배기원

스마트 콘텐츠 융합 응용SW 엔지니어 양성과정(888H)
[강남 M] 2020. 12. 02 ~ 2021. 07. 08 15:30~22:00
김동현 강사 | 진은지 취업담임

남은 시간 10: 19: 43

수강생 평가

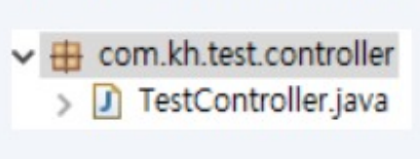
* 첨부파일의 확장자를 소문자로 등록하세요. - 예시 : test.jpg(O), test.JPG(X)

[NCS전공교과] 응용 SW 기초 기술 활용 (평가자체크리스트) 1회차 ⌄

총점 : 90.0

다음 요구사항들에 맞게 프로젝트를 개발하시오

다음 이미지의 TestController.java 는 Servlet을 상속받은 클래스이다. web.xml에서 해당 클래스를 Servlet으로 등록하는 태그를 작성하시오.



```
⌄ com.kh.test.controller
  > TestController.java
```

요구사항

해당 servlet-name은 'TestServlet'으로 작성하시오

수강생 답

```
<servlet>
    <servlet-name>TestServlet </servlet-name>
    <servlet-class>com.kh.test.controller.TestController</servlet-class>
 </servlet>
```

답안첨삭

```
<servlet>
    <servlet-name>TestServlet </servlet-name>
    <servlet-class>com.kh.test.controller.TestController</servlet-class>
 </servlet>
```

요구사항에 맞는 servlet 설정 tag를 명확하게 작성함

모범답안

```
<servlet>
<servlet-name>TestServlet</servlet-name>
<servlet-class>com.kh.test.controller.TestController</servlet-class>
</servlet>
```

요구사항 1번에서 작성한 Servlet 클래스를 mapping하는 태그를 작성하시오.

요구사항

mapping되는 url은 'test.do'로 작성하시오

수강생 답

```
<servlet-mapping>
 <servlet-name>TestServlet</servlet-name>
 <url-pattern>/test.do</url-pattern>
 </servlet-mapping>
```

답안첨삭

```
<servlet-mapping>
 <servlet-name>TestServlet</servlet-name>
 <url-pattern>/test.do</url-pattern>
 </servlet-mapping>
```

요구사항에 맞는 servlet mapping 설정 tag를 명확하게 작성함

모범답안

```
<servlet-mapping>
<servlet-name>TestServlet</servlet-name>
<url-pattern>/test.do</url-pattern>
</servlet-mapping>
```

다음 SQL 문으로 생성되는 table에 저장된 rows의 값을 저장할 수 있는 클래스를 작성하시오.

```
CREATE TABLE TEST(
SEQ NUMBER,
WRITER VARCHAR2(30),
TITLE VARCHAR2(100),
CONTENT VARCHAR2(200),
REGDATE DATE
);
```

요구사항

클래스의 이름은 table의 이름과 같도록 하고, 패키지는 com.kh.test.model로 하시오.
기본 생성자와 파라미터 5개 생성자, 모든 필드에 대한 getter/setter를 작성하시오.

수강생 답

```
package com.kh.test.model;

import java.sql.Date;

public class table {
    private  int SEQ;
    private  String WRITER;
    private  String  TITLE;
    private String CONTENT;
    private Date  REGDATE;
    public table() {
        super();
        // TODO Auto-generated constructor stub
    }
    public table(int sEQ, String wRITER, String tITLE, String cONTENT, Date rEGDATE) {
        super();
        SEQ = sEQ;
        WRITER = wRITER;
        TITLE = tITLE;
        CONTENT = cONTENT;
        REGDATE = rEGDATE;
    }
    public int getSEQ() {
        return SEQ;
    }
    public void setSEQ(int sEQ) {
        SEQ = sEQ;
    }
    public String getWRITER() {
        return WRITER;
    }
    public void setWRITER(String wRITER) {
        WRITER = wRITER;
    }
    public String getTITLE() {
        return TITLE;
    }
    public void setTITLE(String tITLE) {
        TITLE = tITLE;
    }
    public String getCONTENT() {
        return CONTENT;
    }
    public void setCONTENT(String cONTENT) {
        CONTENT = cONTENT;
    }
    public Date getREGDATE() {
        return REGDATE;
    }
    public void setREGDATE(Date rEGDATE) {
        REGDATE = rEGDATE;
    }
    @Override
    public String toString() {
        return "table [SEQ=" + SEQ + ", WRITER=" + WRITER + ", TITLE=" + TITLE + ", CONTENT=" +
CONTENT + ", REGDATE="
                + REGDATE + "]";
    }
}
```

답안첨삭

```
package com.kh.test.model;

import java.sql.Date;

public class table {
    private  int SEQ;
    private  String WRITER;
    private  String  TITLE;
    private String CONTENT;
    private Date  REGDATE;
    public table() {
        super();
        // TODO Auto-generated constructor stub
    }
    public table(int sEQ, String wRITER, String tITLE, String cONTENT, Date rEGDATE) {
        super();
        SEQ = sEQ;
        WRITER = wRITER;
        TITLE = tITLE;
        CONTENT = cONTENT;
        REGDATE = rEGDATE;
    }
    public int getSEQ() {
        return SEQ;
    }
    public void setSEQ(int sEQ) {
        SEQ = sEQ;
    }
    public String getWRITER() {
        return WRITER;
    }
    public void setWRITER(String wRITER) {
        WRITER = wRITER;
    }
    public String getTITLE() {
        return TITLE;
    }
    public void setTITLE(String tITLE) {
        TITLE = tITLE;
    }
    public String getCONTENT() {
        return CONTENT;
    }
    public void setCONTENT(String cONTENT) {
        CONTENT = cONTENT;
    }
    public Date getREGDATE() {
        return REGDATE;
    }
    public void setREGDATE(Date rEGDATE) {
        REGDATE = rEGDATE;
    }
    @Override
    public String toString() {
        return "table [SEQ=" + SEQ + ", WRITER=" + WRITER + ", TITLE=" + TITLE + ", CONTENT=" +
CONTENT + ", REGDATE="
                + REGDATE + "]";
    }
}
```

=> DB는 대소문자 구분을 안해서 관례적으로 대문자로 표현하고, Java App은 CamelCasing을 적용하세요 ;)

모범답안

```
package com.kh.test.model;
import java.util.Date;
public class Test {
private int seq;
private String writer;
private String title;
private String content;
private Date regdate;
public Test() {
}
public Test(int seq, String writer, String title, String content, Date regdate) {
this.seq = seq;
this.writer = writer;
this.title = title;
this.content = content;
this.regdate = regdate;
}
public int getSeq() {
return seq;
}
public void setSeq(int seq) {
this.seq = seq;
}
public String getWriter() {
return writer;
}
public void setWriter(String writer) {
this.writer = writer;
}
public String getTitle() {
return title;
}
public void setTitle(String title) {
this.title = title;
}
public String getContent() {
return content;
}
public void setContent(String content) {
this.content = content;
}
public Date getRegdate() {
return regdate;
}
public void setRegdate(Date regdate) {
this.regdate = regdate;
}
}
```

TEST table과 연결하기 위한 클래스를 작성하고, TEST table의 모든 rows를 리턴하는 메서드를 작성하시오.

요구사항

클래스 : com.kh.test.model.TestDao
메서드 : public List<Test> selectList()
DB url : jdbc:oracle:thin:@192.168.10.3:1521:xe
DB username : kh
DB password : kh

수강생 답

```
public List<Test> selectList() {
      Connection conn = null;
      PreparedStatement pstmt = null;
      ResultSet rset = null;
      String sql = "select * from test";
      List<Test> list = new ArrayList<>();

      try {
         Class.forName(driverClass);
         conn = DriverManager.getConnection('jdbc:oracle:thin:@192.168.10.3:1521:xe',
                              'kh', 'kh');
         pstmt = conn.prepareStatement(sql);
         rset = pstmt.executeQuery();
         while(rset.next()) {
            int seq = rset.getInt("seq");
            String writer = rset.getString("writer");
            String title = rset.getString("title");
            String content = rset.getString("title");
            Date regdate = rset.getDate("regdate");

            Test t = new t(seq, writer, title, content, regdate);
            list.add(t);
         }
      } catch (ClassNotFoundException e) {
         e.printStackTrace();
      } catch (SQLException e) {
         e.printStackTrace();
      } finally {
         try {
            if(rset != null)
               rset.close();
         } catch (SQLException e) {
            e.printStackTrace();
         }
         try {
            if(pstmt != null)
               pstmt.close();
         } catch (SQLException e) {
            e.printStackTrace();
         }
         try {
            if(conn != null)
               conn.close();
         } catch (SQLException e) {
            e.printStackTrace();
         }
      }
      return list;
   }
}
```

답안첨삭

```
public List<Test> selectList() {
      Connection conn = null;
      PreparedStatement pstmt = null;
      ResultSet rset = null;
      String sql = "select * from test";
      List<Test> list = new ArrayList<>();

      try {
         Class.forName(driverClass);
         conn = DriverManager.getConnection('jdbc:oracle:thin:@192.168.10.3:1521:xe',
                              'kh', 'kh');
         pstmt = conn.prepareStatement(sql);
         rset = pstmt.executeQuery();
         while(rset.next()) {
            int seq = rset.getInt("seq");
            String writer = rset.getString("writer");
            String title = rset.getString("title");
            String content = rset.getString("title");
            Date regdate = rset.getDate("regdate");

            Test t = new t(seq, writer, title, content, regdate);
            list.add(t);
         }
      } catch (ClassNotFoundException e) {
         e.printStackTrace();
      } catch (SQLException e) {
         e.printStackTrace();
      } finally {
         try {
            if(rset != null)
               rset.close();
         } catch (SQLException e) {
            e.printStackTrace();
         }
         try {
            if(pstmt != null)
               pstmt.close();
         } catch (SQLException e) {
            e.printStackTrace();
         }
         try {
            if(conn != null)
               conn.close();
         } catch (SQLException e) {
            e.printStackTrace();
         }
      }
      return list;
   }
}
```

요구사항에 맞는 DAO클래스와 method를 명확하게 작성함

모범답안

```
package com.kh.test.model;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
public class TestDao {
public List<Test> selectList(){
try {
Class.forName("oracle.jdbc.driver.OracleDriver");
} catch (ClassNotFoundException e) {
e.printStackTrace();
}
Connection con = null;
String url = "jdbc:oracle:thin:@localhost:1521:xe";
String user = "kh";
String password = "kh";
try {
con = DriverManager.getConnection(url, user, password);
con.setAutoCommit(false);
} catch (SQLException e) {
e.printStackTrace();
}
PreparedStatement pstm = null;
ResultSet rs = null;
List<Test> list = new ArrayList<Test>();
String sql = " SELECT * FROM TEST ";
try {
pstm = con.prepareStatement(sql);
rs = pstm.executeQuery();
while(rs.next()) {
Test test = new Test();
test.setSeq(rs.getInt(1));
test.setWriter(rs.getString(2));
test.setTitle(rs.getString(3));
test.setContent(rs.getString(4));
test.setRegdate(rs.getDate(5));
list.add(test);
}
} catch (SQLException e) {
e.printStackTrace();
} finally {
try {
rs.close();
pstm.close();
con.close();
} catch (SQLException e) {
e.printStackTrace();
}
}
return list;
}
}
```

평가항목

평가내용	평가기준	배점	평가결과
리소스 관리	웹 애플리케이션 서버의 Servlet 라이프 사이클을 정확하게 이해하고 있다.	10.0	예
명령어 제어	CLI(Command Line Interface) 및 GUI(Graphic User Interface) 환경에서 운영체제의 기본명령어를 활용할 수 있다.	10.0	예
입출력 클래스	데이터베이스의 종류를 구분하고 응용 소프트웨어 개발에 필요한 데이터베이스를 선정할 수 있다.	10.0	예
컬렉션 클래스	주어진 E-R 다이어그램을 이용하여 관계형 데이터베이스의 테이블을 정의할 수 있다.	10.0	예
데이터 흐름	데이터베이스의 기본연산을 CRUD(Create, Read, Update, Delete)로 구분하여 설명할 수 있다	10.0	예
서버와 클라이언트	뷰와 컨트롤러 간의 연결 처리가 잘 구동되게 처리할 수 있다.	10.0	예
서버 Request	서버 요청 객체에 대해 정확히 이해하고 사용할 수 있다.	10.0	예
클라이언트 Response	클라이언트 응답 객체에 대해 정확히 이해하고 사용할 수 있다.	10.0	예
응용프로그램 개발	응용프로그램 개발을 위한 클래스들을 적절히 적용해 개발할 수 있다.	10.0	예
서버-클라이언트 구현	서버-클라이언트 간의 데이터 전송 관련 클래스에 대해 적절히 선택 사용할 수 있다.	10.0	아니오

평가 메인으로