KB금융그룹 | 국민의 평생 금융파트너

2024년 상반기 K-디지털 트레이닝

제너릭

[KB] IT's Your Life

KB국민은행

제너릭

- **mywork 프로젝트의 mylib 모듈에 CrudDao 인터페이스를 정의하세요.**
  - 패지키: org.scoula.lib.dao
  - 제너릭
    - K: 키 필드 제너릭 타입
    - E: 데이터 엘리먼트 타입
  - 메서드
    - list()
      - 데이터 목록을 List로 리턴
    - get()
      - 키값을 매개변수로 입력 받아 해당 데이터를 리턴
    - create()
      - 저장할 데이터를 매개변수로 입력 받음
      - 리턴값 없음
    - update()
      - 해당 데이터의 키값과 수정할 데이터를 매개변수로 입력 받음
      - 리턴값 없음
    - delete()
      - 삭제할 데이터의 키값을 매개변수로 입력 받음
      - 리턴값 없음

CrudDao.java

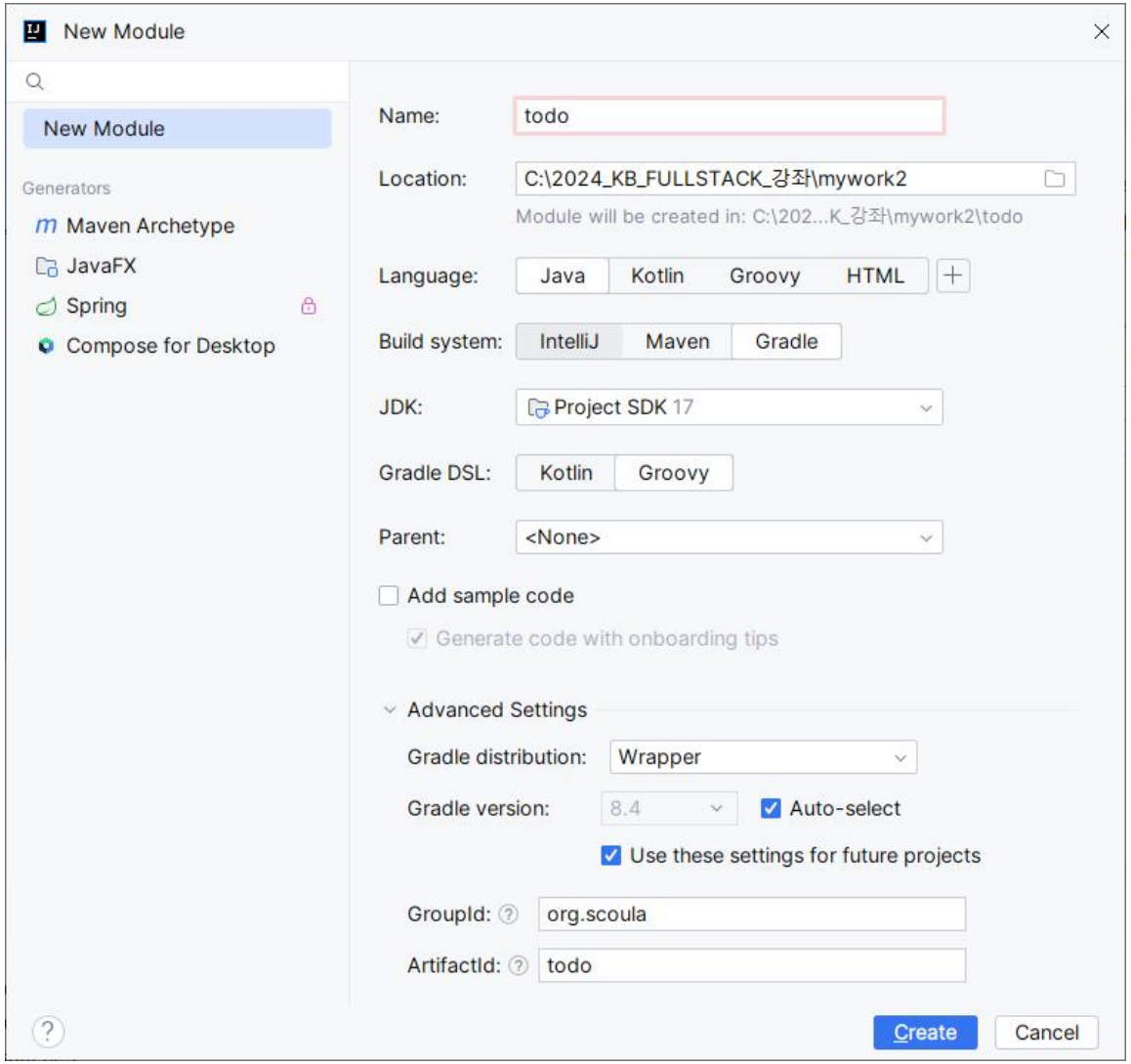
```
package org.scoula.lib.dao;

import java.util.List;

public interface CrudDao<K, E> {
    List<E> list();
    E get(K k);
    void create(E e);
    void update(K k, E e);

    void delete(K k);
}
```

제너릭

- **mywork 프로젝트에 todo 모듈을 추가하세요.**
  - 모듈명: todo
  - Build system: Gradle
  - Gradle DSL: Groovy
  - Advanced Settings:
    - GroupId: org.scoula
    - ArtifactId: todo

제너릭

- **mywork 프로젝트에 todo 모듈을 추가하세요.**
  - 모듈명: todo
  - Build system: Gradle
  - Gradle DSL: Groovy
  - Advanced Settings:
    - GroupId: org.scoula
    - ArtifactId: todo



다음 절차로 기존 mylib를 todo의 의존 모듈로 추가하세요.

settings.gradle(todo)

```
rootProject.name = 'todo'

include ':mylib'
project(':mylib').projectDir=new File('C:\\2024_KB_FULLSTACK_강좌\\mywork2\\mylib')
```

build.gradle(todo) → 수정 후 gradle sync

```
…
dependencies {
    testImplementation platform('org.junit:junit-bom:5.9.1')
    testImplementation 'org.junit.jupiter:junit-jupiter'

    compileOnly 'org.projectlombok:lombok:1.18.32'
    annotationProcessor 'org.projectlombok:lombok:1.18.32'
    implementation project(':mylib')
}
…
```

제너릭

- **다음 Todo 클래스를 롬복을 이용하여 데이터 클래스로 정의하세요.**
  - 패키지: org.scoula.todo.domain
  - 기본 생성자
  - 모든 매개변수를 가지는 생성자
  - 빌더 패턴적용

  - cf)
    - lombok은 build.gradle에 설정
    - 프로젝트 설정에서 어노테이션 프로세서 기능 활성화

Todo.java

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

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

@Data
@NoArgsConstructor
@AllArgsConstructor
@Builder
public class Todo {
    Integer id;
    String title;
    String desc;
    boolean done;
}
```

제너릭

- **다음 조건을 만족하는 TodoDao 클래스를 정의하세요.**
  - 패키지: org.scoula.todo.dao
  - CrudDao 구현
    - 제너릭
      - K: Integer
      - E: Todo
    - 메서드의 내용은 디폴트 자동 생성 코드로 함

TodoDao.java

```
package org.scoula.todo.dao;

import org.scoula.lib.dao.CrudDao;
import org.scoula.todo.domain.Todo;

import java.util.List;

public class TodoDao implements CrudDao<Integer, Todo> {

    @Override
    public List<Todo> list() {
        return null;
    }

   @Override
    public Todo get(Integer integer) {
        return null;
    }
```

TodoDao.java

```
    @Override
    public void create(Todo todo) {

    }

    @Override
    public void update(Integer integer, Todo todo) {

    }

    @Override
    public void delete(Integer integer) {

    }
}
```

제너릭

- 앞에서 정의한 TodoDao 클래스를 Singleton 객체로 재 정의하세요.

TodoDao.java

```
package org.scoula.todo.dao;

import org.scoula.lib.dao.CrudDao;
import org.scoula.todo.domain.Todo;

import java.util.List;

public class TodoDao implements CrudDao<Integer, Todo> {

    private static TodoDao instance = new TodoDao();

    private TodoDao() {
    }

    public static TodoDao getInstance() {
        return instance;
    }
   …
}
```