해당하는 정보들을 추가적으로 정의하는 새로운 클래스들을 정의합니다. 회사, 거래처의 정보를 저장하는 하위 클래스를 정의합니다.
 - ① CompanyContact 회사, 거래처의 정보를 저장하는 하위 클래스를 정의합니다.
 - 회사이름, 부서이름, 직급 변수 추가
 - 정보를 출력하는 메소드를 오버라이딩 해서 추가된 정보를 추가해서 출력
 - ② CustomerContact 회사, 거래처의 정보를 저장하는 하위 클래스를 정의합니다.
 - 거래처회사이름, 거래품목, 직급 변수 추가
 - 정보를 출력하는 메소드를 오버라이딩 해서 추가된 정보를 추가해서 출력
3. SmartPhone 클래스의 배열을 다형성의 특징을 이용해서 상위 타입의 배열을 생성해서 하위 클래스의 인스턴스를 저장하는 형태로 프로그램은 작성해봅시다.

프로젝트-3 에서 구현한 상속 구조를 기초로 Contact 클래스를 추상클래스로 만들어봅시다.

1. ShowData인터페이스에 추상 메소드 void showData() 정의되어 있는 구조
2. Contact클래스가 ShowData인터페이스를 상속하면서 추상 메소드를 보유하는 관계로 생성
3. Contact클래스는 추상 메소드를 가지고 있어 추상클래스가 되는 형태로 정의
4. SmartPhone클래스에 있는 배열의 타입이 추상클래스로 되어도 문제가 없는 것을 확인

이번 Project에서는 메뉴를 두고 연락처의 입출력 및 관리가 프로그램이 종료되지 않고 유지가 되도록 하는 프로그램을 만들어봅시다.

1. 아래의 흐름이 유지가 되도록 메뉴를 무한반복 처리해 봅시다.

```
=====
연락처 관리프로그램
1. 연락처 입력
2. 연락처 검색
3. 연락처 수정
4. 연락처 삭제
5. 연락처 전체 리스트 보기
6. 종료
=====
```

2. 입력 또는 수정할 때 공백 문자열을 입력 받으면 다시 입력 받도록 흐름을 만들어봅시다.
3. 입력할 때 전화번호가 같은 데이터가 입력되면 입력이 되지 않도록 흐름을 만들어봅시다.

프로젝트-5 에서 구현한 구조를 기초로 예외처리를 해봅시다.

1. 메뉴 입력 시 발생할 수 있는 예외에 대하여 예외 처리합니다.
2. 연락처 이름 입력 시에 공백에 대한 예외처리와 영문자와 한글만 허용하는 예외 처리를 해봅시다.
3. 전화번호 형식에 맞지 않을 때 예외처리를 하고 중복될 때 예외 상황이 발생하도록 하고 예외 처리를 합니다.

지금까지 진행한 Project는 배열을 이용해서 연락처를 저장했습니다.

이번 Project에서는 컬렉션 프레임워크를 이용해서 연락처 인스턴스를 저장하도록 프로그램을 만들어 봅시다.

1. List<E>를 이용해서 저장 및 관리하는 프로그램을 만들어 봅시다.
2. HashSet<E>을 이용해서 저장 및 관리하는 프로그램을 만들어 봅시다.
3. HashMap<K,V>를 이용해서 저장 및 관리하는 프로그램을 만들어 봅시다.

프로젝트-7 에서 List<E>에 저장된 인스턴스들을 직렬화 하고 역 직렬화 하는 프로그램을 만들어 봅시다.

1. 파일 저장 메뉴와 파일 읽어오기 기능을 추가합니다.

```
=====
```

연락처 관리프로그램

1. 연락처 입력
2. 연락처 검색
3. 연락처 수정
4. 연락처 삭제
5. 연락처 전체 리스트 보기
6. 파일 저장
7. 파일 로드
8. 종료

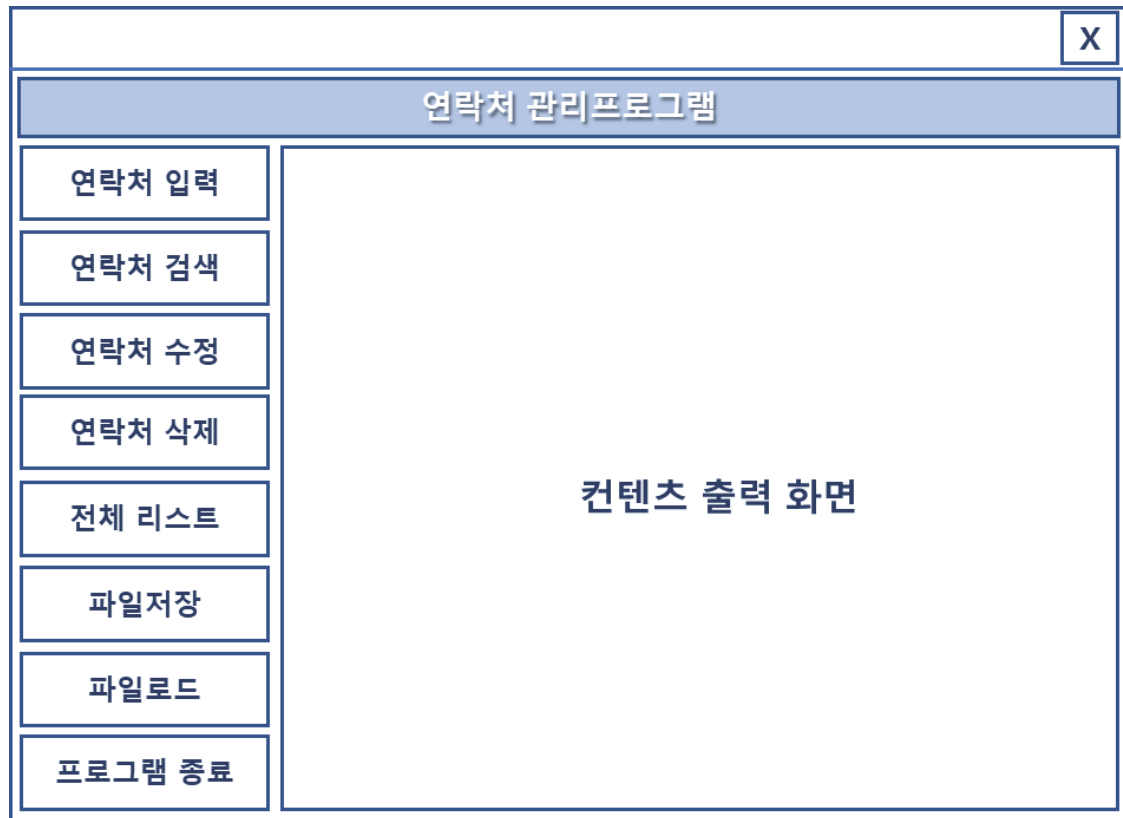
```
=====
```

2. 프로그램이 시작할 때 파일 읽어오기.

파일 저장과 파일 로드 기능 구현 부분을 스레드로 구현해 봅시다.

1. 파일 저장 메뉴와 파일 읽어오기 기능 구현 부분을 스레드로 처리해 봅시다.
2. 스레드 처리 부분은 동기화도 처리합니다.

지금까지 구현한 기능들을 윈도우 안의 UI 컴퍼넌트들을 이용해서 구현해봅시다.



연락처 관리 프로그램

연락처 입력

연락처 검색

연락처 수정

연락처 삭제

전체 리스트

파일저장

파일로드

프로그램 종료

컨텐츠 출력 화면