Day34; 20220817

강의 번호	Lesson 34
유형	@2022년 8월 17일
자료	
속성	
복습	☐
작성일자	@2022년 8월 17일 오전 9:39

Connection Pool - Basic

JDBC를 사용할 때 가장 많은 리소스가 소모되는 부분이 데이터베이스 연동에 필요한 Connection 객체를 생성하는 부분이다. 지금까지 JSP에서 SQL구문을 수행하기 위해서 Connection 객체를 생성하고 사용 후 제거하는 과정을 반복해 왔다. 이것은 접속자가 많아질 경우 시스템의 성능을 급격하게 저하시킬 수 있다.

따라서 이러한 문제를 해결하기 위한 방법으로 데이터베이스 커넥션 풀 Database Connection Pool을 이용하는 방법이 있다. 사용자가 접속할 때마다 매번 새로운 Connection 객체를 생성하는 것이 아니라 일정 개수의 Connection 객체를 미리 생성해 놓고 사용자의 요청이 있을 때마다 가용한 객체를 할당하고 다시 회수하는 방식이다.

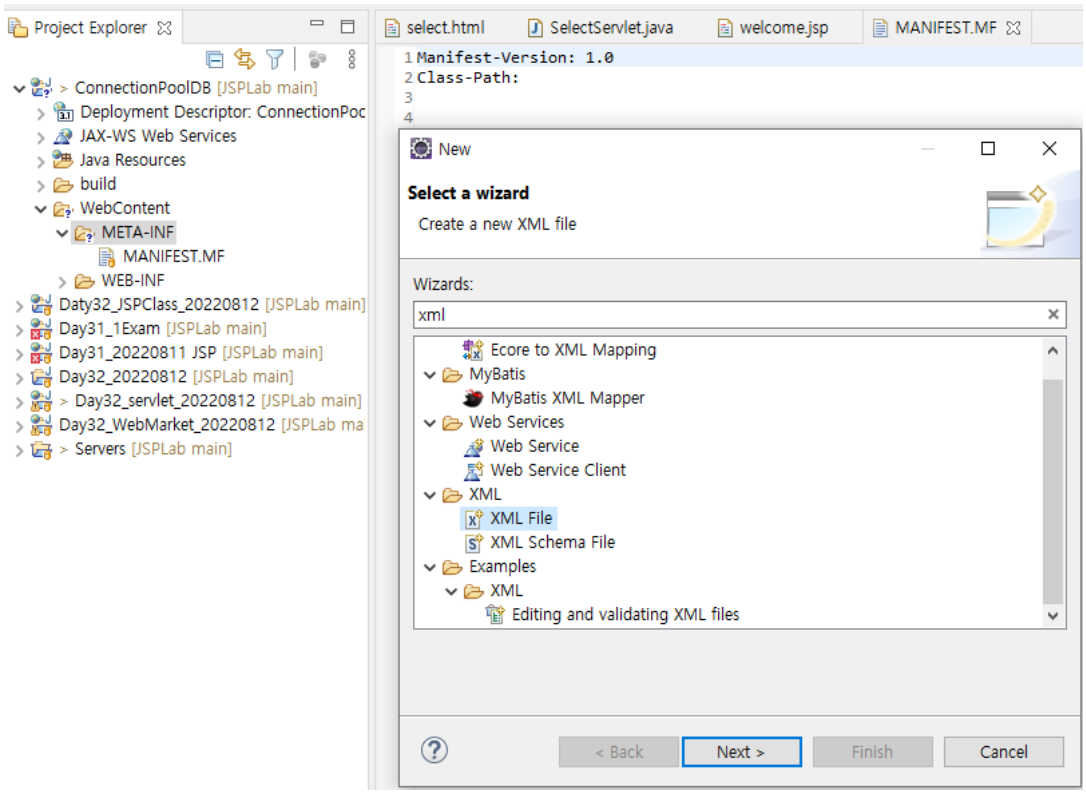
1. context.xml

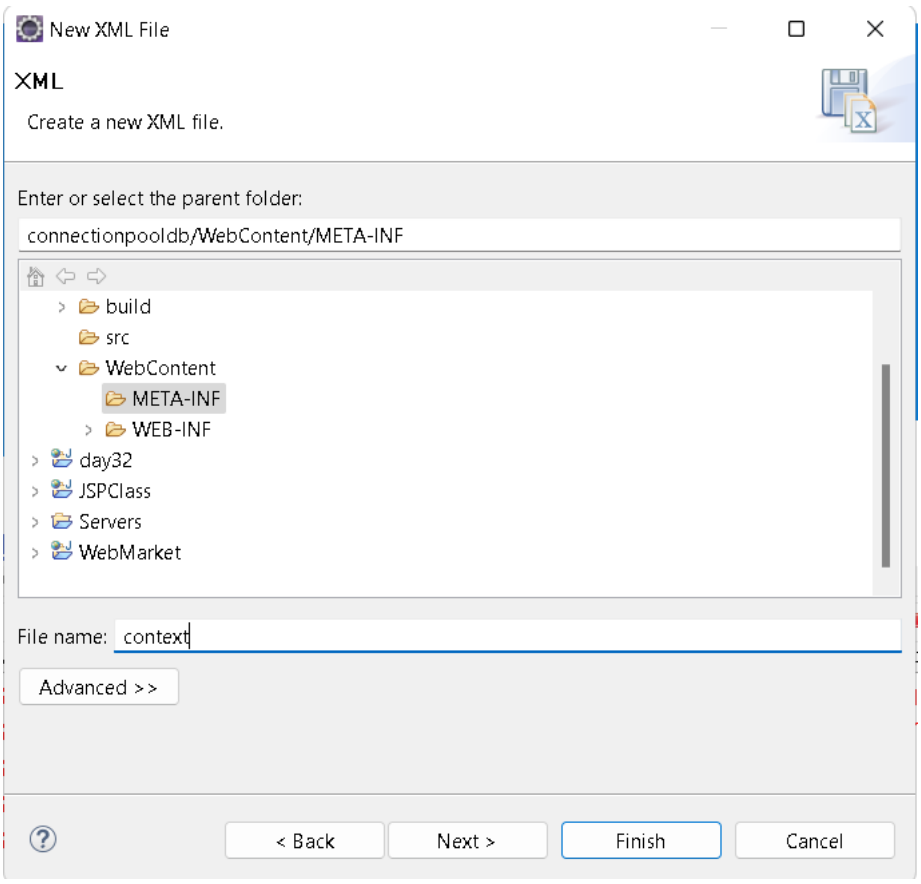
   데이터베이스에 대한 커넥션 풀을 생성하기 위한 설정을 정의한다.

   위치는 WebContent > META-INF > context.xml

   설정에는 연결할 데이터베이스 이름과 데이터베이스 접근 아이디와 패스워드, 최대 커넥션 갯수, 최소유지 커넥션 갯수 등등을 정의한다.

[xml 만들기]

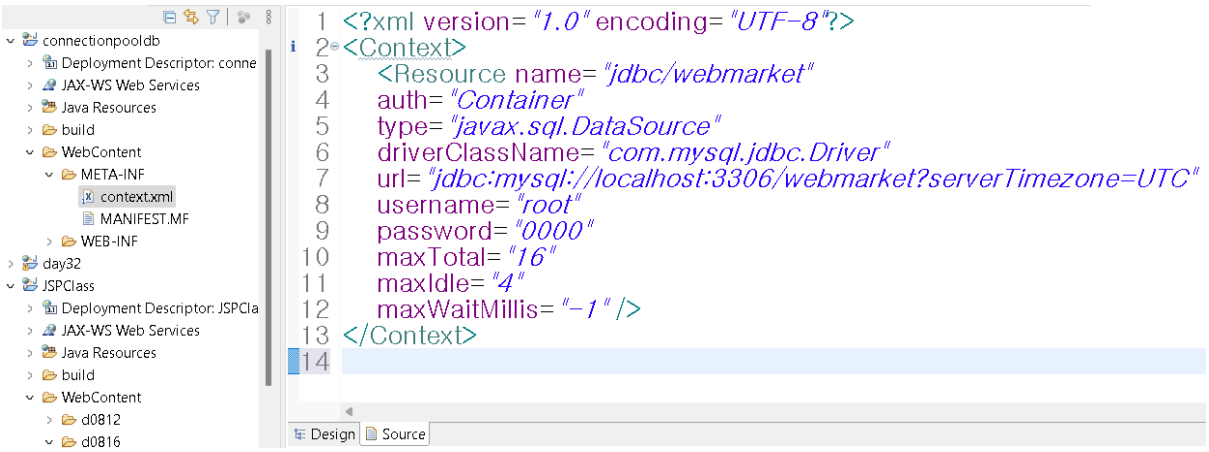




Source 탭

```
<?xml version="1.0" encoding="UTF-8"?>
<Context>
```

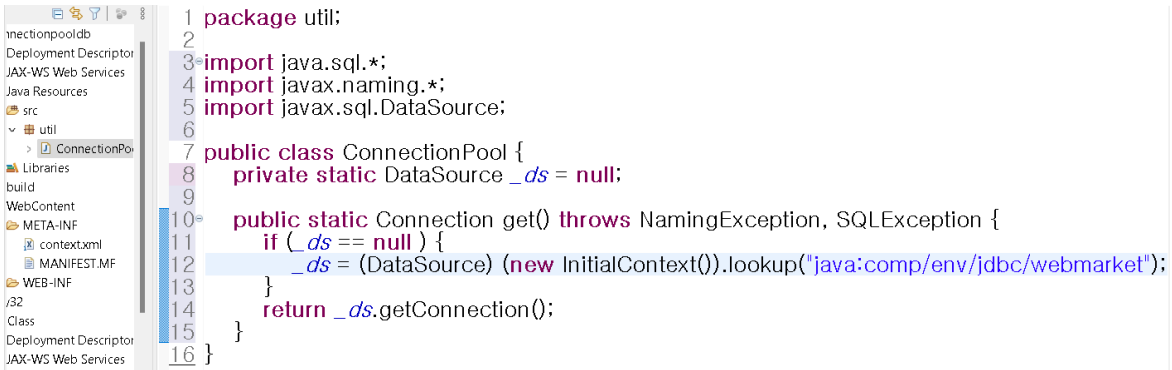
```
  <Resource name = "jdbc/webmarket"
  auth = "Container"
  type = "javax.sql.DataSource"
  driverClassName = "com.mysql.jdbc.Driver"
  url = "jdbc:mysql://localhost:3307/webmarket?serverTimezone=UTC"
  username = "root"
  password = "0000"
  maxTotal = "16"
  maxIdle = "4"
  maxWaitMillis = "-1" />
</Context>
```



2. ConnectionPool.java

   context.xml에 정의된 내용으로 실제 데이터베이스에 연결과 객체 생성 등의 직접적인 실행이 이루어지는 자바 클래스를 만든다.

   위치는 src > util > ConnectionPool.java



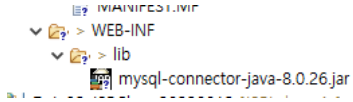

```
package util;

import java.sql.*;
import javax.naming.*;
import javax.sql.DataSource;

public class ConnectionPool {
  private static DataSource _ds = null;

  public static Connection get() throws NamingException, SQLException {
    if (_ds == null ) {
      _ds = (DataSource) (new InitialContext()).lookup("java:comp/env/jdbc/webmarket");
    }
```

```
    return _ds.getConnection();
  }
}
```

3. mysql-connector-java-8.0.26.jar

   JDBC 연결 드라이버 붙여넣기

   위치는 WEB-INF > lib > mysql-connector-java-8.0.26.jar

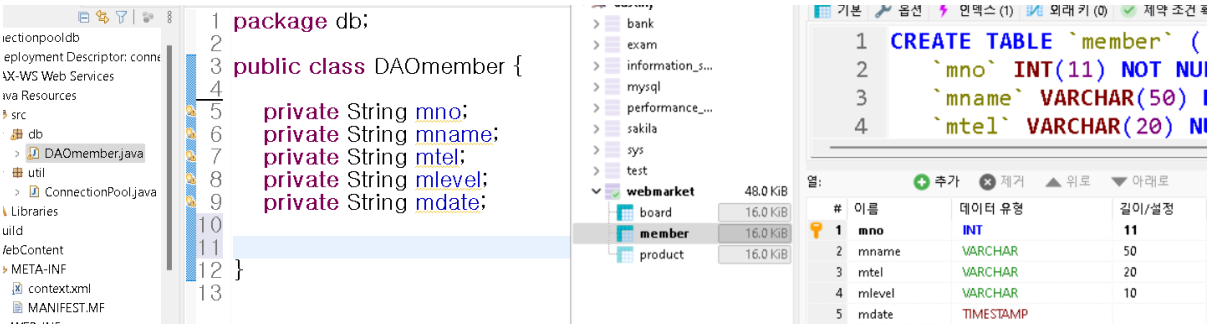


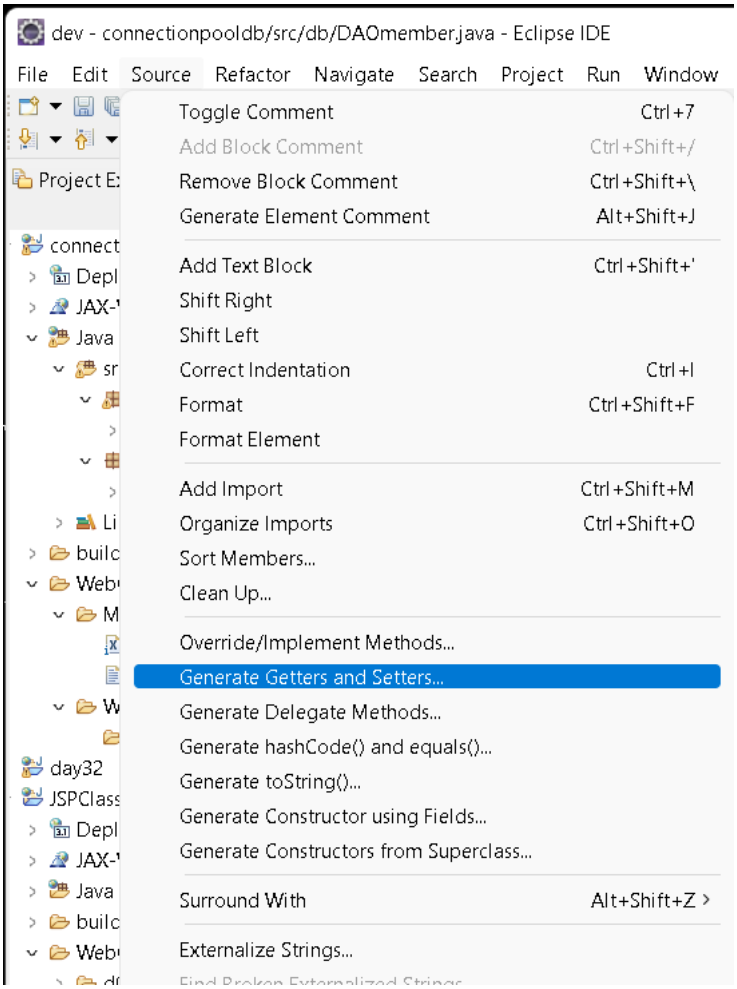
4. 테이블 클래스 만들기

   (DB에 테이블 생성 방법은 생략)

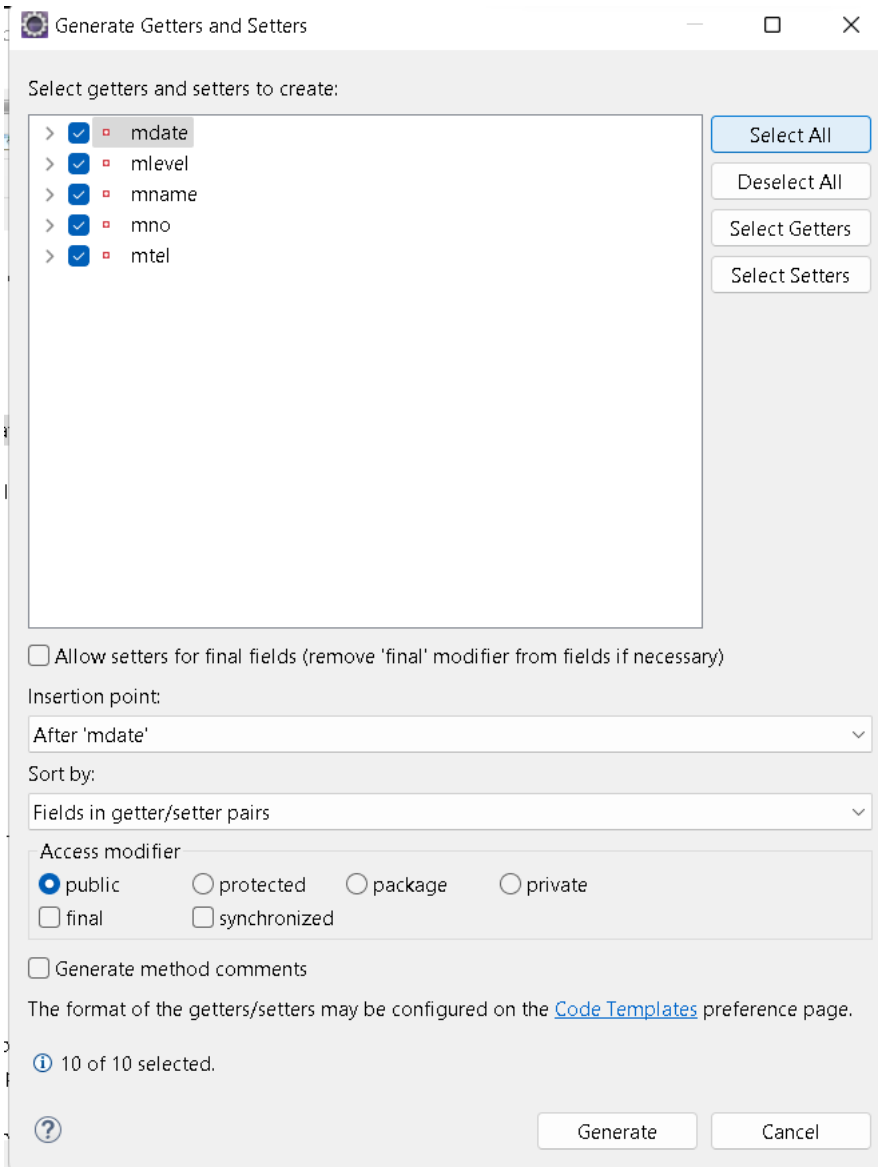
   만들어 놓은 테이블의 필드값들과 1:1 매칭이 되는 클래스를 작성한다.

   클래스명에는 일반적으로 DTO (Data Tranfer Object)

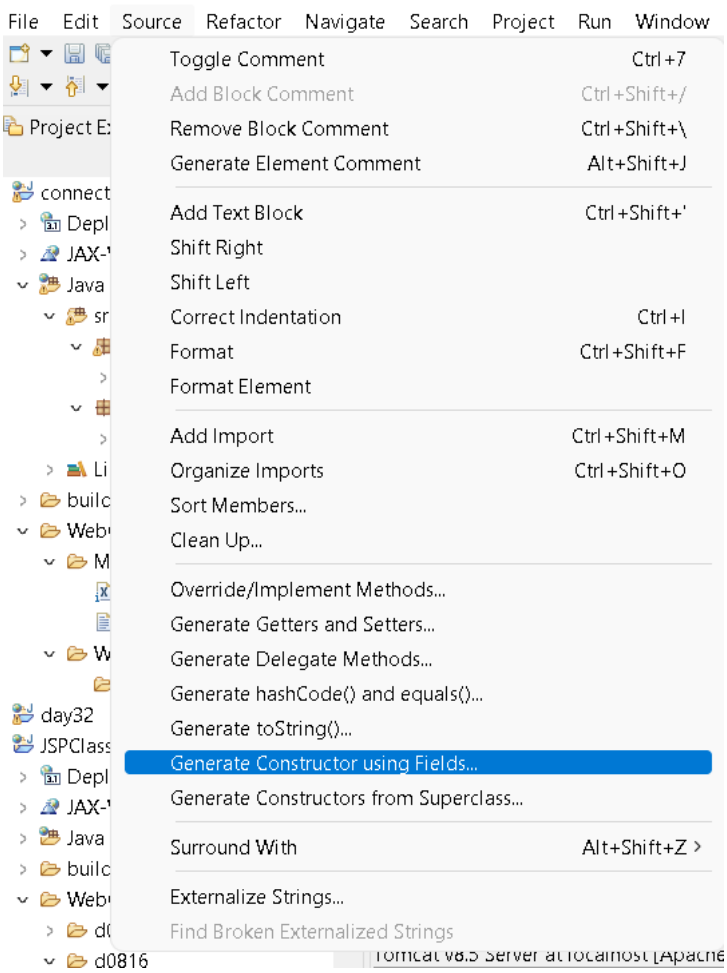


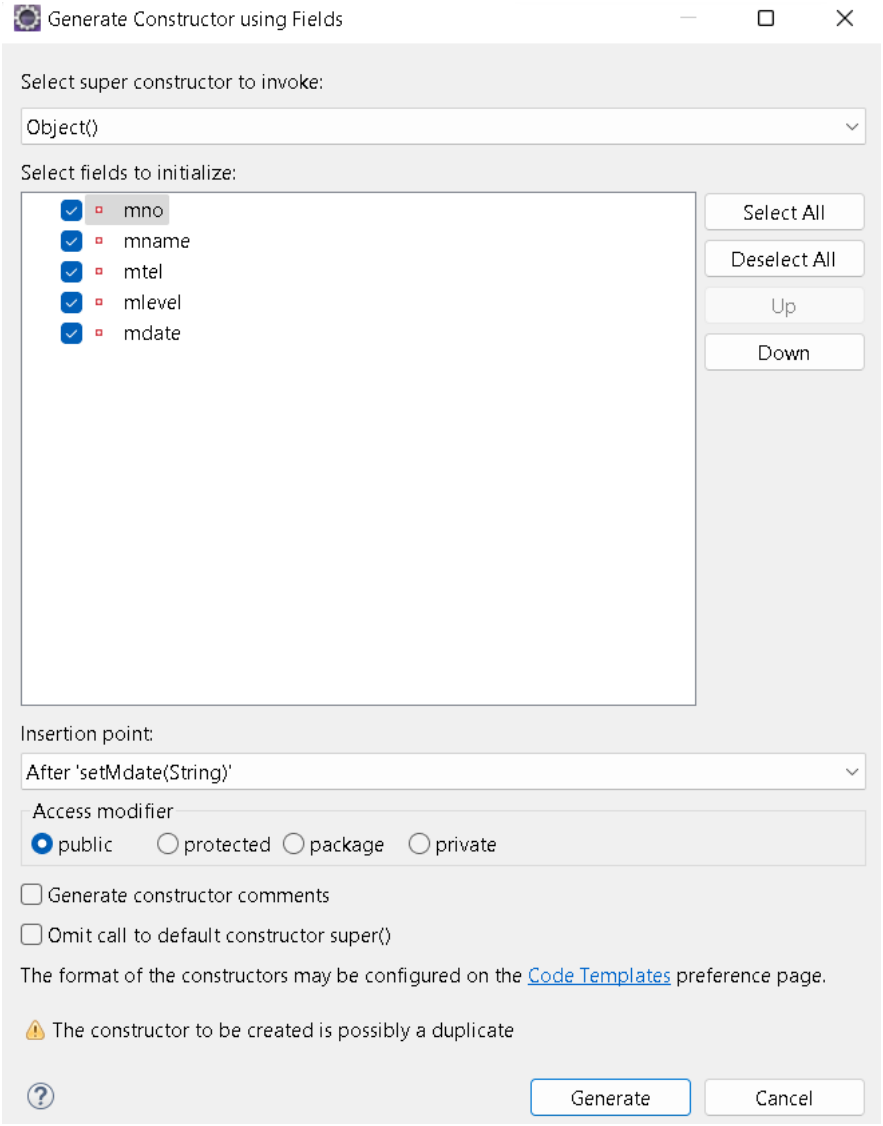


Getter와 Setter 자동 생성


Generate Getters and Setters
Select getters and setters to create:
mdate
mlevel
mname
mno
mtel
Select All
Deselect All
Select Getters
Select Setters
Allow setters for final fields (remove 'final' modifier from fields if necessary)
Insertion point:
After 'mdate'
Sort by:
Fields in getter/setter pairs
Access modifier
public
protected
package
private
final
synchronized
Generate method comments
The format of the getters/setters may be configured on the Code Templates preference page.
10 of 10 selected.
Generate
Cancel

생성자 자동생성


File Edit Source Refactor Navigate Search Project Run Window
Toggle Comment Ctrl+7
Add Block Comment Ctrl+Shift+/
Remove Block Comment Ctrl+Shift+\
Generate Element Comment Alt+Shift+J
Add Text Block Ctrl+Shift+'
Shift Right
Shift Left
Correct Indentation Ctrl+I
Format Ctrl+Shift+F
Format Element
Add Import Ctrl+Shift+M
Organize Imports Ctrl+Shift+O
Sort Members...
Clean Up...
Override/Implement Methods...
Generate Getters and Setters...
Generate Delegate Methods...
Generate hashCode() and equals()...
Generate toString()...
Generate Constructor using Fields...
Generate Constructors from Superclass...
Surround With Alt+Shift+Z
Externalize Strings...
Find Broken Externalized Strings

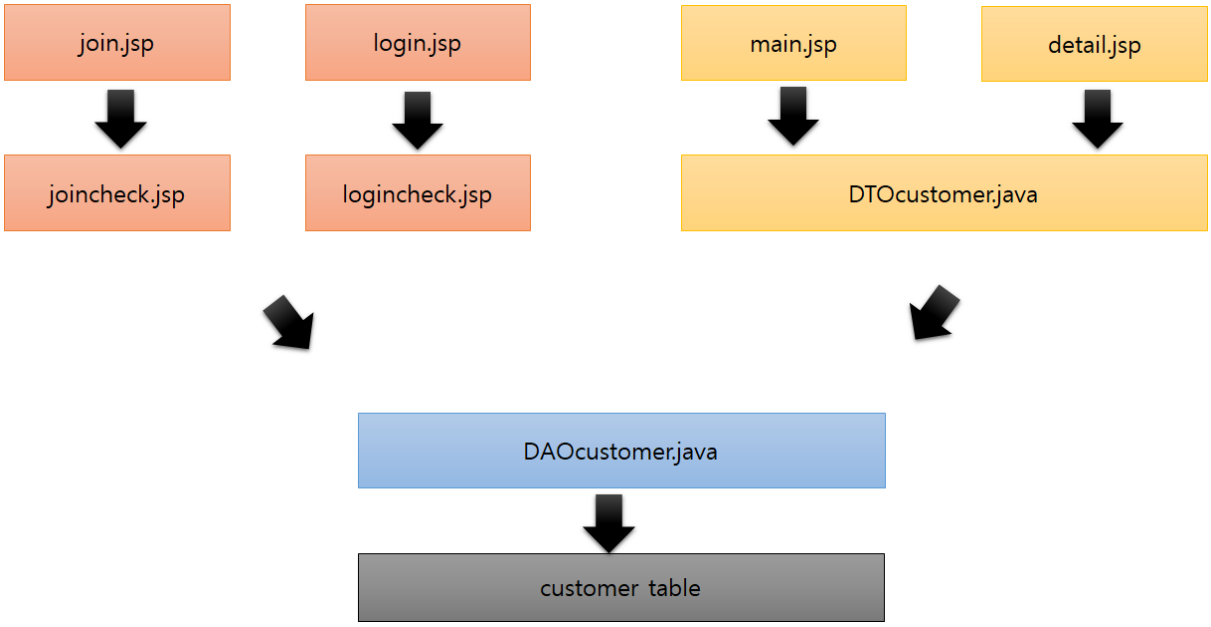


5. 데이터 접근 클래스 작성

   클래스명에는 일반적으로 DAO (Data Access Object)

```
package db;

import java.sql.*;

import javax.naming.NamingException;

import util.*;

public class DAOmember {

  // C create R read U update D delete
  //회원 가입 (테이블에 데이터 넣기)
  public static int join(String mname, String mtel) throws NamingException, SQLException {

    Connection conn = null;
    PreparedStatement stmt = null;

    String sql = "INSERT INTO member (mname, mtel) VALUES(?,?)";
    //Connection Pool 이용
    conn= ConnectionPool.get();
```

```
    stmt = conn.prepareStatement(sql);
      stmt.setString(1, mname);
      stmt.setString(2, mtel);

    int result = stmt.executeUpdate();
    // 결과가 1 과 2로 넘어 온다.

    return result;
  }

}
```



로그인 처리 - 3개 login.jsp, logincheck.jsp, login() method

login.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>



<!-- CSS only -->
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.0/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-gH2yIJqKdNHPEq


    <!-- Favicons -->
<link rel="apple-touch-icon" href="/docs/5.2/assets/img/favicons/apple-touch-icon.png" sizes="180x180">
<link rel="icon" href="/docs/5.2/assets/img/favicons/favicon-32x32.png" sizes="32x32" type="image/png">
<link rel="icon" href="/docs/5.2/assets/img/favicons/favicon-16x16.png" sizes="16x16" type="image/png">
<link rel="manifest" href="/docs/5.2/assets/img/favicons/manifest.json">
<link rel="mask-icon" href="/docs/5.2/assets/img/favicons/safari-pinned-tab.svg" color="#712cf9">
<link rel="icon" href="/docs/5.2/assets/img/favicons/favicon.ico">
<meta name="theme-color" content="#712cf9">


    <style>
      .bd-placeholder-img {
        font-size: 1.125rem;
        text-anchor: middle;
        -webkit-user-select: none;
        -moz-user-select: none;
        user-select: none;
      }

      @media (min-width: 768px) {
        .bd-placeholder-img-lg {
          font-size: 3.5rem;
```

```
        }
      }

      .b-example-divider {
        height: 3rem;
        background-color: rgba(0, 0, 0, .1);
        border: solid rgba(0, 0, 0, .15);
        border-width: 1px 0;
        box-shadow: inset 0 .5em 1.5em rgba(0, 0, 0, .1), inset 0 .125em .5em rgba(0, 0, 0, .15);
      }

      .b-example-vr {
        flex-shrink: 0;
        width: 1.5rem;
        height: 100vh;
      }

      .bi {
        vertical-align: -.125em;
        fill: currentColor;
      }

      .nav-scroller {
        position: relative;
        z-index: 2;
        height: 2.75rem;
        overflow-y: hidden;
      }

      .nav-scroller .nav {
        display: flex;
        flex-wrap: nowrap;
        padding-bottom: 1rem;
        margin-top: -1px;
        overflow-x: auto;
        text-align: center;
        white-space: nowrap;
        -webkit-overflow-scrolling: touch;
      }
    </style>


    <!-- Custom styles for this template -->
    <link href="css/signin.css" rel="stylesheet">

  </head>
  <body class="text-center">

<main class="form-signin w-100 m-auto">


  <form action="logincheck.jsp">
    <img class="mb-4" src="http://austiny.snu.ac.kr/image/mine.jpg" alt="" width="72" height="57">
    <h1 class="h3 mb-3 fw-normal">로그인 하세요</h1>

    <div class="form-floating">
      <input type="text" name="id" class="form-control" id="floatingInput" placeholder="id">
      <label for="floatingInput">아이디</label>
    </div>
    <div class="form-floating">
      <input type="password" name="pass"  class="form-control" id="floatingPassword" placeholder="Password">
      <label for="floatingPassword">패스워드</label>
    </div>

    <div class="checkbox mb-3">
      <label>
        <input type="checkbox" value="remember-me"> 로그인 기억
      </label>
    </div>
    <button class="w-100 btn btn-lg btn-primary" type="submit">로그인</button>
    <p class="mt-5 mb-3 text-muted">&copy; 2017–2022</p>
  </form>
</main>



  </body>
</html>
```

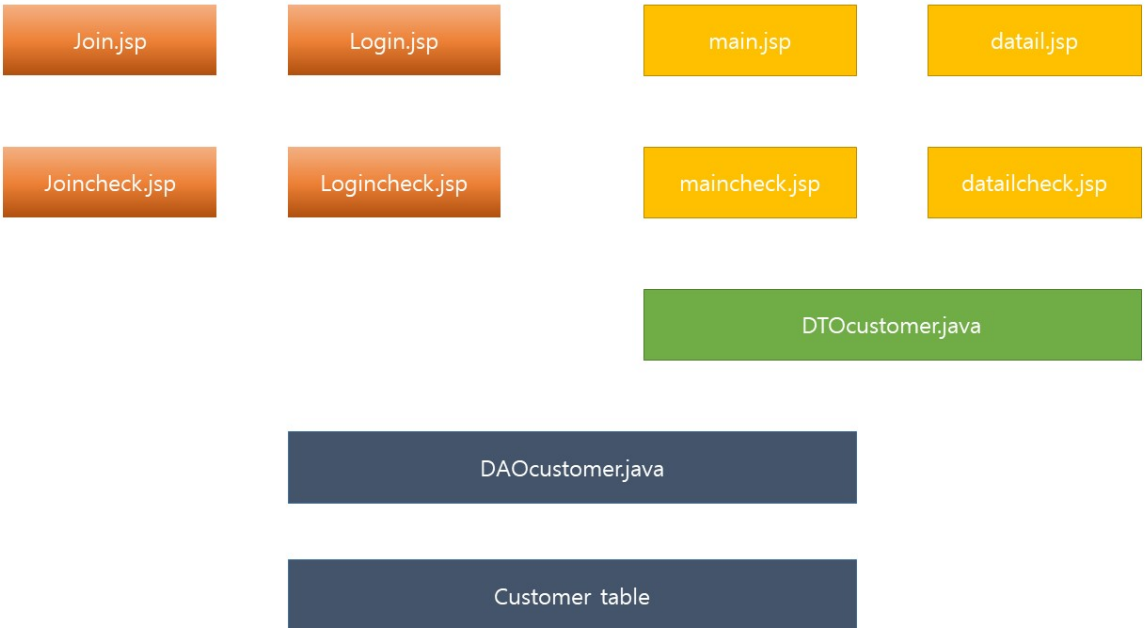
logincheck.jsp

```
<%@page import="db.*"%>
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
```

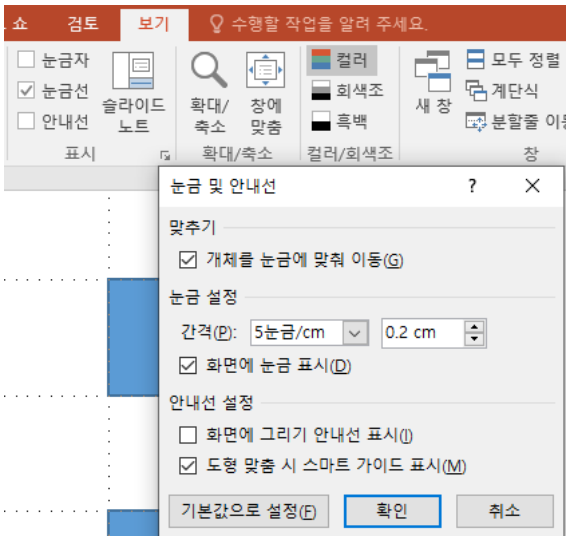
```
<%

  String id = request.getParameter("id");
  String pass = request.getParameter("pass");

  int result = DAOcustomer.login(id, pass);

  //로그인시 3가지 경우의 수가 발생된다.
  //1로그인 성공(아이디/비번 일치) -> 로그인 승인 메인으로 보내기
  //2로그인 실패(아이디는일치/비번 불일치) -> 다시 로그인으로 보내기
  //3로그인 실패(아이디가 없는 경우) -> 회원가입으로 보내기

  if (result == 1) {
    response.sendRedirect("main.jsp");
  }else if (result == 2) {
    response.sendRedirect("login.jsp");
  }else {
    response.sendRedirect("join.jsp");
  }

%>
```

class DAOcustomer > login() method

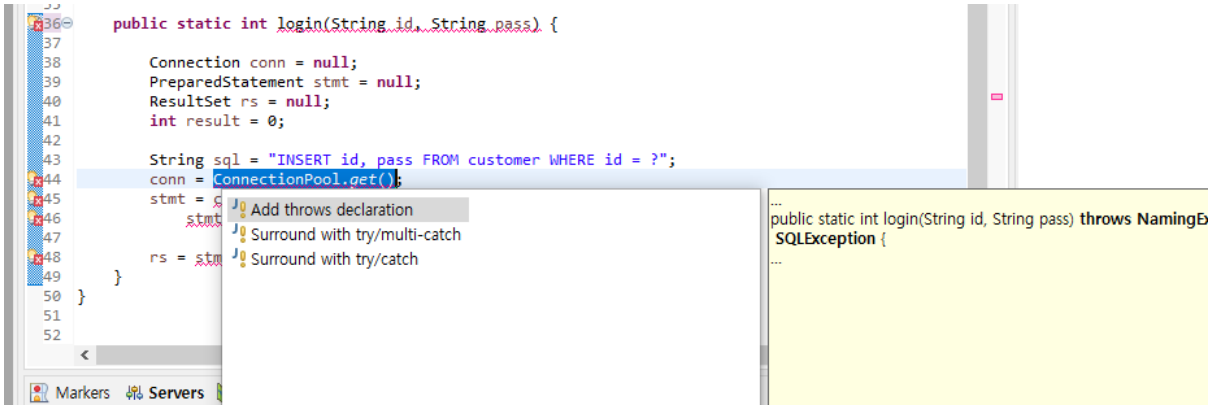
```
public static int login(String id, String pass) throws NamingException, SQLException {

    Connection conn = null;
    PreparedStatement stmt = null;
    ResultSet rs = null;

    int result = 0;

    String sql = "SELECT id, pass FROM customer WHERE id=?";
    conn = ConnectionPool.get();
    stmt = conn.prepareStatement(sql);
      stmt.setString(1, id);

    rs = stmt.executeQuery();

    if(!rs.next()) return 3;

    if(pass.equals(rs.getString("pass"))) return 1;

    return 2;

  }
```



[파워포인트를 draw.io처럼]



에러를 던짐 <에러 나면 홈런>



회원 전체 목록 보기

main.jsp

```
<%@page import="db.*"%>

<%@page import="java.util.ArrayList"%>
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
<h3>고객 목록</h3>

<%

  ArrayList<DTOcustomer> members = DAOcustomer.getList();

  for (DTOcustomer member : members) {

%>
  <%=member.getNo() %>
  <%=member.getId() %>
  <%=member.getName() %>
  <%=member.getPass() %>
  <%=member.getCdate() %>
  <br>
<%
  }
%>

</body>
</html>
```

getList() 매서드

```
public static ArrayList<DTOcustomer> getList() throws NamingException, SQLException {

    Connection conn = null;
    PreparedStatement stmt = null;
    ResultSet rs = null;

    String sql = "SELECT * FROM customer";

    conn = ConnectionPool.get();
    stmt = conn.prepareStatement(sql);
    rs = stmt.executeQuery();
```

```
    ArrayList<DTOcustomer> members = new ArrayList<DTOcustomer>();

    while(rs.next()) {
      members.add(new DTOcustomer(rs.getString(1),
                                  rs.getString(2),
                                  rs.getString(3),
                                  rs.getString(4),
                                  rs.getString(5)));
    }

    return members;
  }
```

한명 정보 상세 보기

detail.jsp

```
<%@page import="db.*"%>
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
<%
  String no = request.getParameter("no");

  DTOcustomer member = DAOcustomer.getDetail(no);

%>

  <%=member.getNo()%>
  <%=member.getId()%>
  <%=member.getName()%>
  <%=member.getPass()%>
  <%=member.getCdate()%>
</body>
</html>
```

class DAOcustomer > getDetail() 매서드

```
public static DTOcustomer getDetail(String no) throws NamingException, SQLException  {
    Connection conn = null;
    PreparedStatement stmt = null;
    ResultSet rs = null;

    String sql = "SELECT * FROM customer WHERE no=?";

    conn = ConnectionPool.get();
    stmt = conn.prepareStatement(sql);
      stmt.setNString(1, no);
    rs = stmt.executeQuery();

    rs.next();

    no = rs.getString(1);
    String id = rs.getString(2);
    String name = rs.getString(3);
    String pass = rs.getString(4);
    String cdate = rs.getString(5);

    DTOcustomer member = new DTOcustomer(no,id,name,pass,cdate);

    return member;

  }
```