

## 데이터베이스프로그래밍

### 실습 #4 (DAO, DTO)

실습 #3의 결과를 참조하여 동일한 기능을 수행하는 프로그램을 아래와 같이 DTO와 DAO를 이용해서 구현하시오.

1. **Department** DTO 클래스: 부서번호, 부서명, 관리자사번, 소속사원수 등 부서 정보를 포함

```
public class Department {
    // 필드: 부서번호, 부서명, 관리자사번, 소속사원수
    // default 생성자와, 위 필드들에 대한 getter & setter method 정의 (Source > Generate
    getters and setters 메뉴 이용)
    // 위 필드들을 초기화하는 생성자를 정의해서 활용 가능 (Source > Generate Constructor using
    fields 메뉴 이용)
}
```

2. **Employee** DTO 클래스: 사번, 이름, 직무, 입사일, 월급, 수당, 소속부서명 등 사원 정보를 포함

```
public class Employee {
    // 필드: 사번, 이름, 직무, 입사일, 월급, 수당, 소속부서명
    // default 생성자와, 위 필드들에 대한 getter & setter method 정의
    // 위 필드들을 초기화하는 생성자를 정의해서 활용 가능
    // 모든 필드 값들을 출력하기 위한 toString() 정의 (Source > Generate toString() 메뉴 이용)
}
```

3. **Appointment** DTO 클래스: 부서번호, 새 관리자 사번, 새 관리자 보직수당 등 부서 관리자 임명에 필요한 데이터를 포함 (위 클래스와 유사하게 정의)

4. **CompanyDAO** 클래스: 네 가지 메소드를 포함하되 위 DTO를 이용하여 데이터를 전달/반환함

```
public class CompanyDao {
    private JDBCUtil jdbcUtil = null;    // JDBCUtil 클래스 활용
    public CompanyDao() {
        jdbcUtil = new JDBCUtil();
    }
    public Department findDeptByName(String deptName) {
        // 실습 #3의 printDeptInfo()를 변형
        // 부서정보 검색 후 Department DTO를 생성하고 검색 결과를 저장해서 반환함
        // (검색 결과가 없을 경우 null을 반환)
    }
    public List<Employee> getAllEmpsInDept(int deptNo) {
        // 실습 #3의 printAllEmpsInDept()를 변형
        // ArrayList<Employee> 타입 객체를 생성한 후, 검색된 각 사원 정보에 대해
        // Employee DTO를 생성 및 저장하고 List 객체에 추가함. List 객체를 결과로 반환함
    }
}
```

```

public Employee replaceManagerOfDept(Appointment appo) {
    // 실습 #3 의 replaceManagerOfDept()와 유사하나, 매개변수 전달을 위해 Appointment
    // DTO 를 이용하고, 실행 결과로 기존 관리자 사원 정보를 포함하는 Employee DTO 를 반환함
}

public Employee findEmployee(int empNo) {
    // 실습 #3 의 printEmpInfo()를 변형
    // 주어진 사번에 해당하는 사원 정보를 검색 후 Employee DTO 를 생성 및 반환함
    // (검색 결과가 없을 경우 null 을 반환)
} }

```

4. **Lab4** 클래스: 위에 정의된 DTO와 DAO를 이용해서 실습 #3와 동일한 기능을 수행

```

public class Lab4 {
    private static CompanyDao compDao = new CompanyDao();
    public static void main(String[] args) {
        ...
        Department dept = compDao.findDeptByName(deptName);
        // dept 객체의 필드 값 출력

        List<Employee> empList = compDao.getAllEmpsInDept( /*부서번호*/ );
        // empList 에 포함된 모든 Employee 객체들의 필드 값을 출력:
        //   Employee 객체들을 하나씩 접근하기 위해 empList 로부터 Iterator 를 구해서 활용
        Iterator<Employee> iter = empList.iterator();
        while (iter.hasNext()) {
            Employee emp = iter.next();
            ...
        }
        ...
        Appointment appo = new Appointment( /*부서번호, 새 관리자 사번, 보직 수당*/ );
        Employee oldMgr = compDao.replaceManagerOfDept(appo);
        // oldMgr 의 모든 필드 값 출력 (toString() 이용)
        Employee newMgr = compDao.findEmployee( /*새 관리자 사번*/ );
        // newMgr 의 모든 필드 값 출력 (toString() 이용)
    }
}

```

**실행 예:** 실습 #3와 동일

**제출기한:** 11월 11일(월) 13:30pm

**제출방법:** 소스 코드와 실행 결과(텍스트 또는 capture image)를 과제 게시글로 작성함  
(실습 #4 게시글의 <과제 제출 시 주의사항> 참조)