

# First Java

유영진 · 이도연



XD 연두어디션

JAVA 객체지향 프로그래밍

# First Java

first  
coding

# 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

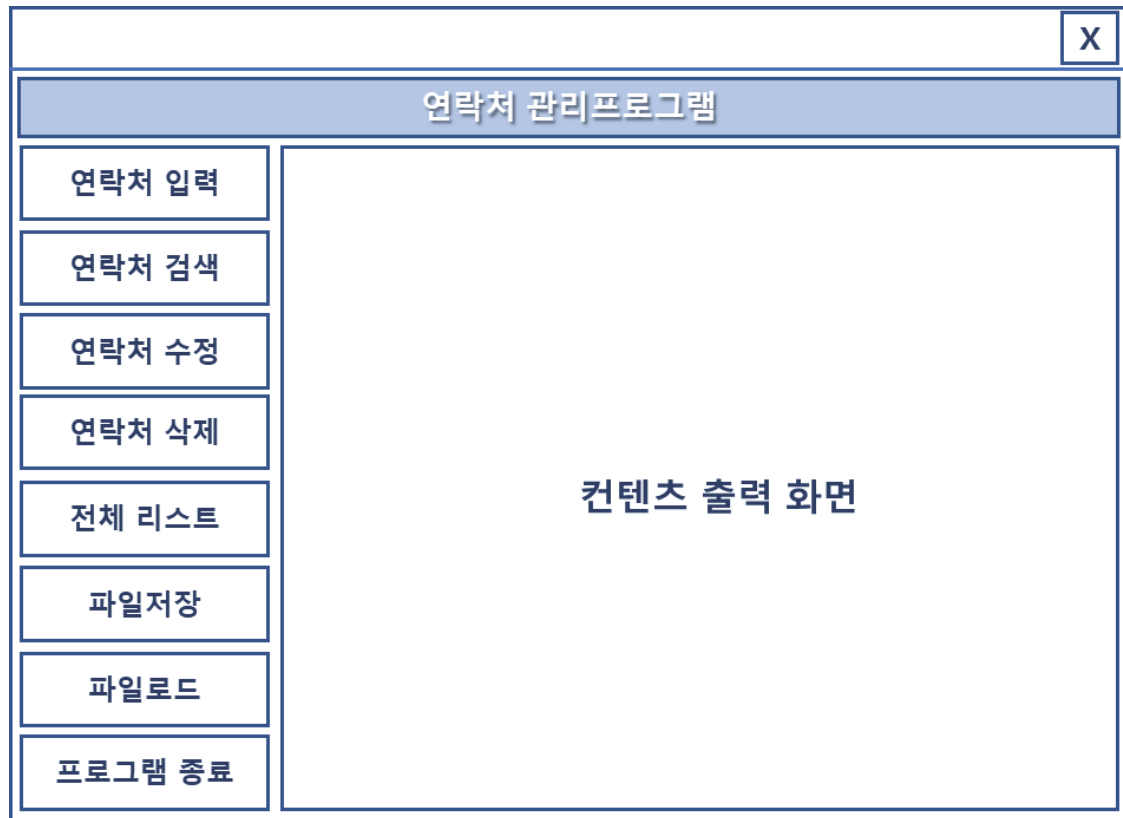
```
=====
```

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