KB금융그룹 | 국민의 평생 금융파트너

2024년 상반기 K-디지털 트레이닝

ScoulaTodo – 로그인/로그아웃/회원가입

[KB] IT's Your Life

KB 국민은행

프로젝트 생성

- Name: ScoulaTodo
- Build System: Gradle

settings.gradle

```
rootProject.name = 'ScoulaTodo'
include ':ScoulaCli'
project(':ScoulaCli').projectDir=new File('C:\\KB_Fullstack\\04_Java\\ScoulaLib\\scoulacli')
```

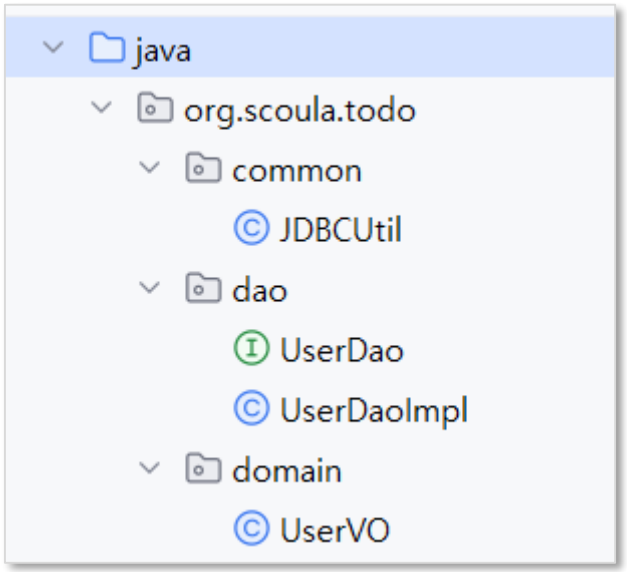
build.gradle

```
dependencies {
    implementation project(':ScoulaCli')

    implementation 'com.mysql:mysql-connector-j:8.3.0'
    compileOnly 'org.projectlombok:lombok:1.18.30'
    annotationProcessor 'org.projectlombok:lombok:1.18.30'

    testCompileOnly 'org.projectlombok:lombok:1.18.30'
    testAnnotationProcessor 'org.projectlombok:lombok:1.18.30'

    testImplementation platform('org.junit:junit-bom:5.10.0')
    testImplementation 'org.junit.jupiter:junit-jupiter'
}
```

- 수정 후 Sync 실행
- 프로젝트 설정
  - Annotation Processing 활성화

- IntelliJ DataSource 등록
- 프로젝트 루트에 todo.sql 파일 추가

기존 코드에서 가져오기



- 기존 코드에서 해당 클래스 가져오기
  - 복사 시 패키지 조정 필요할 수 있음

UserVO.java

```
package org.scoula.todo.domain;

import lombok.AllArgsConstructor;
import lombok.Builder;
import lombok.Data;
import lombok.NoArgsConstructor;

@Data
@NoArgsConstructor
@AllArgsConstructor
@Builder
public class UserVO {
    private String id;
    private String password;
    private String name;
    private String role;
}
```

구현할 기능

- 회원가입
- 로그인
- 로그아웃
- 종료

- 로그인 상태에 따라 보여주는 메뉴가 달라져야 함.
  - 기존 createMenu() 대신 다른 방식으로 메뉴 운영

  - App 클래스에 void setMenu(Menu menu) 메서드 추가

App.java

```
public abstract class App {
    Menu menu;

    public App() {
    }

    public void setMenu(Menu menu) {
        this.menu = menu;
    }
  …
}
```

메뉴 구성하기

annoymousMenu

- 로그인하지 않은 상태 때 운영

```
------------------------------------------------------------
1.로그인 |2.가입 |3.종료 |
------------------------------------------------------------
```

userMenu

- 로그인 한 상태 때 운영

```
------------------------------------------------------------
1.로그아웃 |2.종료 |
------------------------------------------------------------
```

TodoApp.java

```
public class TodoApp extends App {
    Menu userMenu;      // 로그인한 상태의 메뉴
    Menu anonymousMenu; // 로그아웃한 상태의 메뉴

    @Override
    public void init() {
        anonymousMenu = new Menu();
        anonymousMenu.add(new MenuItem("로그인", this::login));
        anonymousMenu.add(new MenuItem("가입", this::join));
        anonymousMenu.add(new MenuItem("종료", this::exit));

        userMenu = new Menu();
        userMenu.add(new MenuItem("로그아웃", this::logout));
        userMenu.add(new MenuItem("종료", this::exit));

        setMenu(anonymousMenu); // 시작은 annonymousMenu로
    }
```

TodoApp.java

```
    public void join(){

    }

    public void login() {
        setMenu(userMenu);   // 메뉴 교체
    }

    public void logout() {
        if(Input.confirm("로그아웃 할까요?")) {
            setMenu(anonymousMenu); // 메뉴 교체
        }
    }

    public static void main(final String[] args) {
        final TodoApp app = new TodoApp();
        app.run();
    }
}
```

```
---------------------------------------------------------
1.로그인 |2.가입 |3.종료 |
---------------------------------------------------------
선택> 1
---------------------------------------------------------
1.로그아웃 |2.종료 |
---------------------------------------------------------
선택> 1
로그아웃 할까요? (Y/n):
---------------------------------------------------------
1.로그인 |2.가입 |3.종료 |
---------------------------------------------------------
선택>
```

회원가입

- 사용자 ID, 비밀번호, 비밀번호2, 이름, 역할 입력 받아 DB에 등록
- 예외 상황
  - 사용자 ID 중복 → UsernameDuplicateException 예외 발생
  - 비밀번호, 비밀번호2 불일치 → PasswordMissmatchException 예외 발생

```
------------------------------------------------------------
1.로그인 |2.가입 |3.종료 |
------------------------------------------------------------
선택> 2
사용자 ID: hong
이미 사용중인 사용자 ID 입니다.
```

```
------------------------------------------------------------
1.로그인 |2.가입 |3.종료 |
------------------------------------------------------------
선택> 2
사용자 ID: kim
비밀번호: 1234
비밀번호 확인: 123
비밀번호가 맞지 안습니다.
```

```
------------------------------------------------------------
1.로그인 |2.가입 |3.종료 |
------------------------------------------------------------
선택> 2
사용자 ID: lee
비밀번호: 1234
비밀번호 확인: 1234
이름: 이세돌
역할: 관리자
```

UsernameDuplicateException.java

```
package org.scoula.todo.exception;

public class UsernameDuplicateException extends Exception{
    public UsernameDuplicateException(){
        super("이미 사용중인 사용자 ID 입니다.");
    }
}
```

PasswordMissmatchException.java

```
public class PasswordMissmatchException extends Exception{
    public PasswordMissmatchException(){
        super("비밀번호가 맞지 안습니다.");
    }
}
```

AccountService.java

```
package org.scoula.todo.service;
…
public class AccountService {
    UserDao dao = new UserDaoImpl();

    public void join() {
        try {
            UserVO user = getUser();
            dao.create(user);
        } catch (UsernameDuplicateException ¦ PasswordMissmatchException e) {
            System.out.println(e.getMessage());
        } catch (Exception e) {
            throw new RuntimeException(e);
        }
    }

    public boolean checkDuplication(String username) throws UsernameDuplicateException, SQLException {
        Optional<UserVO> result = dao.get(username);
        if(result.isPresent()) {
            throw new UsernameDuplicateException();
        }
        return false;
    }
```

AccountService.java

```
    private UserVO getUser() throws SQLException, UsernameDuplicateException, PasswordMissmatchException {
        String username = Input.getLine("사용자 ID: ");
        checkDuplication(username);

        String password = Input.getLine("비밀번호: ");
        String password2 = Input.getLine("비밀번호 확인: ");
        if(!password.equals(password2)) {
            throw new PasswordMissmatchException();
        }

        String name= Input.getLine("이름: ");
        String role = Input.getLine("역할: ");

        return UserVO.builder()
                .id(username)
                .password(password)
                .name(name)
                .role(role)
                .build();
    }
}
```

TodoApp.java

```
public class TodoApp extends App {
    …
    AccountService accountService = new AccountService();

    …

    @Override
    public void init() {
        anonymousManu = new Menu();
        anonymousManu.add(new MenuItem("로그인", this::login));
        anonymousManu.add(new MenuItem("가입", accountService::join));
        anonymousManu.add(new MenuItem("종료", this::exit));
        …
    }
    …
}
```

로그인

- 로그인 사용자(User) 정보는 어플리케이션 전체에서 접근할 수 있어야 함
  - Context 객체를 싱글톤으로 운영
  - User 정보는 Context 객체를 통해 접근

- 사용자 ID와 비밀번호를 입력 받아 DB의 내용과 일치하는지 체크
- 로그인에 성공하면
  - 로그인 사용자 정보 user 설정
  - 메뉴를 userMenu로 교체

- 예외 상황
  - 사용자 ID가 DB에 없는 경우
  - 비밀번호가 DB의 비밀번호와 맞지않는 경우

  → LoginFailException 발생

```
------------------------------------------------------------
1.로그인 |2.가입 |3.종료 |
------------------------------------------------------------
선택> 1
사용자 ID: park
비밀번호: 12
사용자 ID 또는 비밀번호가 일치하지 않습니다.
```

Context.java

```
package org.scoula.todo.context;

import lombok.Getter;
import lombok.Setter;
import org.scoula.todo.domain.UserVO;

@Getter
@Setter
public class Context {
    private UserVO user;  // 로그인 사용자 정보, null이면 로그아웃 상태

    private Context() {}

    private static Context context = new Context();

    public static Context getContext() {
        return context;
    }

}
```

LoginFailException.java

```
package org.scoula.todo.exception;

public class LoginFailException extends Exception{
    public LoginFailException(){
        super("사용자 ID 또는 비밀번호가 일치하지 않습니다.");
    }
}
```

LoginService.java

```
package org.scoula.todo.service;

import org.scoula.lib.cli.ui.Input;
import org.scoula.todo.context.Context;
import org.scoula.todo.dao.UserDao;
import org.scoula.todo.dao.UserDaoImpl;
import org.scoula.todo.domain.UserVO;
import org.scoula.todo.exception.LoginFailException;

import java.sql.SQLException;

public class LoginService {
    UserDao dao = new UserDaoImpl();

    public void login() throws SQLException, LoginFailException {
        String username = Input.getLine("사용자 ID: ");
        String password = Input.getLine("비밀번호: ");
        UserVO user = dao.get(username).orElseThrow(LoginFailException::new);
        if(user.getPassword().equals(password)) {
            Context ctx = Context.getContext();
            ctx.setUser(user);
        } else {
            throw new LoginFailException();
        }
    }
}
```

TodoApp.java

```
public class TodoApp extends App {
    …
    LoginService loginService = new LoginService();
    AccountService accountService = new AccountService();
    …

    public void login() {
        try {
            loginService.login();
            setMenu(userMenu);
        } catch (SQLException e) {
            throw new RuntimeException(e);
        } catch (LoginFailException e) {
            System.out.println(e.getMessage());
        }
    }

…
```

로그아웃

- 로그인 사용자 정보를 null로 설정하고,
- 메뉴를 anonymousMenu로 교체

TodoApp.java

```
public class TodoApp extends App {
    …

    public void logout() {
        if(Input.confirm("로그아웃 할까요?")) {
            Context.getContext().setUser(null);
            setMenu(anonymousManu);
        }
    }
    …
```