KB금융그룹 | 국민의 평생 금융파트너

2024년 상반기 K-디지털 트레이닝

ScoulaTodo – Todo

[KB] IT's Your Life

KB국민은행

사용자별 Todo 관리

데이터베이스 Todo 테이블 준비

todo.sql

```
CREATE TABLE TODO (
  ID          INTEGER AUTO_INCREMENT NOT NULL PRIMARY KEY,
  TITLE       VARCHAR(128) NOT NULL,
  DESCRIPTION VARCHAR(512) NULL,
  DONE        BOOLEAN,
  USERID     VARCHAR(12) NOT NULL,
  FOREIGN KEY (USERID) REFERENCES USERS(ID)
     ON UPDATE CASCADE
     ON DELETE CASCADE
);

INSERT INTO todo(title, description, done, userid)
values ('야구장', '프로야구 경기도 봐야합니다.', false, 'guest'),
       ('놀기', '노는 것도 중요합니다.', false, 'guest'),
       ('Vue 학습', 'Vue 학습을 해야 합니다', false, 'member'),
       ('ES6 공부', 'ES6공부를 해야 합니다', true, 'guest');
```

구현할 기능

- 로그인 한 상태에서
  - 목록
  - 검색
  - 상세
  - 추가
  - 수정
  - 삭제

```
------------------------------------------------------------
1.로그인 |2.가입 |3.종료 |
------------------------------------------------------------
선택> 1
사용자 ID: guest
비밀번호: guest123
------------------------------------------------------------
1.목록 |2.검색 |3.상세 |4.추가 |5.수정 |6.삭제 |7.로그아웃 |8.종료 |
------------------------------------------------------------
선택>
```

TodoApp.java

```
@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));

    userMenu = new Menu();
    userMenu.add(new MenuItem("목록", null));
    userMenu.add(new MenuItem("검색", null));
    userMenu.add(new MenuItem("상세", null));
    userMenu.add(new MenuItem("추가", null));
    userMenu.add(new MenuItem("수정", null));
    userMenu.add(new MenuItem("삭제", null));

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

    setMenu(anonymousManu);
}
```

TodoVO

- Todo 테이블에 대한 모델 객체
- 빌더 패턴 적용

TodoApp.java

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

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

@Data
@NoArgsConstructor
@AllArgsConstructor
@Builder
public class TodoVO {
    private Long id;
    private String title;
    private String description;
    private Boolean done;
    private String userId;
}
```

TodoDao

- 인터페이스 정의 후 구현
  - 사용자 ID를 인자로 항상 전달

TodoDao.java

```
package org.scoula.todo.dao;
…
public interface TodoDao {
    // 총 데이터 건수
    int getTotalCount(String userId) throws SQLException;

    // Todo 등록
    int create(TodoVO todo) throws SQLException;

    // Todo 목록 조회
    List<TodoVO> getList(String userId) throws SQLException;

    // Todo 정보 조회
    Optional<TodoVO> get(String userId, Long id) throws SQLException;

    // 검색
    List<TodoVO> search(String userId, String keyword) throws SQLException;

   // Todo 수정
    int update(String userId, TodoVO todo) throws SQLException;

    // Todo 삭제
    int delete(String userId, Long id) throws SQLException;
}
```

TodoDao.java

```
package org.scoula.todo.dao;
…
public class TodoDaoImpl implements TodoDao{
    Connection conn = JDBCUtil.getConnection();

    @Override
    public int getTotalCount(String userId) throws SQLException {
        String sql = "select count(*) from todo where userid=?";
        try (PreparedStatement stmt = conn.prepareStatement(sql)) {
            stmt.setString(1, userId);
            try(ResultSet rs = stmt.executeQuery()) {
                rs.next();
                return rs.getInt(1);
            }
        }
    }
```

TodoDao.java

```
@Override
public int create(TodoVO todo) throws SQLException {
    String sql = "insert into todo(title, description, done, userid) values(?,?,?,?)";
    try (PreparedStatement stmt = conn.prepareStatement(sql)) {
        stmt.setString(1, todo.getTitle());
        stmt.setString(2, todo.getDescription());
        stmt.setBoolean(3, todo.getDone());
        stmt.setString(4, todo.getUserId());
        return stmt.executeUpdate();
    }
}
```

TodoDao.java

```
private TodoVO map(ResultSet rs) throws SQLException {
    TodoVO todo = new TodoVO();
    todo.setId(rs.getLong("id"));
    todo.setTitle(rs.getString("title"));
    todo.setDescription(rs.getString("description"));
    todo.setDone(rs.getBoolean("done"));
    todo.setUserId(rs.getString("userid"));
    return todo;
}

private List<TodoVO> mapList(ResultSet rs) throws SQLException {
    List<TodoVO> todoList = new ArrayList<TodoVO>();
    while (rs.next()) {
        TodoVO todo = map(rs);
        todoList.add(todo);
    }
    return todoList;
}
```

TodoDao.java

```
@Override
public List<TodoVO> getList(String user) throws SQLException {
    String sql = "select * from todo where userId = ?";
    try (PreparedStatement stmt = conn.prepareStatement(sql)){
        stmt.setString(1, user);
        try(ResultSet rs = stmt.executeQuery()) {
            return mapList(rs);
        }
    }
}

@Override
public Optional<TodoVO> get(String userId, Long id) throws SQLException {
    String sql = "select * from todo where userId = ? and id = ?";
    try (PreparedStatement stmt = conn.prepareStatement(sql)) {
        stmt.setString(1, userId);
        stmt.setLong(2, id);
        try(ResultSet rs = stmt.executeQuery()) {
            if(rs.next()) {
                return Optional.of(map(rs));
            }
        }
    }
    return Optional.empty();
}
```

TodoDao.java

```
@Override
public List<TodoVO> search(String userId, String keyword) throws SQLException {
    String sql = "select * from todo where userId = ? and (title like ? or description like ?)";
    try (PreparedStatement stmt = conn.prepareStatement(sql)){
        stmt.setString(1, userId);
        stmt.setString(2, keyword);
        stmt.setString(3, keyword);
        try(ResultSet rs = stmt.executeQuery()) {
            return mapList(rs);
        }
    }
}
```

TodoDao.java

```
    @Override
    public int update(String userId, TodoVO todo) throws SQLException {
        String sql = "update todo set title = ?, description = ?, done = ? where userId =? and id = ?";
        try ( PreparedStatement stmt = conn.prepareStatement(sql)) {
            stmt.setString(1, todo.getTitle());
            stmt.setString(2, todo.getDescription());
            stmt.setBoolean(3, todo.getDone());
            stmt.setString(4, userId);
            stmt.setLong(5, todo.getId());
            stmt.executeUpdate();
        }
        return 0;
    }

    @Override
    public int delete(String userId, Long id) throws SQLException {
        String sql = "delete from todo where userId = ? and id = ?";
        try(PreparedStatement stmt = conn.prepareStatement(sql)) {
            stmt.setString(1, userId);
            stmt.setLong(2, id);
            stmt.executeUpdate();
        }
        return 0;
    }
}
```

TodoTest.java를 작성하여 단위 테스트 진행

TodoService 작성

- 각각의 기능을 구현 후 메뉴에 추가
- 로그인 사용자 정보는 컨텍스트를 통해 획득
  - 람다식을 이용해 표현식 간소화
  - Supplier 인터페이스 활용
    - Supplier<T>
      T void get()

TodoService.java

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

import org.scoula.todo.context.Context;
import java.util.function.Supplier;

public class TodoService {
    TodoDao dao = new TodoDaoImpl();

    Supplier<String> userId = ()->Context.getContext().getUser().getId();

}
```

목록 보기

- 로그인 사용자의 목록 만 추출
- 총 데이터 건수 출력

```
------------------------------------------------------------
1.목록 |2.검색 |3.상세 |4.추가 |5.수정 |6.삭제 |7.로그아웃 |8.종료 |
------------------------------------------------------------
선택> 1
총 건수: 3
==========================================
001] 야구장
002] 놀기
004] ES6 공부(완료)
==========================================
```

TodoService.java

```
public void print() {
    try {
        int count = dao.getTotalCount(userId.get());
        List<TodoVO> list = dao.getList(userId.get());

        System.out.println("총 건수: " + count);
        System.out.println("========================================");
        for (TodoVO todo : list) {
            System.out.printf("%03d] %s%s\n", todo.getId(), todo.getTitle(),
                    todo.getDone() ? "(완료)" : "");
        }
        System.out.println("========================================");

    } catch (SQLException e) {
        throw new RuntimeException(e);
    }
}
```

TodoApp.java

```
@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));

    userMenu = new Menu();
    userMenu.add(new MenuItem("목록", todoService::print));
    …
```

검색

- 입력 받은 keyword를 title과 description 컬럼에서 검색
- keyword는 '%keyword%' 패턴 문자열

```
------------------------------------------------------------
1.목록 |2.검색 |3.상세 |4.추가 |5.수정 |6.삭제 |7.로그아웃 |8.종료 |
------------------------------------------------------------
선택> 2
검색어: 놀기
총 건수: 1
==========================================
002] 놀기
==========================================
```

TodoService.java

```
public void search() {
    String keyword = Input.getLine("검색어: ");
    keyword = "%" + keyword + "%";
    try {
        List<TodoVO> list = dao.search(userId.get(), keyword);
        System.out.println("총 건수: " + list.size());

        System.out.println("========================================");
        for (TodoVO todo : list) {
            System.out.printf("%03d] %s%s\n", todo.getId(), todo.getTitle(),
                    todo.getDone() ? "(완료)" : "");
        }
        System.out.println("========================================");

    } catch (SQLException e) {
        throw new RuntimeException(e);
    }
}
```

TodoApp.java

```
@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));

    userMenu = new Menu();
    userMenu.add(new MenuItem("목록", todoService::print));
    userMenu.add(new MenuItem("검색", todoService::search));

    …
```

상세보기

상세 보기할 id 선택

```
------------------------------------------------------------
1.목록 |2.검색 |3.상세 |4.추가 |5.수정 |6.삭제 |7.로그아웃 |8.종료 |
------------------------------------------------------------
선택> 3
Todo ID: 2
==========================================
Todo ID : 2
제목    : 놀기
설명    : 노는 것도 중요합니다.
완료여부 : 미완료
==========================================
```

TodoService.java

```
public void detail() {
    long id = (long)Input.getInt("Todo ID: ");
    try {
        TodoVO todo = dao.get(userId.get(), id).orElseThrow(NoSuchElementException::new);
        System.out.println("========================================");
        System.out.printf("Todo ID : %s\n", todo.getId());
        System.out.printf("제목    : %s\n", todo.getTitle());
        System.out.printf("설명    : %s\n", todo.getDescription());
        System.out.printf("완료여부 : %s\n", todo.getDone() ? "완료" : "미완료");
        System.out.println("========================================");
    } catch (SQLException e) {
        throw new RuntimeException(e);
    }
}
```

TodoApp.java

```
@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));

    userMenu = new Menu();
    userMenu.add(new MenuItem("목록", todoService::print));
    userMenu.add(new MenuItem("검색", todoService::search));
    userMenu.add(new MenuItem("상세", todoService::detail));
    …
```

추가

```
------------------------------------------------------------
1.목록 |2.검색 |3.상세 |4.추가 |5.수정 |6.삭제 |7.로그아웃 |8.종료 |
------------------------------------------------------------
선택> 4
제목: JDBC 연동
설명: Java와 MySQL 데이터베이스 연동 학습
------------------------------------------------------------
1.목록 |2.검색 |3.상세 |4.추가 |5.수정 |6.삭제 |7.로그아웃 |8.종료 |
------------------------------------------------------------
선택> 1
총 건수: 4
============================================
001] 야구장
002] 놀기
004] ES6 공부(완료)
007] JDBC 연동
============================================
```

TodoService.java

```
public void create() {
    String title = Input.getLine("제목: ");
    String description = Input.getLine("설명: ");
    TodoVO todo = TodoVO.builder()
            .title(title)
            .description(description)
            .userId(userId.get())
            .done(false)
            .build();
    try {
        dao.create(todo);
    } catch (SQLException e) {
        throw new RuntimeException(e);
    }
}
```

TodoApp.java

```
@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));

    userMenu = new Menu();
    userMenu.add(new MenuItem("목록", todoService::print));
    userMenu.add(new MenuItem("검색", todoService::search));
    userMenu.add(new MenuItem("상세", todoService::detail));
    userMenu.add(new MenuItem("추가", todoService::create));

    …
```

수정

```
------------------------------------------------------------
1.목록 |2.검색 |3.상세 |4.추가 |5.수정 |6.삭제 |7.로그아웃 |8.종료 |
------------------------------------------------------------
선택> 5
수정할 Todo ID: 7
==========================================
제목(JDBC 연동): JDBC연동 학습
설명(Java와 MySQL 데이터베이스 연동 학습):
완료 (y/N):
==========================================
```

TodoService.java

```
public void update() {
    Long id = (long)Input.getInt("수정할 Todo ID: ");
    try {
        TodoVO todo = dao.get(userId.get(), id).orElseThrow(NoSuchElementException::new);
        System.out.println("========================================");
        String title = Input.getLine("제목", todo.getTitle());
        String description = Input.getLine("설명", todo.getDescription());
        boolean done = Input.confirm("완료", todo.getDone());
        System.out.println("========================================");

        todo.setTitle(title);
        todo.setDescription(description);
        todo.setDone(done);

        dao.update(userId.get(), todo);
    } catch (SQLException e) {
        throw new RuntimeException(e);
    }
}
```

TodoApp.java

```
@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));

    userMenu = new Menu();
    userMenu.add(new MenuItem("목록", todoService::print));
    userMenu.add(new MenuItem("검색", todoService::search));
    userMenu.add(new MenuItem("상세", todoService::detail));
    userMenu.add(new MenuItem("추가", todoService::create));
    userMenu.add(new MenuItem("수정", todoService::update));

    …
```

삭제

```
------------------------------------------------------------
1.목록 |2.검색 |3.상세 |4.추가 |5.수정 |6.삭제 |7.로그아웃 |8.종료 |
------------------------------------------------------------
선택> 6
삭제할 Todo ID: 7
삭제할까요? (Y/n):
------------------------------------------------------------
1.목록 |2.검색 |3.상세 |4.추가 |5.수정 |6.삭제 |7.로그아웃 |8.종료 |
------------------------------------------------------------
선택> 1
총 건수: 3
============================================
001] 야구장
002] 놀기
004] ES6 공부(완료)
============================================
```

TodoService.java

```
public void delete() {
    Long id = (long)Input.getInt("삭제할 Todo ID: ");
    boolean answer = Input.confirm("삭제할까요?");
    if (answer) {
        try {
            dao.delete(userId.get(), id);
        } catch (SQLException e) {
            throw new RuntimeException(e);
        }
    }
}
```

TodoApp.java

```
@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));

    userMenu = new Menu();
    userMenu.add(new MenuItem("목록", todoService::print));
    userMenu.add(new MenuItem("검색", todoService::search));
    userMenu.add(new MenuItem("상세", todoService::detail));
    userMenu.add(new MenuItem("추가", todoService::create));
    userMenu.add(new MenuItem("수정", todoService::update));
    userMenu.add(new MenuItem("삭제", todoService::delete));
    …
```