객체지향프로그래밍

Contact App 구현 및 배포

4. 화면 장식과 데이터 저장

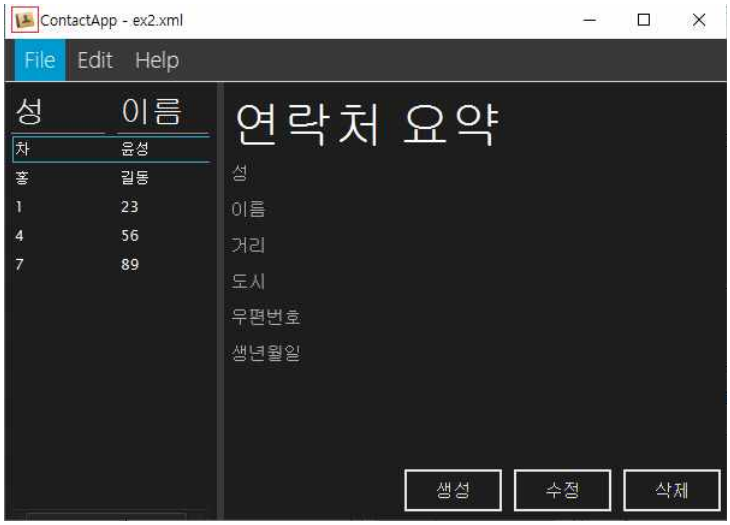
소프트웨어학부

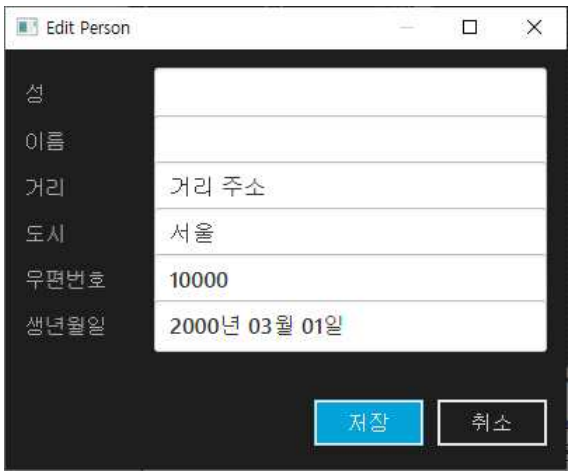
20163162

차윤성

1. 화면 장식

1) 아이콘






```
this.primaryStage.getIcons().add(new Image("file:resources/images/contact_book_512.png"));
```

- Project의 root directory의 resources/images 폴더 안의 address_book_32.png를 primaryStage의 아이콘으로 설정함
- MainApp.java의 start method에 구현 되어있음.

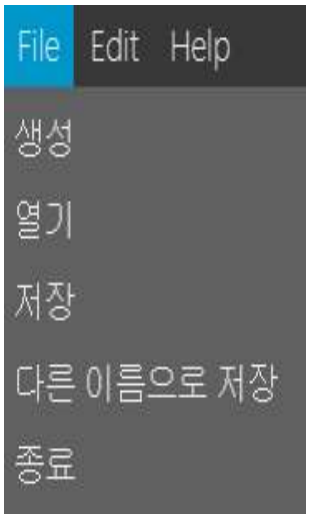
2) RootLayout / Overview / EditDialog

- 각 Stage의 구성을 새로 추가한 contact/view/DarkTheme.css 의 css 파일을 이용해 구성.
- css 파일은 표시되는 폰트의 크기, 폰트 종류, 폰트의 색 등을 정의하고 클래스로 묶음
- Stage 및 button의 배경색도 검은색으로 지정

2. Menu Bar 구성

1) File Menu

- File Menu는 생성/열기/저장/다른 이름으로 저장/종료로 구성됨



1-1) 생성

- “생성”을 선택하면 handleNew()으로 연결되어 Table을 비워줌

1-2) 열기

- “열기”를 선택하면 handleOpen()으로 연결되어 xml 파일을 열도록 함

1-3) 저장

- “저장”을 선택하면 현재 handleSave()으로 연결되어 현재 filePath에 data 저장

1-4) 다른 이름으로 저장

- “다른 이름으로 저장”을 선택하면 handleSaveAs()로 연결되어 filePath를 선택 후 해당 filePath에 data 저장

1-5) 종료

- “종료”를 선택하면 handleExit()으로 연결되어 프로그램을 종료하도록 함

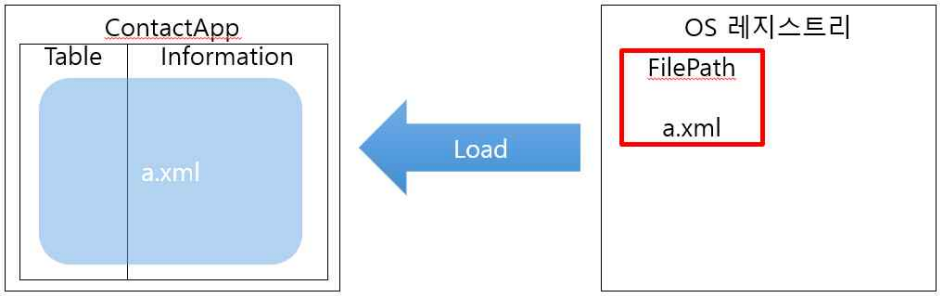
2) Help Menu-프로그램 소개

- "프로그램 소개“를 선택하면 handleAbout()으로 연결되어 프로그램 정보를 소개하는 Alert Message를 표시

3. Table Data Load/Store

private List<Persons> persons; (PersonListWrapper.java in contact.model package)

- Person 객체의 List를 가지는 class 작성
- Observable 형태의 data는 HDD에 저장 불가
  -> ObservableList 형태의 data를 일반 list 형태로 변형 시킨 후 HDD에 저장하도록 함
- Contact App이 처음 시작 될 때 마지막으로 사용한 xml file을 open함
  - 마지막으로 사용한 xml file 정보는 OS 특정 레지스트리에 저장됨



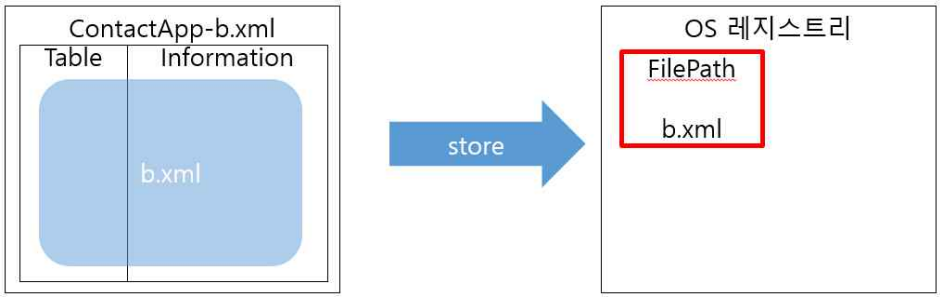
public File getPersonFilePath();

: 레지스트리에 저장되어 있는 xml file 정보를 읽어 반환해줌

: Preference - 프로그램의 상태 정보를 저장할 수 있는 공간(4k bytes)

: OS 레지스트리에 저장되어있는 filePath 정보를 받아 String 객체 filePath에 저장

- 한 번도 프로그램을 실행하지 않아 레지스트리 filePath 공간에 아무런 정보도 없다면 MainApp.java에서 샘플 데이터를 추가한 후 읽게 됨



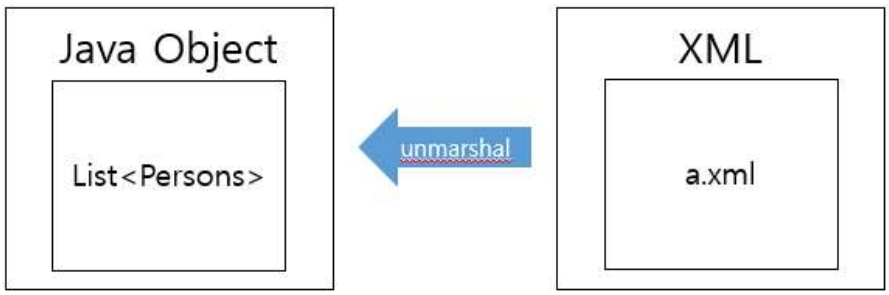
public void setPersonFilePath(File file);

: 현재 ContactApp에서 불러온 xml file 경로 정보를 인자로 받아 OS 레지스트리 filePath 공간에 저장

: 현재 load 되어있는 xml file의 Name을 primaryStage Title에 추가함(ContactApp-file.xml)

: 현재 load 되어있는 xml file이 없을 경우, primaryStage의 Title을 ContactApp으로 유지

1) XML File Load



- Load한 XML File에서의 Person Data들은 MainApp.java의 observable List에 저장됨(master data)

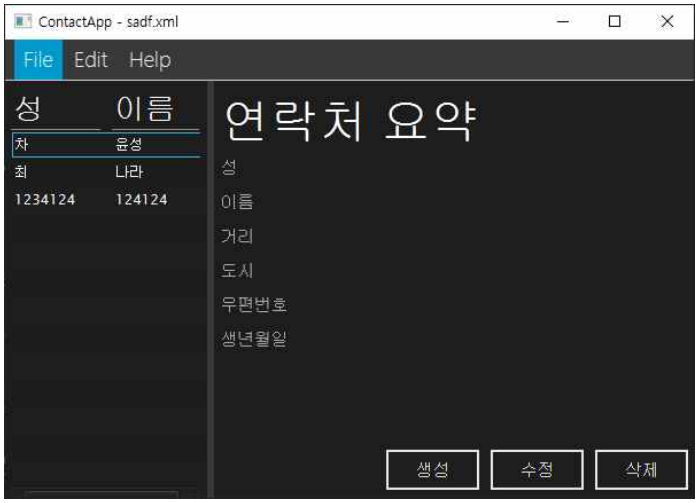
private ObservableList<Person> personData = FXcollections.*observableArray*();

Unmarshaller um = context.createUnmarshaller();

PersonListWrapper wrapper = (PersonListWrapper) um.unmarshal(file);

- xml file을 open 시 xml 형태의 data를 java Object로 변환해주는 JAXBContext를 이용해 언마샬링을 수행 후 disk에 저장되어 있는 xml File의 data를 Table에 표시함

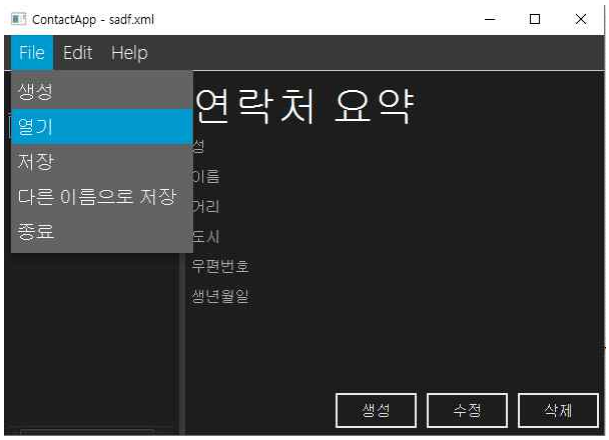
1-1) Program Run

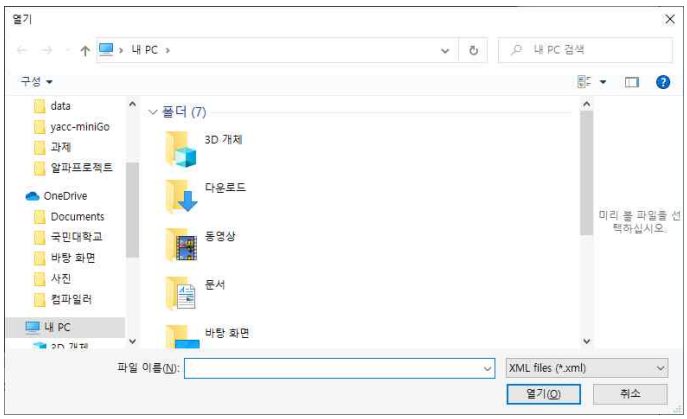


public void initRootLayout();

- 기존 프로그램에서 RootLayout을 초기화 할 때와 달리, getPersonFilePath를 이용해 OS 레지스트리에서 마지막으로 열었던 xml 파일을 읽어옴.
- OS 레지스트리에서 xml 파일을 읽어오지 못한 경우엔 filePath가 null로 전달되어 Table에 아무런 정보가 로드되지 않음

1-2) Open



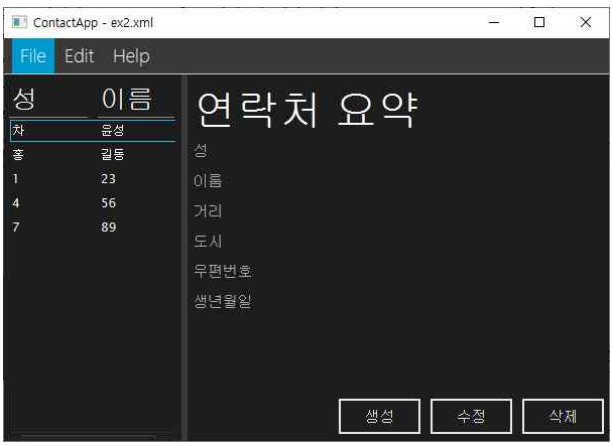


private void handleOpen();

- Menu bar의 "열기"를 선택 시 SceneBuilder에서 On Action으로 연동한 handleOpen 메소드 실행

File file = fileChooser.showOpenDialog(mainApp.getPrimaryStage());

- java의 fileChooser 객체를 통해 Window의 파일 탐색기를 실행
  open하려는 xml 파일을 선택하면, xml 파일을 java Object로 언마샬링하여 Table에 표시
  (선택한 xml 파일의 filePath를 가져와 loadPersonDataFromFile 호출)



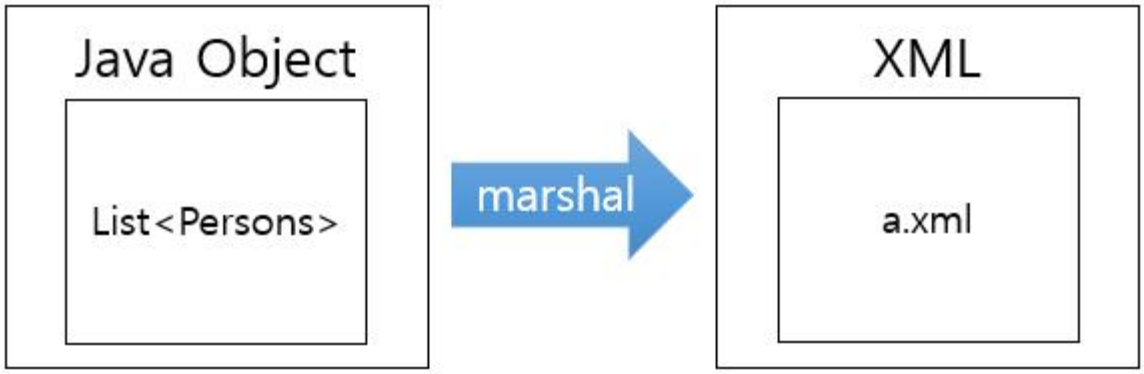
2) XML File Store

`public void savePersonDataToFile(File file);`

- 현재 ContactApp에서 수용하고 있는 Table data를 HDD에 저장하는 메소드

```
PersonListWrapper wrapper = new PersonListWrapper();
wrapper.setPersons(personData);
```

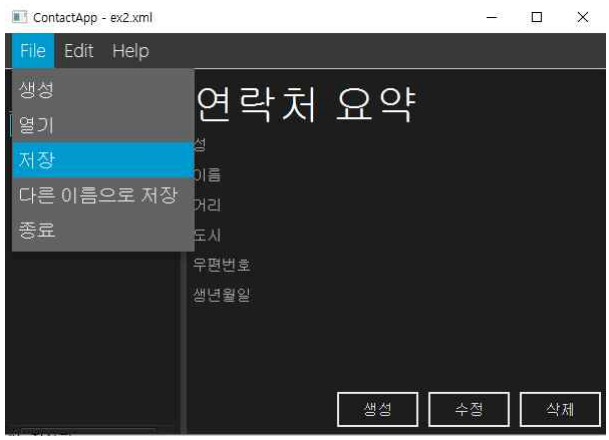
- 현재 ContactApp이 가지고 있는 Person 객체의 ObservableList 형태를 PersonListWrapper 클래스의 Person 객체들을 가지고 있는 List 형태로 저장하여 HDD에 저장될 수 있는 형태로 변환시켜줌




```
JAXBContext context = JAXBContext.newInstance(PersonListWrapper.class);
Marshaller m = context.createMarshaller();
m.setProperty(Marshaller.JAXB_FORMATTED_OUTPUT, true);
m.marshal(wrapper, file);
```

- JAXB(Java Architecture for XML Binding): Java Object를 XML로 직렬화하고, XML을 Java Object로 역직렬화 해주는 자바 API
- Marshaller의 porperty를 JAXB_FORMATTED_OUTPUT로 설정해 Object를 XML 형식의 data로 설정
- 현재 wrapper가 가지고 있는 Object인 Person 객체의 List를 인자로 받은 filePath에 XML형식으로 마샬링하여 저장함
- 이 후 저장한 xml file 경로를 setPersonFilePath를 이용해 OS 레지스트리에 저장함
- file path를 잡지 못하거나 마샬링이 불가능하는 등의 예외가 발생할 경우 Alert message를 출력하여 Data를 저장할 수 없다고 표시

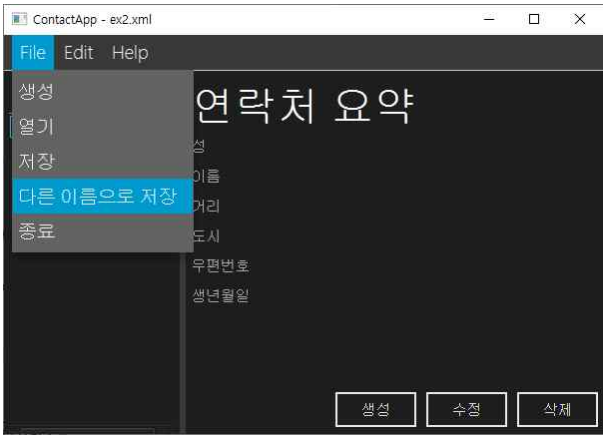
2-1) Save

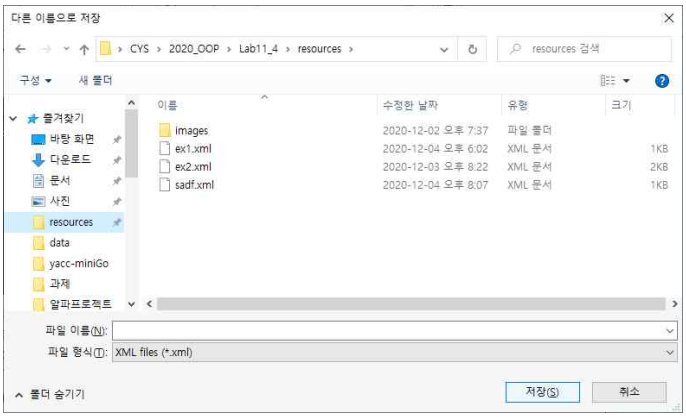


`public void handleSave();`

- 현재 mainApp의 getPersonFilePath를 호출하여 OS 레지스트리에 저장된 마지막 xml 파일 반환 (OS 레지스트리에 저장된 마지막 xml 파일이 현재 ContactApp에서 작업 중인 xml 파일임)
- 반환된 filaPath에 해당하는 파일 경로에 현재 Table의 data를 xml로 마샬링하여 저장 (mainApp.setPersonFilePath)
- filePath를 반환받지 못했을 땐, handleSaveAs() 호출

2-2) Save As






```
public void handleSaveAs();
```

- java의 fileChooser 객체를 이용해 window 파일 탐색기를 실행시켜 filePath를 반환받음
- 반환받은 filePath에 현재 Table의 data를 xml로 마샬링하여 저장

```
file.getPath().endsWith(".xml")
file = new File(file.getPath() + ".xml");
```

- filePath는 .xml로 끝나야 하며, filePath 마지막에 .xml이 없을 경우, 추가하여 저장