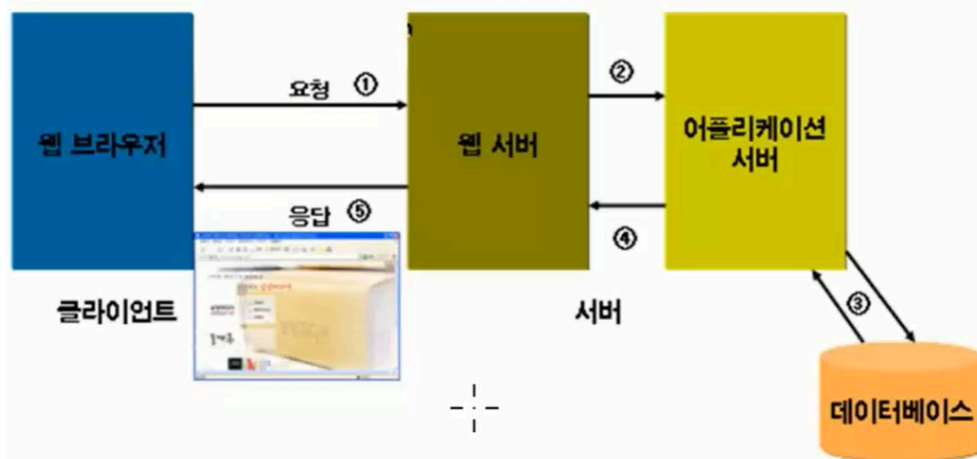


1월 10일 수요일

## 7장 웹 프로그래밍 개요

- 클라이언트 (다수) <-----network-----> 서버
  - 클라이언트 <form> 태그
  - 서버 (request, response) CRUD : insert, select, update, delete

### □ 웹 프로그래밍과 웹 어플리케이션(2/4)



- 웹서버와 데이터베이스

### □ 웹 프로그래밍과 웹 어플리케이션(4/4)

웹 어플리케이션의 구성요소	기능
웹 브라우저	웹에서 클라이언트이며, 사용자의 작업창이라 할 수 있다.
웹 서버	웹 브라우저의 요청을 받아들이는 곳으로 작업의 결과도 웹 브라우저에게 응답을 하는 곳이다. 요청된 페이지의 로직 및 데이터베이스와의 연동을 위해 어플리케이션 서버에 이들의 처리를 요청하는 작업을 수행한다.
웹 어플리케이션 서버(WAS)	요청된 페이지의 로직 및 데이터베이스와의 연동을 처리하는 부분이다.
데이터베이스	데이터의 저장소로 웹에서 발생한 데이터는 모두 이곳에 저장된다. 게시판의 글들, 회원의 정보 등등

## 1. JSP JAVA WEB 개발환경

### 2 JSP JAVA WEB 개발환경

- ㄱ. 자바 (CTRL + r, cmd 입력 엔터)  
자바가상머신 + 자바번역기(컴파일러)
- ㄴ. 자바코드작성 (메모장)  
이클립스 + 스프링 ==> STS  
다운로드 sts\_4.21 + 압축해제  
content.zip 파일 한번더 압축해제 => sts-4.21.0.RELEASE 폴더확인  
STS 작업워크폴더 d:\java\_web\_sol\workspace\_sts  
  
편집기의 한글세팅 (옵션: 한글 utf-8 ) window -> preference  
contenttype - java class  
editor - text editor - spelling  
workspace - 이미 설정됨
- ㄷ. 가상서버 (톰캣서버) (ㄴ번관련해서 : 다이내믹 웹프로젝트 생성)  
  
help - eclipseMarketPlace  
  
Eclipse Enterprise 검색 -> 라이선스 동의 -> 설치 진행 확인  
-> Trust Selected -> restart now
- ㄹ. 디비서버 (오라클) + SQL문작성 (디벨로버)

설치 - spring tool suite <https://spring.io/tools/>

## Spring Tools 4 for Eclipse

The all-new Spring Tool Suite 4. Free.  
Open source.

4.21.0 - LINUX X86\_64

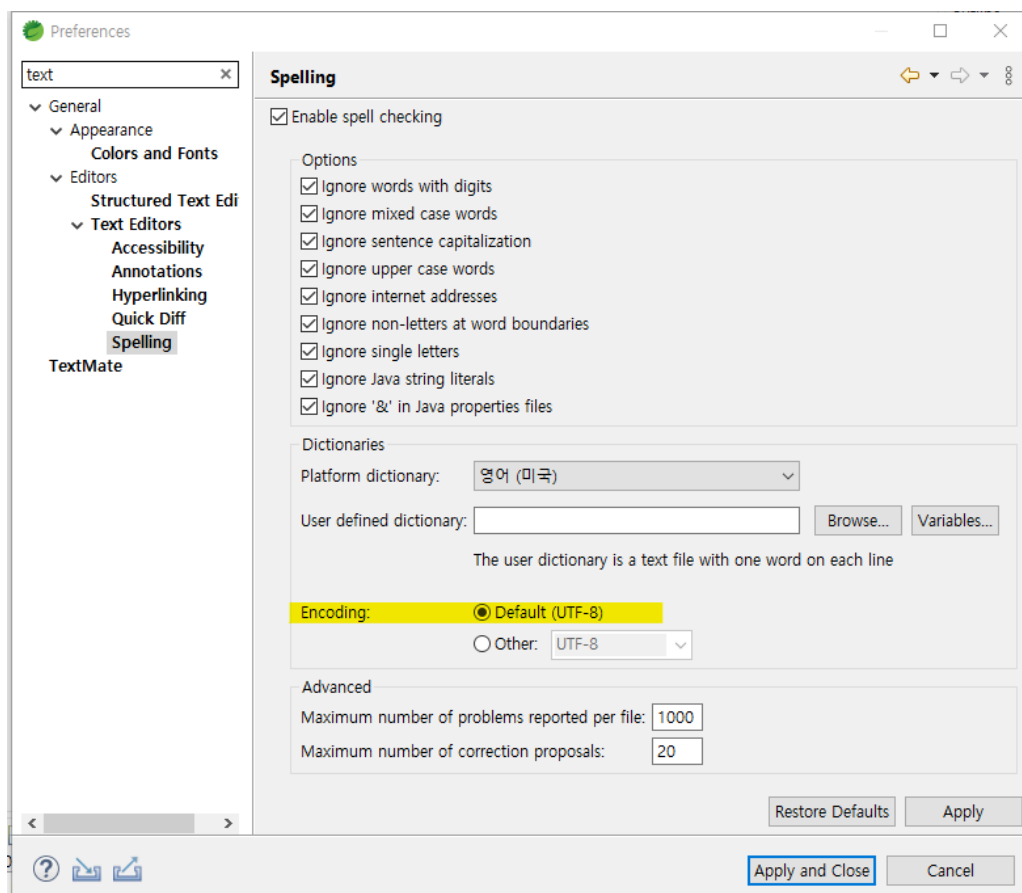
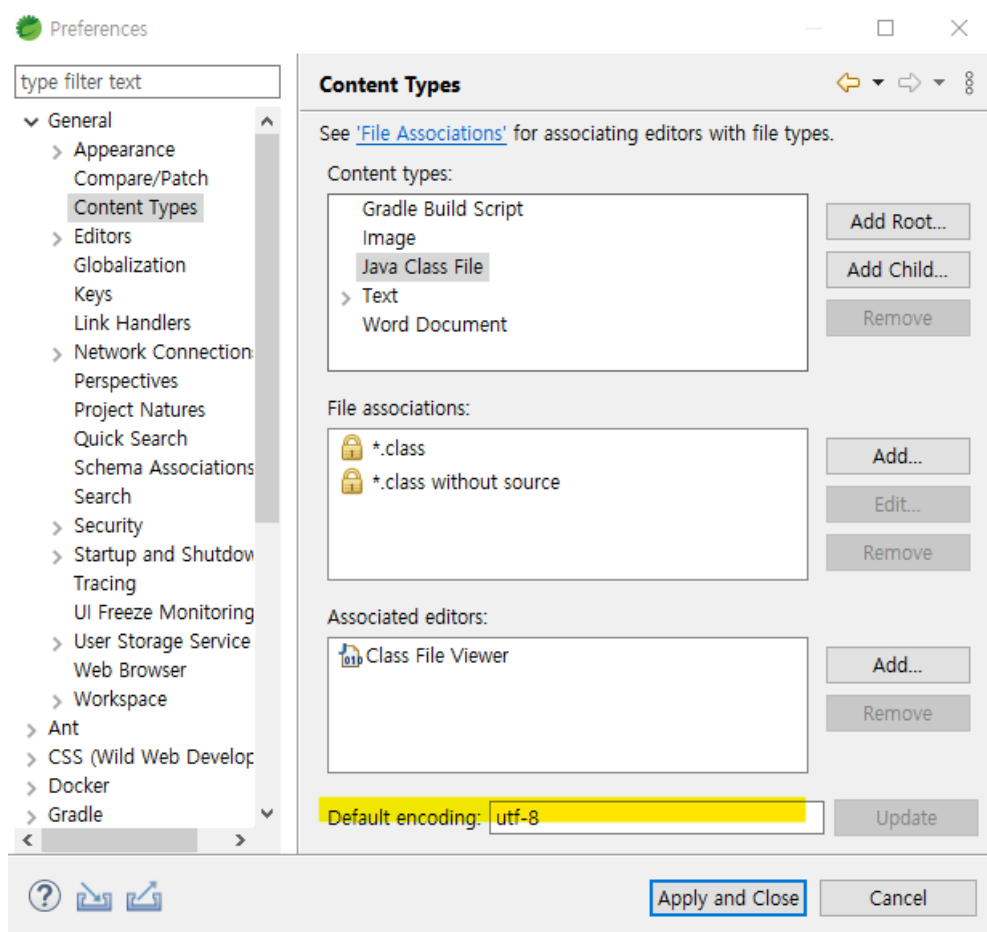
4.21.0 - LINUX ARM\_64

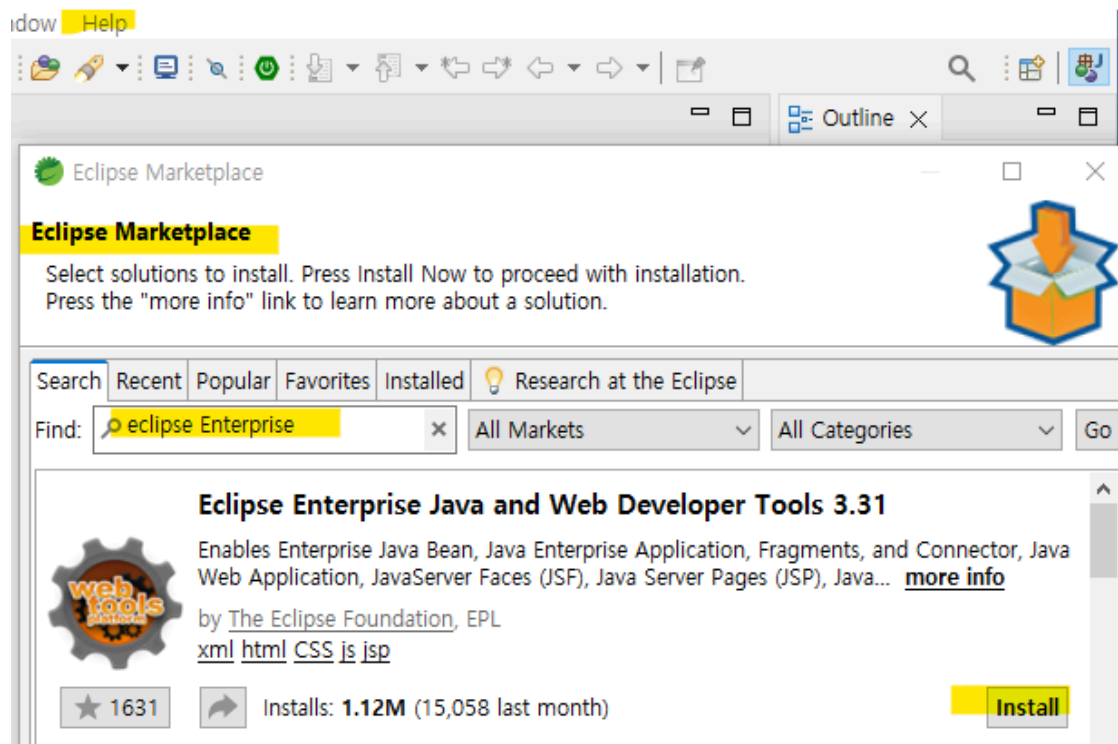
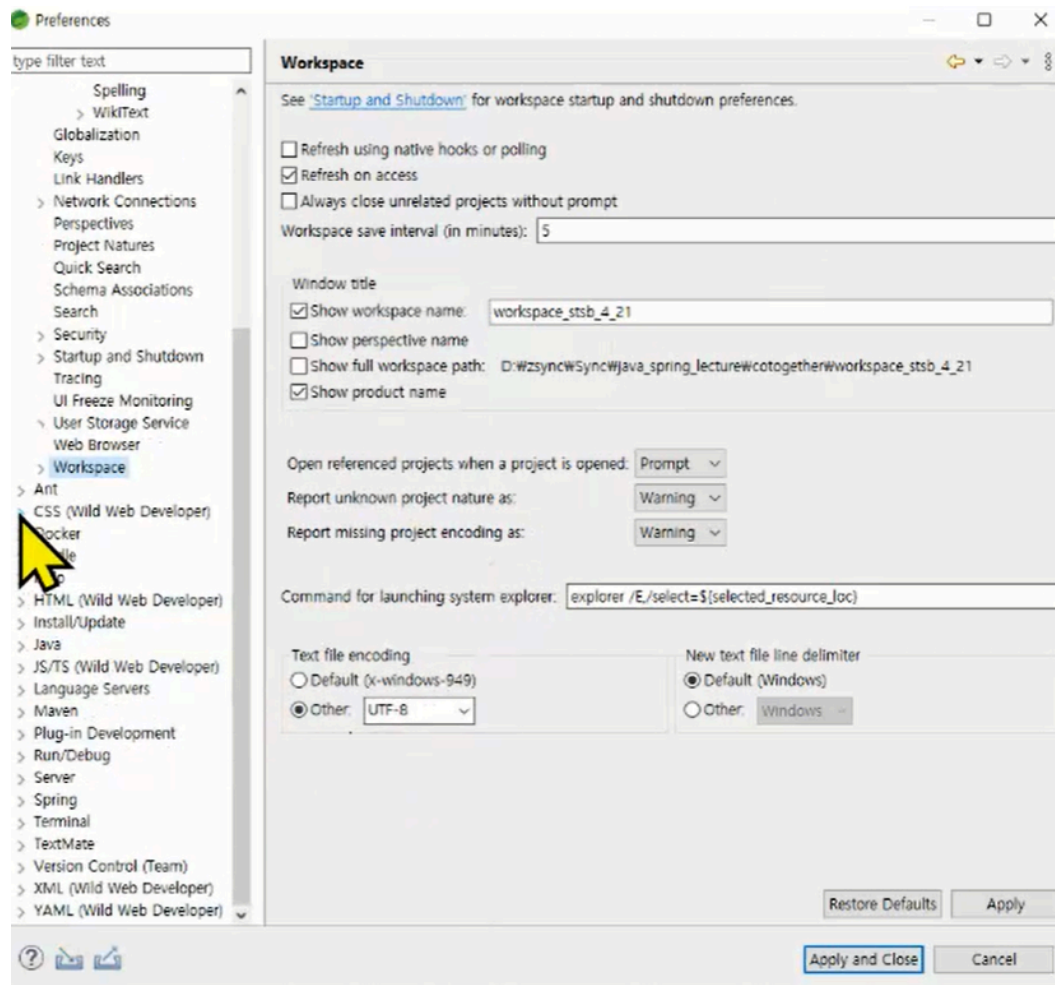
4.21.0 - MACOS X86\_64

4.21.0 - MACOS ARM\_64

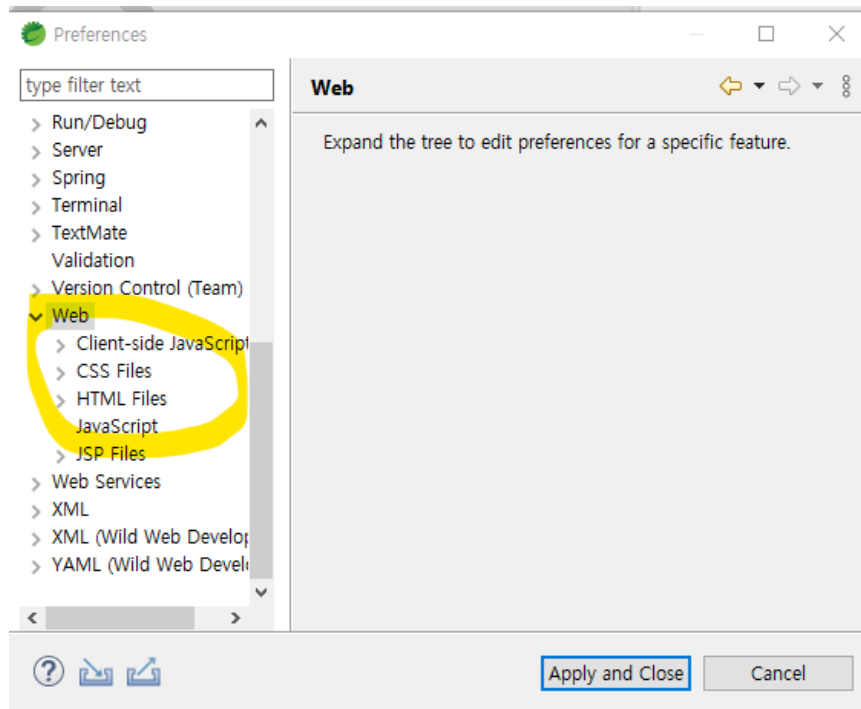
4.21.0 - WINDOWS X86\_64

## spring tool 한글세팅

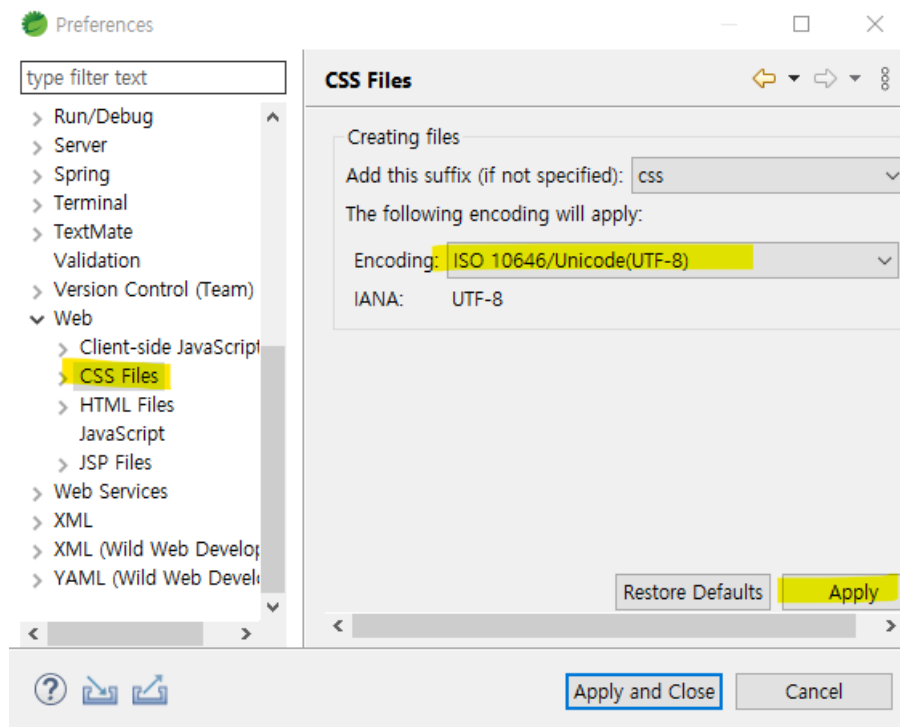


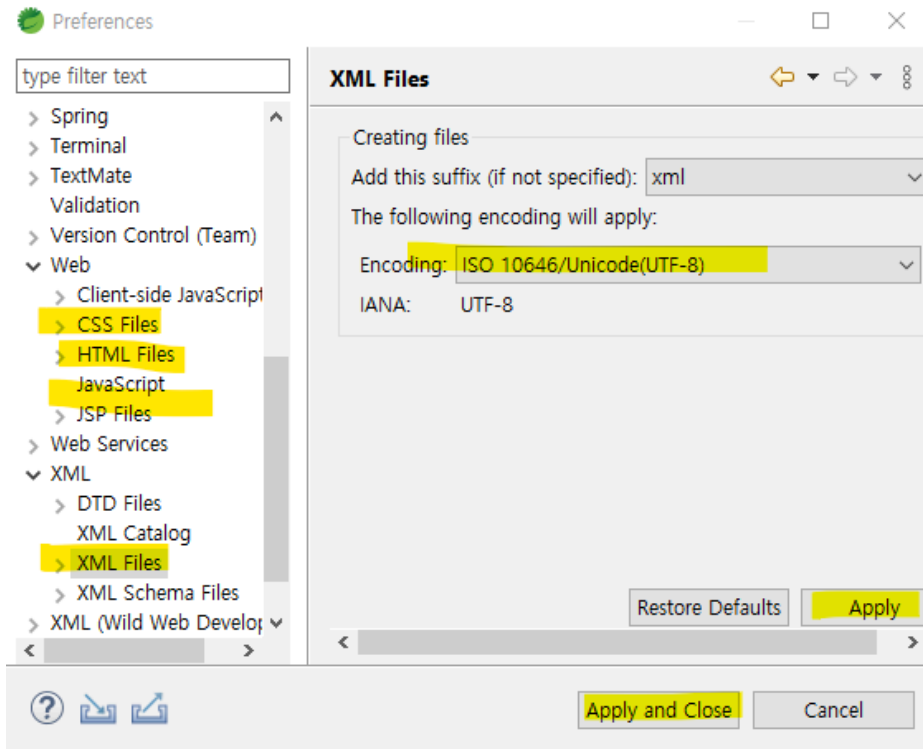


잘 설치되면 web 이 나온다



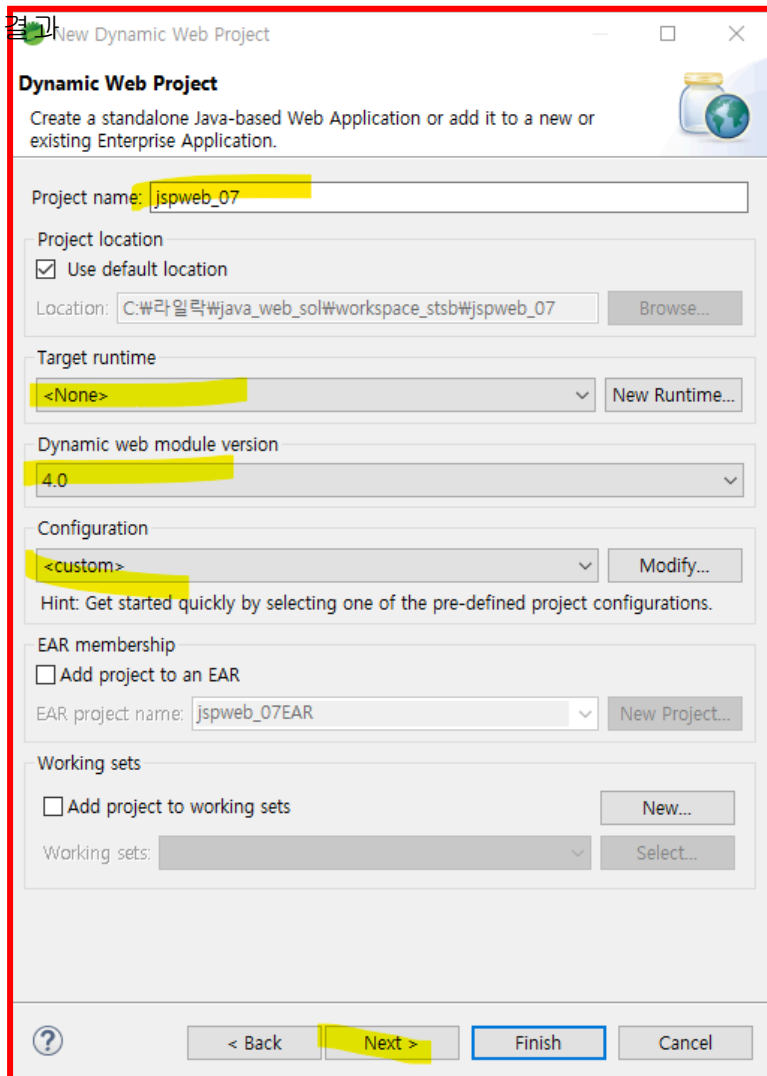
설치 후에 window - preference 에서 web - css, html, javascript, jsp, xml 모두 (옵션: 한글 utf-8 )



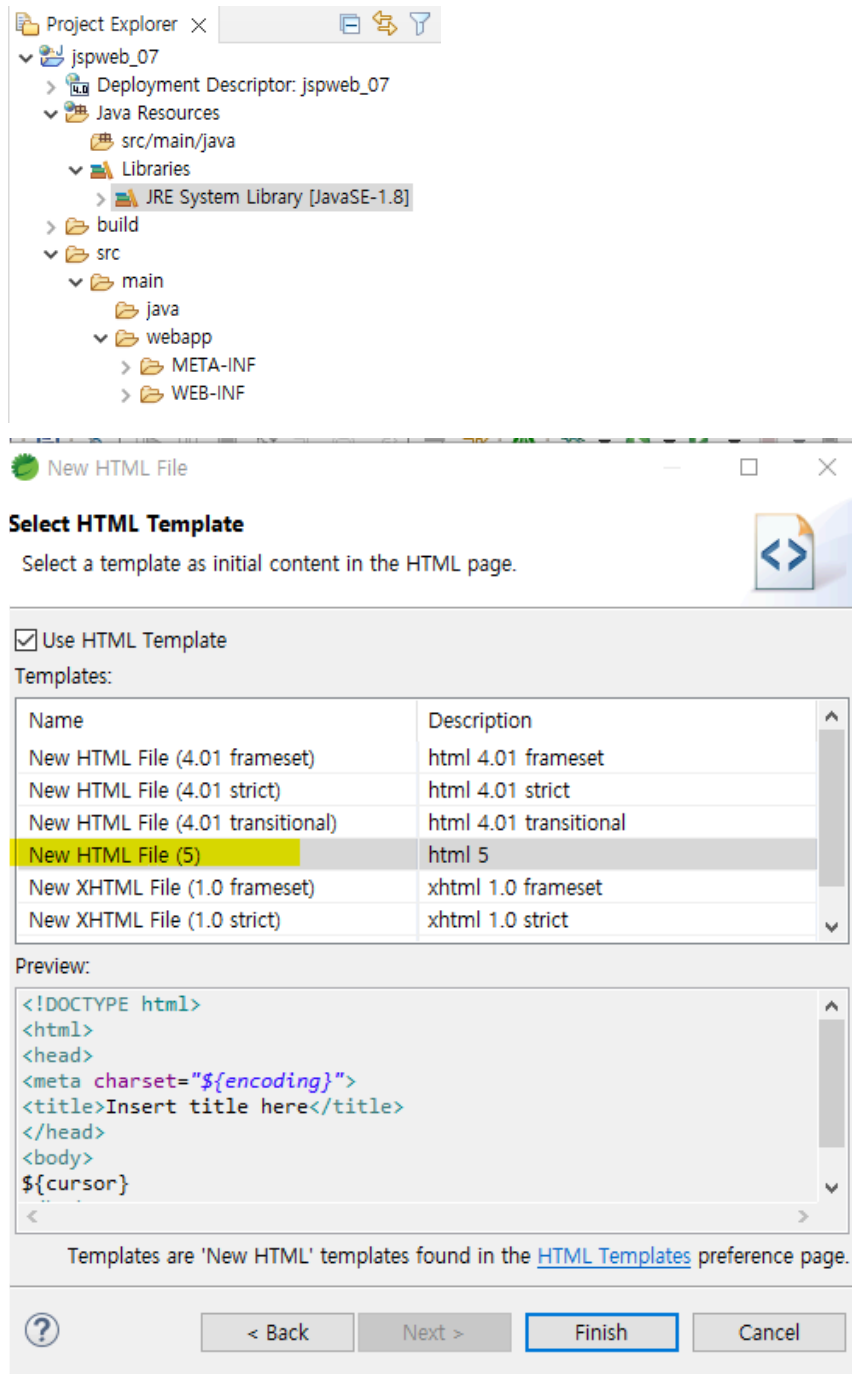


파일생성 file > new > others > web > dynamic

\*\* Dynamics web module: 기본 5인데 > 4로 변경 tomcat 9를 연동하기 위해서!



webapp > new > html



자바 사용시

src/main/java > new > class

<https://tomcat.apache.org/download-90.cgi> 톰캣 다운로드 > [64-bit Windows zip \(pgp, sha512\)](#)

톰캣웹서버는 tomcat 9 : java8, java11 이후 버전과 잘 맞는다

다운로드 - 압축해제 : d:\apache-tomcat-9

new - other - server - tomcat9 - 압축해제한 폴더를 연결

복습 : 다이내믹 웹프로젝트 jspweb\_07

톰캣웹서버 : 새로 만들고

jspweb\_07 -> webapp -> index.html, first.htm

## 9.0.85

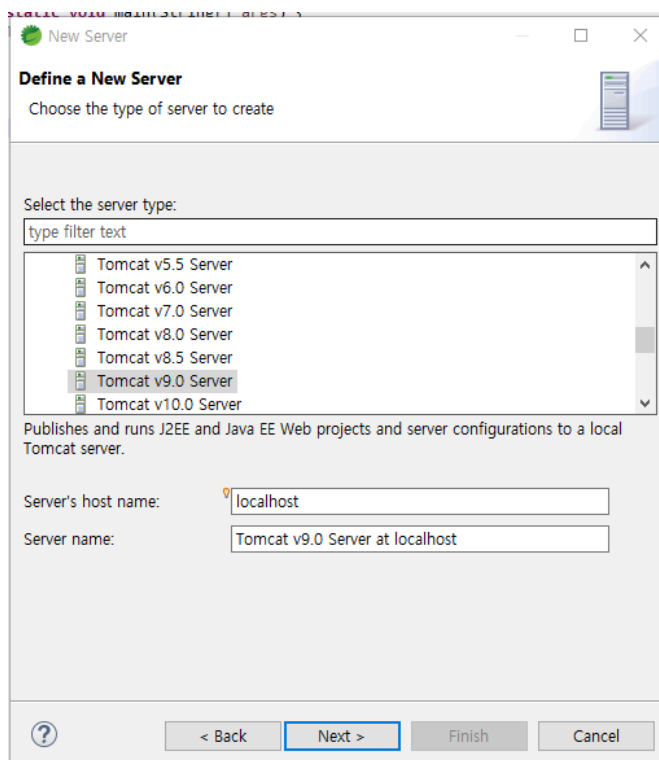
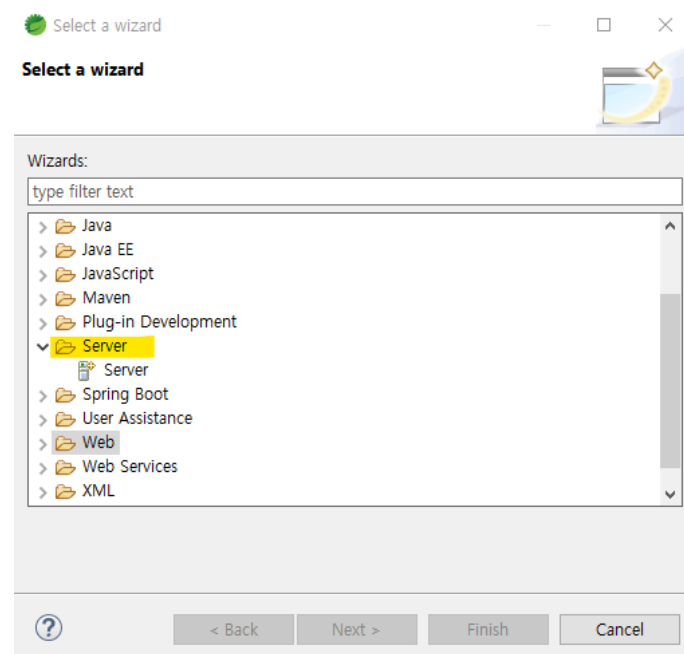
Please see the [README](#) file for packaging information. It explains wha

### Binary Distributions

- Core:
  - [zip \(pgp, sha512\)](#)
  - [tar.gz \(pgp, sha512\)](#)
  - [32-bit Windows zip \(pgp, sha512\)](#)
  - [64-bit Windows zip \(pgp, sha512\)](#)
  - [32-bit/64-bit Windows Service Installer \(pgp, sha512\)](#)
- Full documentation:

압축 풀기

다시 **spring**으로가서 만들어놓은 폴더에 오른쪽 마우스 클릭 > new > other > server





New Server

### Tomcat Server

Specify the Tomcat installation directory and JRE for this runtime. The specified

Name:  
Apache Tomcat v9.0

Tomcat installation directory:  
C:\tomcat\apache-tomcat-9.0.85-windows-x64\apache-tomcat-9.0.85  
Browse...  
apache-tomcat-9.0.82 Download and Install...

JRE:  
Workbench default JRE  
Installed JREs...

? < Back Next > Finish Cancel

New Server

### Add and Remove

Modify the resources that are configured on the server

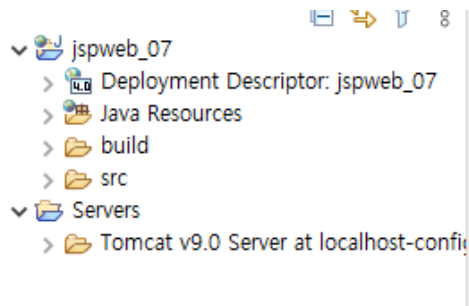
Move resources to the right to configure them on the server

Available:

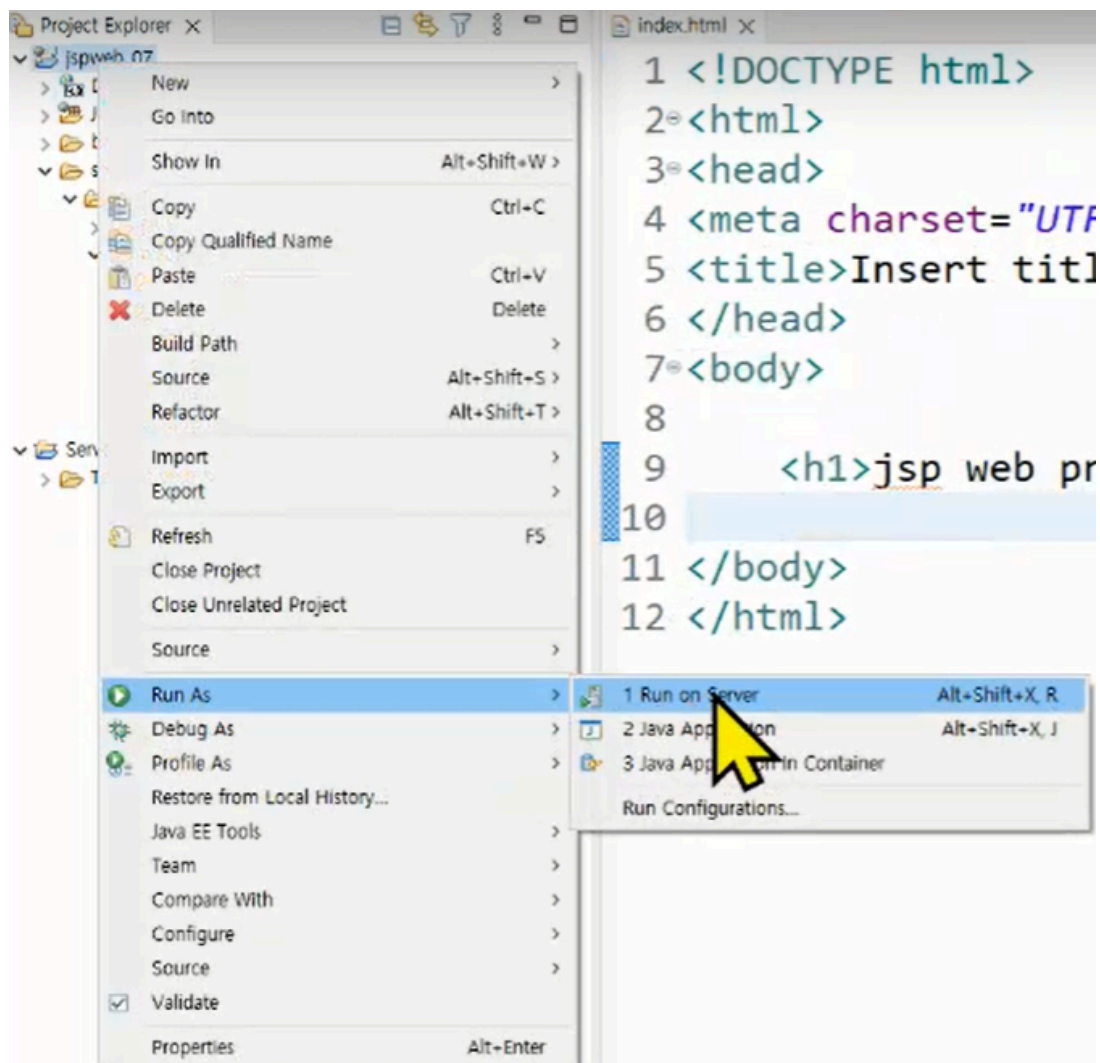
Configured:

Add >  
< Remove  
Add All >>  
<< Remove All

? < Back Next > Finish Cancel




src > main > webapp > new > html file



Run On Server

Run On Server

Select which server to use






How do you want to select the server?

☒ Choose an existing server

☐ Manually define a new server

Select the server that you want to use:


type filter text

Server	State
▼  localhost	
 Tomcat v9.0 Server at localhost	 Stopped

Apache Tomcat v9.0 supports J2EE 1.2, 1.3, 1.4, and Java EE 5, 6, 7, and 8 Web modules.

Columns...

☐ Always use this server when running this project



< Back

Next >


Finish

Cancel

Run On Server

Add and Remove

Modify the resources that are configured on the server



Move resources to the right to configure them on the server

Available:


Configured:


Add >

< Remove

Add All >>

<< Remove All

 jspweb\_07



< Back

Next >

Finish

Cancel

webapp 파일에서 new > JSP file 생성  
<% %> 맨 위는 jsp 파일이라는 설명을 해준다.

```
NewFile.html *main.jsp X
1 <%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
2 <!DOCTYPE html>
3 <html>
4 <head>
5 <meta charset="UTF-8">
6 <title>Insert title here</title>
7 </head>
8 <body>
9 <h1>jsp file sample</h1>
10 <%
11     out.println("hi");
12 %>
13 </body>
14 </html>
```

## 2. jsp 파일 이해

- 1) jsp (java servlet page) = html + java 를 혼합해 새롭게 만든 파일 형식이다
- 2) jsp 안에 자바코드를 작성하는 방법
  - <% 자바 연산 로직 코드%>
  - 두번째 <%= 출력하고 싶은 변수 %>
  - 세번째 <%! 함수정의 %> : 잘 안쓴다

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3 <!DOCTYPE html>
4 <html>
5 <head>
6 <meta charset="UTF-8">
7 <title>Insert title here</title>
8 </head>
9 <body>
10 <!-- 문자열 출력 --%>
11 <h1>Hello, JSP!</h1>
12
13 <!-- 숫자 출력 --%>
14 <%
15     int number = 123;
16     out.println("<p>Number: " + number + "</p>");
17 %>
18 </body>
19 </html>
```

<h1> 는 html  
<%> </%>는 자바

```

1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3 <!DOCTYPE html>
4 <html>
5 <head>
6 <meta charset="UTF-8">
7 <title>Insert title here</title>
8 </head>
9 <body>
10   <h2>간단한 연산 결과 출력</h2>
11
12   <!-- 자바코드 -->
13   <%
14     int num1 = 32;
15     int num2 = 16;
16
17     // 더하기, 빼기, 곱하기, 나누기 수행
18     double sum = num1 + num2;
19     double difference = num1 - num2;
20     double product = num1 * num2;
21     double quotient = (num2 != 0) ? num1 / num2 : Double.POSITIVE_INFINITY;
22   %>
23
24   <h2>계산 결과:</h2>
25   <p><%= num1 %> + <%= num2 %> = <%= sum %></p>
26   <p><%= num1 %> - <%= num2 %> = <%= difference %></p>
27   <p><%= num1 %> * <%= num2 %> = <%= product %></p>
28   <p><%= num1 %> / <%= num2 %> = <%= quotient %></p>
29
30 </body>
31 </html>

```

## 자바 vs 자바스크립트

### 예제 1

```

</head>
<body>
  <h2>간단한 연산 예제</h2>

  <%
    int apples = 232; //갯수
    int cntperbox = 20; // 사과박스에 20개씩 담을 수 있다

    // 사과박스 몇 개 필요?
    int totalbox = apples/cntperbox;
    // 남은 사과의 갯수는 몇개인가?

    int mod = apples % cntperbox;
  %>

  <p>총 사과 갯수: <%= apples %></p>
  <p>박스 갯수: <%= totalbox %></p>
  <p>남은 사과 갯수: <%= mod %></p>

</body>

```

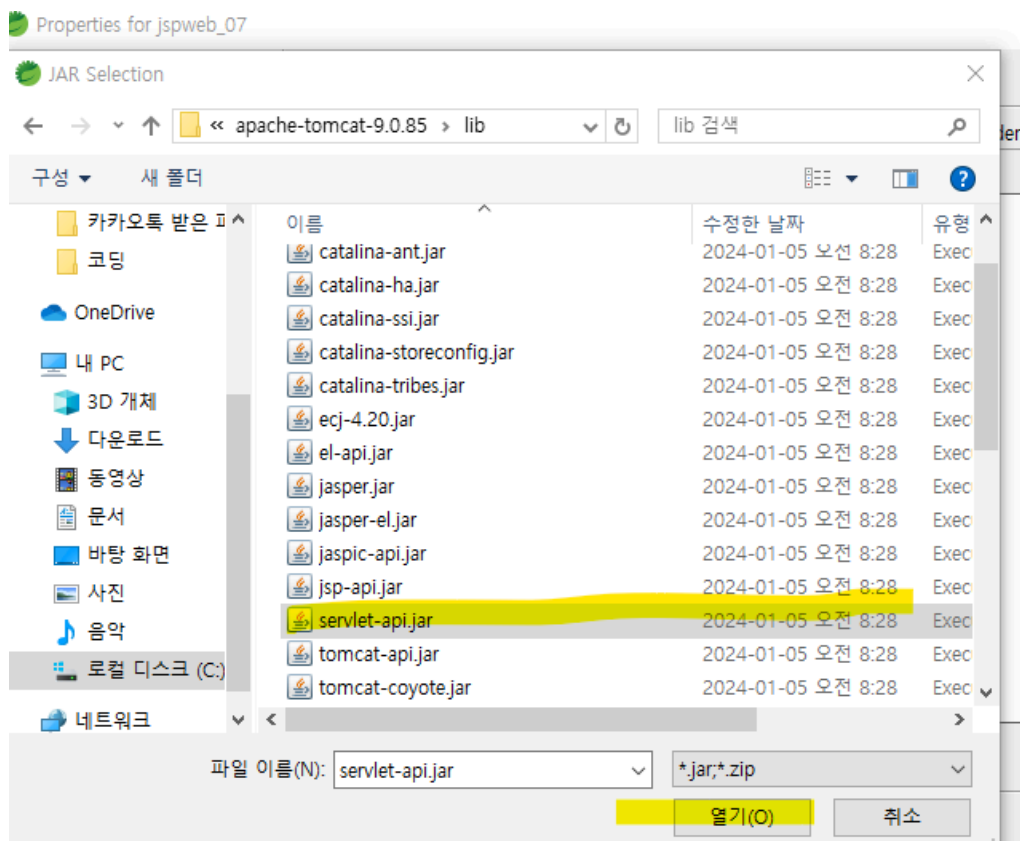
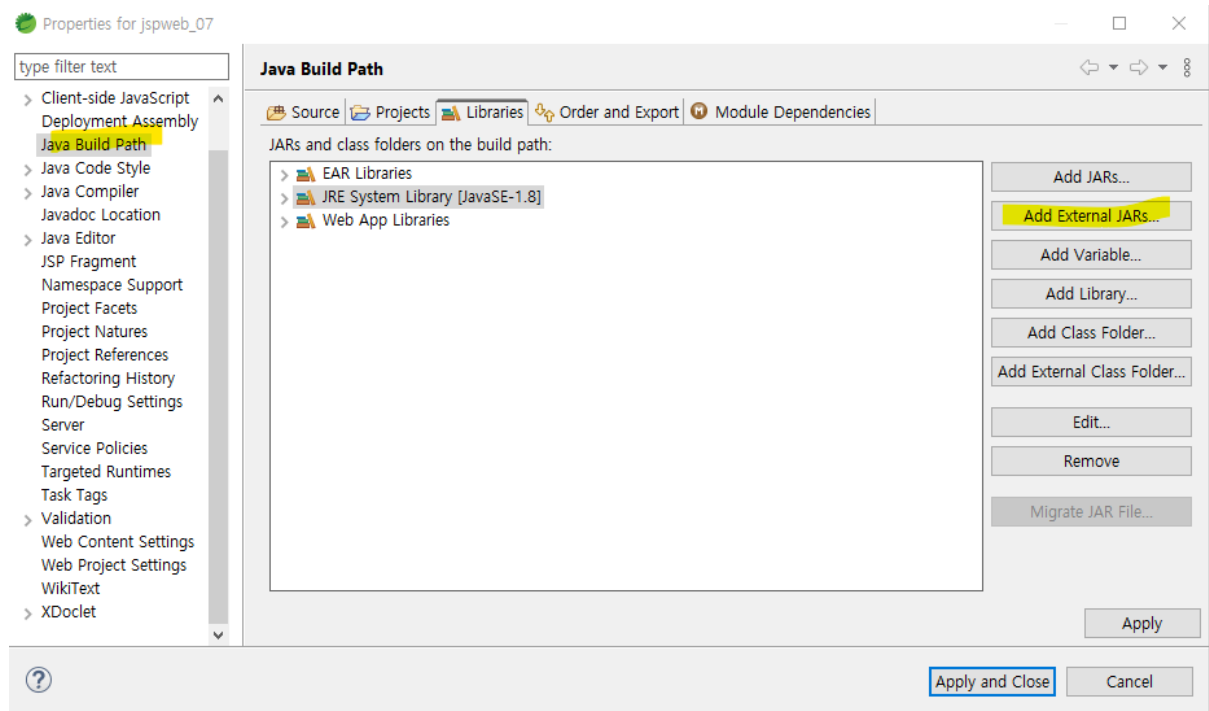
## 예제2

```
.  
:body>  
<h2>조건문 예제</h2>  
  
    <%  
        // 사용자의 나이 변수 설정  
        int age = 25;  
  
        // 조건문을 사용하여 나이에 따른 환영 메시지 출력  
        if (age < 18) {  
            out.println("<p>안녕하세요! 어린이 여러분!</p>");  
        } else if (age >= 18 && age < 30) {  
            out.println("<p>안녕하세요! 젊은이 여러분!</p>");  
        } else {  
            out.println("<p>안녕하세요! 성인 여러분!</p>");  
        }  
    %>  
    <div>  
:body>
```

## 예제3

```
<body>  
  
<h2>학점 계산 예제</h2>  
  
    <%  
        int score = 83;  
  
        //조건문을 A, B, C, D로 판별하는 코드 구현  
        //성적이 83이면 B  
  
        if(score >= 90){  
            out.println(score + "점수는 A학점입니다.");  
        }else if(score >= 80){  
            out.println(score + "점수는 B학점입니다.");  
        }else if(score >= 70){  
            out.println(score + "점수는 C학점입니다.");  
        }else{  
            out.println(score + "점수는 D학점입니다.");  
        }  
    %>  
    <div>  
</body>
```

에러뜨는거 지우기

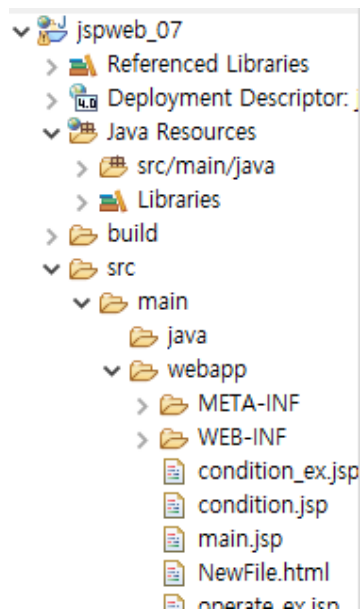


## 자바 라이브러리 импорт

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3 <% page import ="java.util.*"%>
4 <!DOCTYPE html>
5 <html>
6 <head>
7 <meta charset="UTF-8">
8 <title>Insert title here</title>
9 </head>
10 <body>
11
12 <h2>학점 계산 예제</h2>
13
14 <%
15     int score = 83;
16
17     //조건문을 A, B, C, D로 판별하는 코드 구현
18     //성적이 83이면 B
19
```

후 삭제하면 오류가 사라진다

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3
4 <!DOCTYPE html>
5 <html>
6 <head>
```





1월 11일 목요일

## 반복문 예제 1

```
1  <%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
2
3  <!DOCTYPE html>
4  <html>
5  <head>
6  <meta charset="UTF-8">
7  <title>Insert title here</title>
8  </head>
9  <body>
10  <%
11      out.println("반복문 예제");
12      out.println("<h2>테이블에 데이터를 넣는 예제</h2>");
13  %>
14
15
16  <%
17      // 사용자 리스트 (가정)
18      ArrayList<String> userList = new ArrayList<>();
19      userList.add("사용자1");
20      userList.add("사용자2");
21      userList.add("사용자3");
22
23      // 테이블 시작
24      out.println("<table border='1'>");
25
26      // 테이블 헤더
27      out.println("<tr>");
28      out.println("<th>ID</th>");
29      out.println("<th>이름</th>");
30      out.println("</tr>");
31
32      // 사용자 리스트를 테이블에 추가
33      for (String user : userList) {
34          out.println("<tr>");
35          out.println("<td>" + user + "</td>");
36          out.println("</tr>");
37      }
38
39      // 테이블 종료
40      out.println("</table>");
41  %>
42
43  </body>
44  </html>
```

Array에서 빨간 밑줄이 그어져 있다

자바를 임포트 해준다 > <%@ page import = "java.util.\*" %>

## index 추가

```
<%
// 사용자 리스트 (가정)
ArrayList<String> userList = new ArrayList<>();
userList.add("사용자1");
userList.add("사용자2");
userList.add("사용자3");

// 테이블 시작
out.println("<table border='1'>");

// 테이블 헤더
out.println("<tr>");
out.println("<th>ID</th>");
out.println("<th>이름</th>");
out.println("</tr>");

// 사용자 리스트를 테이블에 추가
int index = 0;
for (String user : userList) {
    out.println("<tr>");
    out.println("<td>" + index + "</td>");
    out.println("<td>" + user + "</td>");
    out.println("</tr>");
}

// 테이블 종료
out.println("</table>");
%>
```

## 반복문 예제2

```
for2.jsp ×
1  <%@ page language="java" contentType="text/html; charset=UTF-8"
2     pageEncoding="UTF-8"%>
3
4  <%@ page import = "java.util.*" %>
5  <!DOCTYPE html>
6  <html>
7  <head>
8  <meta charset="UTF-8">
9  <title>Insert title here</title>
10 </head>
11 <body>
12     <h2>jsp를 사용해 테이블 구현</h2>
13
14     <%
15         // 이름 저장 배열
16         ArrayList name = new ArrayList();
17         name.add("김씨");
18         name.add("이씨");
19         name.add("박씨");
20         ArrayList departName = new ArrayList();
21         departName.add("기획팀");
22         departName.add("개발팀");
23         departName.add("보안팀");
24
25         out.println("<table border = '1'>");
26         out.println("<tr>");
27         out.println("<th>직원명</th>");
28         out.println("<th>부서명</th>");
29         out.println("</tr>");
30
31
32         out.print("</table>");
33     %>
34
35
36 </body>
37 </html>
```

노란색 오류는 **ArrayList**에 **<String>** 넣어서 없애줄 수 있다: 문자열 정의

```
<%
// 이름 저장 배열
ArrayList<String> name = new ArrayList<String>();
name.add("김씨");
name.add("이씨");
name.add("박씨");
ArrayList<String> departName = new ArrayList<String>();
departName.add("기획팀");
departName.add("개발팀");
departName.add("보안팀");

out.println("<table border = '1'>");
out.println("<tr>");
out.println("<th>직원명</th>");
out.println("<th>부서명</th>");
out.println("</tr>");

out.print("</table>");
%>
```

자바 - size() 자바스크립트 .length

```
<%
// 이름 저장 배열
ArrayList<String> name = new ArrayList<String>();
name.add("김씨");
name.add("이씨");
name.add("박씨");
ArrayList<String> departName = new ArrayList<String>();
departName.add("기획팀");
departName.add("개발팀");
departName.add("보안팀");

out.println("<table border = '1'>");
out.println("<tr>");
out.println("<th>직원명</th>");
out.println("<th>부서명</th>");
out.println("</tr>");

for(int i=0; i<name.size(); i++){

    out.println("<tr>");
    out.println("<td>" + name.get(i) + "</td>");
    out.println("<td>" + departName.get(i) + "</td>");
    out.println("</tr>");
}

out.print("</table>");
%>
```

## ❖ 폼 구현 후 데이터 전송하기

### 예제 1

```
*for_client.jsp ×
1  <%@ page language="java" contentType="text/html; charset=UTF-8"
2      pageEncoding="UTF-8"%>
3  <!DOCTYPE html>
4  <html>
5  <head>
6  <meta charset="UTF-8">
7  <title>Insert title here</title>
8  </head>
9  <body>
10     <h2>폼 데이터 전송 화면</h2>
11
12     <!-- action: 결과를 전송해서 표시할 jsp파일, GET: 값을 텍스트형태로 넘겨준다-->
13     <form action = "pro_server.jsp" method = "GET">
14         <p>
15             <label>이름:</label>
16             <input type = "text" name = "name" >
17         </p>
18
19         <p>
20             <label>이메일:</label>
21             <input type = "text" name = "email" >
22         </p>
23
24         <p>
25             <input type = "submit" value = "전송" >
26         </p>
27
28     </form>
29 </body>
30 </html>
```

pro\_server.jsp 를 만들어준다

```
pro_server.jsp ×
1  <%@ page language="java" contentType="text/html; charset=UTF-8"
2      pageEncoding="UTF-8"%>
3  <!DOCTYPE html>
4  <html>
5  <head>
6  <meta charset="UTF-8">
7  <title>Insert title here</title>
8  </head>
9  <body>
10
11     <%
12         // 폼 데이터 수신
13         //getParameter 변수의 이름을 알고 저장해준다
14         String name = request.getParameter("name");
15         String email = request.getParameter("email");
16
17         // 폼 데이터 출력
18         out.println("<h2>폼 데이터 출력 결과</h2>");
19         out.println("<p>이름: " + name + "</p>");
20         out.println("<p>Email: " + email + "</p>");
21     %>
22
23 </body>
24 </html>
```

## 예제2 아이디와 비밀번호 데이터 전송

```
form_login.jsp ×
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3 <!DOCTYPE html>
4 <html>
5 <head>
6   <meta charset="UTF-8">
7   <title>Insert title here</title>
8 </head>
9 <body>
10   <h2>로그인 폼 전송 예제</h2>
11
12 <form action = "pro_login.jsp" method = "GET">
13
14   <p>
15     <label>아이디:</label>
16     <input type = "text" name = "id" >
17   </p>
18
19   <p>
20     <label>비밀번호:</label>
21     <input type = "password" name = "pw" >
22   </p>
23
24   <p>
25     <input type = "submit" value = "send" >
26   </p>
27
28 </form>
29
30 </body>
31 </html>
```

또는 아래처럼 쓸 수 있다 <label> 을 지우고 placeholder = ""

```
<body>
  <h2>로그인 폼 전송 예제</h2>

  <form action = "pro_login.jsp" method = "GET">

    <p><input type = "text" name = "id" placeholder="아이디 입력"></p>
    <p><input type = "password" name = "pw" placeholder="비밀번호 입력"></p>

    <p><input type = "submit" value = "send" ></p>

  </form>
```

pro\_login.jsp 를 만들어준다 >>

### 폼 데이터 출력 결과

이름: 김

Email: test

```
1 <%@ page language="java" contentType="text/html; charset=UTF-
2   pageEncoding="UTF-8"%>
3 <!DOCTYPE html>
4 <html>
5 <head>
6   <meta charset="UTF-8">
7   <title>Insert title here</title>
8 </head>
9 <body>
10 <%
11
12   //아이디와 비밀번호를 받아 출력해라
13   String id = request.getParameter("id");
14   String pw = request.getParameter("pw");
15
16   out.println("<h2>폼 데이터 출력 결과</h2>");
17   out.println("<p>아이디" + id + "</p>");
18   out.println("<p>비밀번호" + pw + "</p>");
19
20   %>
21 </body>
22 </html>
```

## ❖ GET 과 POST 의 차이

GET 사용시는 주소창에 값이 보이지만

`http://localhost:8080/jspweb_07/pro_login.jsp?id=luna&pw=1234`

POST는 보이지 않는다

`http://localhost:8080/jspweb_07/pro_login.jsp`

이해하기 위해 GET 사용을 많이 한다

## ❖ jsp 파일을 합치는 예제

`<%@ include file = "다른 jsp 파일명" %>`

상단메뉴 : header.jsp

메인: main2.jsp

```
header.jsp  main2.jsp ×
1  <%@ page language="java" contentType="text/h
2      pageEncoding="UTF-8"%>
3  <!DOCTYPE html>
4  <html>
5  <head>
6  <meta charset="UTF-8">
7  <title>Insert title here</title>
8  </head>
9  <body>
10
11      <!-- 이미 만들어진 header 파일을 include -->
12      <%@ include file="header.jsp" %>
13
14      <!-- 현재 작업하고 있는 파일 -->
15      <div id = "myworkcontent">
16          <h1>홈페이지 소개</h1>
17          <h2>새로운 콘텐츠를 작성 중입니다...</h2>
18          <h2>처리 중 입니다...</h2>
19      </div>
20
21  </body>
22  </html>
```

```
header.jsp ×  main2.jsp
1  <%@ page language="java" contentType="text/html;
2      pageEncoding="UTF-8"%>
3  <!DOCTYPE html>
4  <html>
5  <head>
6  <meta charset="UTF-8">
7  <title>Insert title here</title>
8  </head>
9  <body>
10
11      <div id="header">
12          <ul>
13              <li><a href="#">Home</a></li>
14              <li><a href="#">About</a></li>
15              <li><a href="#">Services</a></li>
16              <li><a href="#">Contact</a></li>
17          </ul>
18      </div>
19
20  </body>
21  </html>
```

## JSP 반복문을 사용해 구구단 만들기

### 구구단

2단	3단	4단	5단	6단	7단	8단	9단
2x1=2	3x1=3	4x1=4	5x1=5	6x1=6	7x1=7	8x1=8	9x1=9
2x2=4	3x2=6	4x2=8	5x2=10	6x2=12	7x2=14	8x2=16	9x2=18
2x3=6	3x3=9	4x3=12	5x3=15	6x3=18	7x3=21	8x3=24	9x3=27
2x4=8	3x4=12	4x4=16	5x4=20	6x4=24	7x4=28	8x4=32	9x4=36
2x5=10	3x5=15	4x5=20	5x5=25	6x5=30	7x5=35	8x5=40	9x5=45
2x6=12	3x6=18	4x6=24	5x6=30	6x6=36	7x6=42	8x6=48	9x6=54
2x7=14	3x7=21	4x7=28	5x7=35	6x7=42	7x7=49	8x7=56	9x7=63
2x8=16	3x8=24	4x8=32	5x8=40	6x8=48	7x8=56	8x8=64	9x8=72
2x9=18	3x9=27	4x9=36	5x9=45	6x9=54	7x9=63	8x9=72	9x9=81

```
<body>
  <h2>구구단</h2>

  <%

    out.println("<table border = '1'>");
    out.println("<tr>"); //행만들어 주기

    for(int i=2; i<10; i++){
      out.println("<th>" + i + "단</th>"); //행 상단 제목
    }
    out.println("</tr>");

    for(int i=1; i<10; i++){
      out.println("<tr>");

      for (int j=2; j<10; j++){
        //컬럼 안 값
        out.println("<td>" + j + "x" + i + "=" + (i*j) + "</td>");
      }
      out.println("</tr>");
    }

    out.println("</table>");

  %>

</body>
```

## 8장 데이터베이스와 JSP

자바스크립트 vs 자바

var 정수나 실수 <==> int와 double 로 구분

-----

-- 관리자 ora\_user 로 접속해서 test 계정 생성

```
CREATE USER test IDENTIFIED BY 123456 DEFAULT TABLESPACE USERS;
```

```
GRANT CONNECT, RESOURCE TO test;
```

```
ALTER USER test DEFAULT TABLESPACE USERS QUOTA UNLIMITED ON USERS;
```

-- test, jsp, ....

```
CREATE USER jsp IDENTIFIED BY 123456 DEFAULT TABLESPACE USERS;
```

```
GRANT CONNECT, RESOURCE TO jsp;
```

```
ALTER USER jsp DEFAULT TABLESPACE USERS QUOTA UNLIMITED ON USERS;
```

- 오라클에서 EMPLOYEES 테이블 생성

```
CREATE TABLE EMPLOYEES (  
    EMPLOYEE_ID NUMBER PRIMARY KEY,  
    FIRST_NAME VARCHAR2(50),  
    LAST_NAME VARCHAR2(50),  
    EMAIL VARCHAR2(100),  
    PHONE_NUMBER VARCHAR2(20),  
    HIRE_DATE DATE,  
    JOB_ID VARCHAR2(50),  
    SALARY NUMBER,  
    MANAGER_ID NUMBER,  
    DEPARTMENT_ID NUMBER  
);
```



- 테이블에 데이터 추가

--INSERT INTO 테이블명 (컬럼명,...) VALUES (값,...) 맞춰서 써주기

```
INSERT INTO EMPLOYEES VALUES (1, 'John', 'Doe', 'john.doe@example.com',  
'123-456-7890', TO_DATE('2022-01-01', 'YYYY-MM-DD'), 'MANAGER', 50000, NULL, 10);  
INSERT INTO EMPLOYEES VALUES (2, 'Jane', 'Smith', 'jane.smith@example.com',  
'987-654-3210', TO_DATE('2022-02-01', 'YYYY-MM-DD'), 'CLERK', 30000, 1, 20);  
INSERT INTO EMPLOYEES VALUES (3, 'Bob', 'Johnson', 'bob.johnson@example.com',  
'555-123-4567', TO_DATE('2022-03-01', 'YYYY-MM-DD'), 'ANALYST', 60000, 1, 30);
```

- 테이블 데이터 수정하는 구문

--UPDATE 테이블명 SET 컬럼명=수정값, , , WHERE 조건  
-- EMPLOYEE\_ID가 2인 직원의 정보를 수정

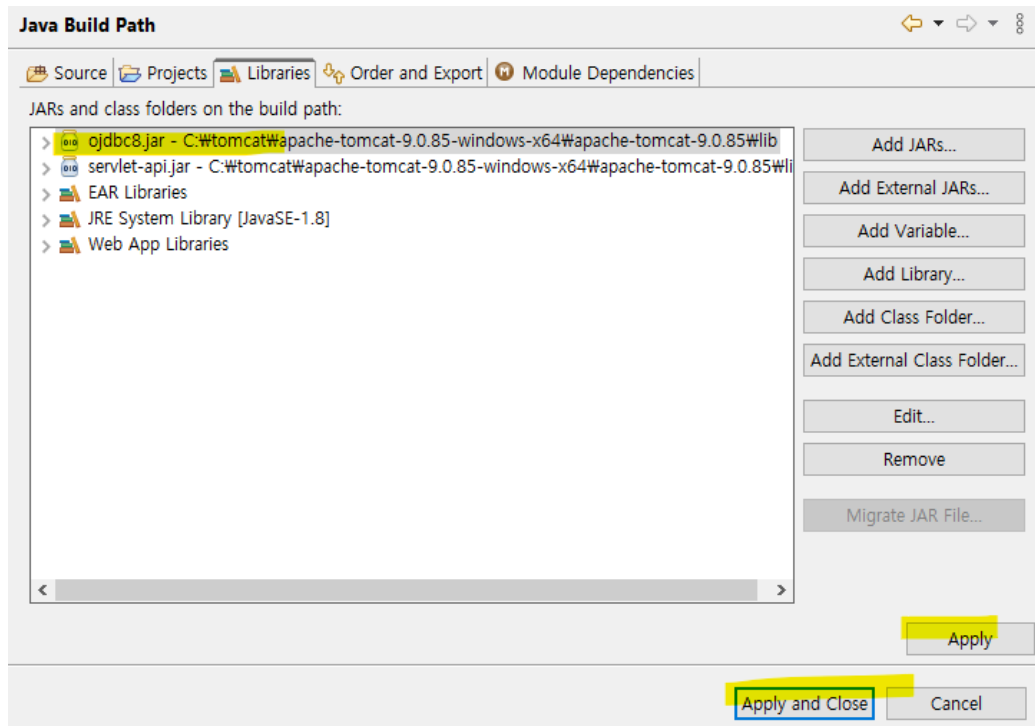
```
UPDATE EMPLOYEES  
SET FIRST_NAME = 'UpdatedFirstName',  
    LAST_NAME = 'UpdatedLastName',  
    EMAIL = 'updated.email@example.com',  
    PHONE_NUMBER = '999-999-9999',  
    HIRE_DATE = TO_DATE('2022-04-01', 'YYYY-MM-DD'),  
    JOB_ID = 'UPDATED_JOB',  
    SALARY = 40000,  
    MANAGER_ID = 1,  
    DEPARTMENT_ID = 20  
WHERE EMPLOYEE_ID = 2;
```

- 테이블 삭제

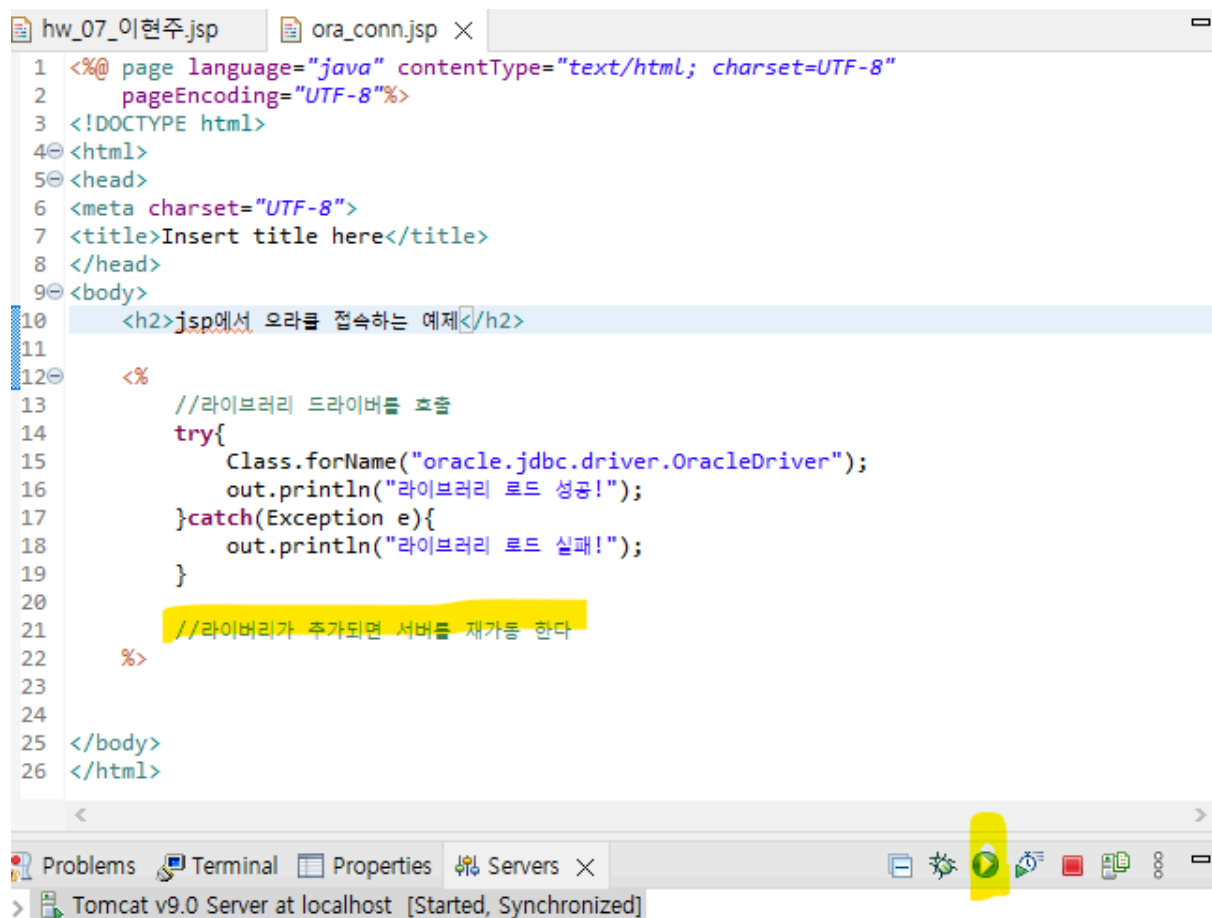
```
DELETE FROM EMPLOYEES  
WHERE EMPLOYEE_ID = 2;
```

파일은 **apache tomcat lib** 파일에 없다

## 2. ojdbc8.jar 파일을 먼저 JSP apache tomcat lib파일에 복사붙여넣기.



### 3. ora\_conn.jsp 파일 만들고 라이브러리 로드 > 재가동 해주기!!



#### 4. Oracle 데이터 베이스 연결 정보 입력

```
hw_07_이현주.jsp *ora_conn.jsp X
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3
4 <%@ page import = "java.sql.*" %>
5 <!DOCTYPE html>
6 <html>
7 <head>
8   <meta charset="UTF-8">
9   <title>Insert title here</title>
10 </head>
11 <body>
12   <h2>jsp에서 오라클 접속하는 예제</h2>
13
14   <%
15     //라이브러리 드라이버를 호출
16     try{
17       Class.forName("oracle.jdbc.driver.OracleDriver");
18       out.println("라이브러리 로드 성공!");
19     }catch(Exception e){
20       out.println("라이브러리 로드 실패!");
21     }
22
23     //라이버리가 추가되면 서버를 재가동 한다
24
25
26     // Oracle 데이터베이스 연결 정보 설정
27     final String JDBC_URL = "jdbc:oracle:thin:@localhost:1521:orcl";
28     final String USER = "jsp";
29     final String PASSWORD = "123456";
30
31     //접속하기
32     try{
33       DriverManager.getConnection(JDBC_URL, USER, PASSWORD);
34       out.println("jsp계정 접속성공");
35     }catch(Exception e){
36       out.println("jsp계정 접속실패");
37     }
38   }
39
40   %>
41
42 </body>
43 </html>
```

<body>

<h2>jsp에서 오라클 접속하는 예제</h2>

<%

//라이브러리 드라이버를 호출

```
try{
    Class.forName("oracle.jdbc.driver.OracleDriver");
    out.println("라이브러리 로드 성공!");
}catch(Exception e){
    out.println("라이브러리 로드 실패!");
}
```

//라이버리가 추가되면 서버를 재가동 한다

// Oracle 데이터베이스 연결 정보 설정

```
final String JDBC_URL = "jdbc:oracle:thin:@localhost:1521:orcl";
final String USER = "jsp";
final String PASSWORD = "123456";
```

//접속하기

```
try{
    DriverManager.getConnection(JDBC_URL, USER, PASSWORD);
    out.println("jsp계정 접속성공");
}catch(Exception e){
    out.println("jsp계정 접속실패");
}
```

%>

</body>

Run 을 해주면 아래처럼 화면이 나온다

## jsp에서 오라클 접속하는 예제

라이브러리 로드 성공! jsp계정 접속성공

## 1. 새로운 파일 만들기

### File > New > Dynamic Web Project

New Dynamic Web Project

**Dynamic Web Project**  
Create a standalone Java-based Web Application or add it to a new or existing Enterprise Application.

Project name:

Project location  
☒ Use default location  
Location:

Target runtime

Dynamic web module version

Configuration  
   
Hint: Get started quickly by selecting one of the pre-defined project configurations.

EAR membership  
☐ Add project to an EAR  
EAR project name:

Working sets  
☐ Add project to working sets   
Working sets:

## 2. Java Build Path 에서 라이브러리 2개 넣어주기

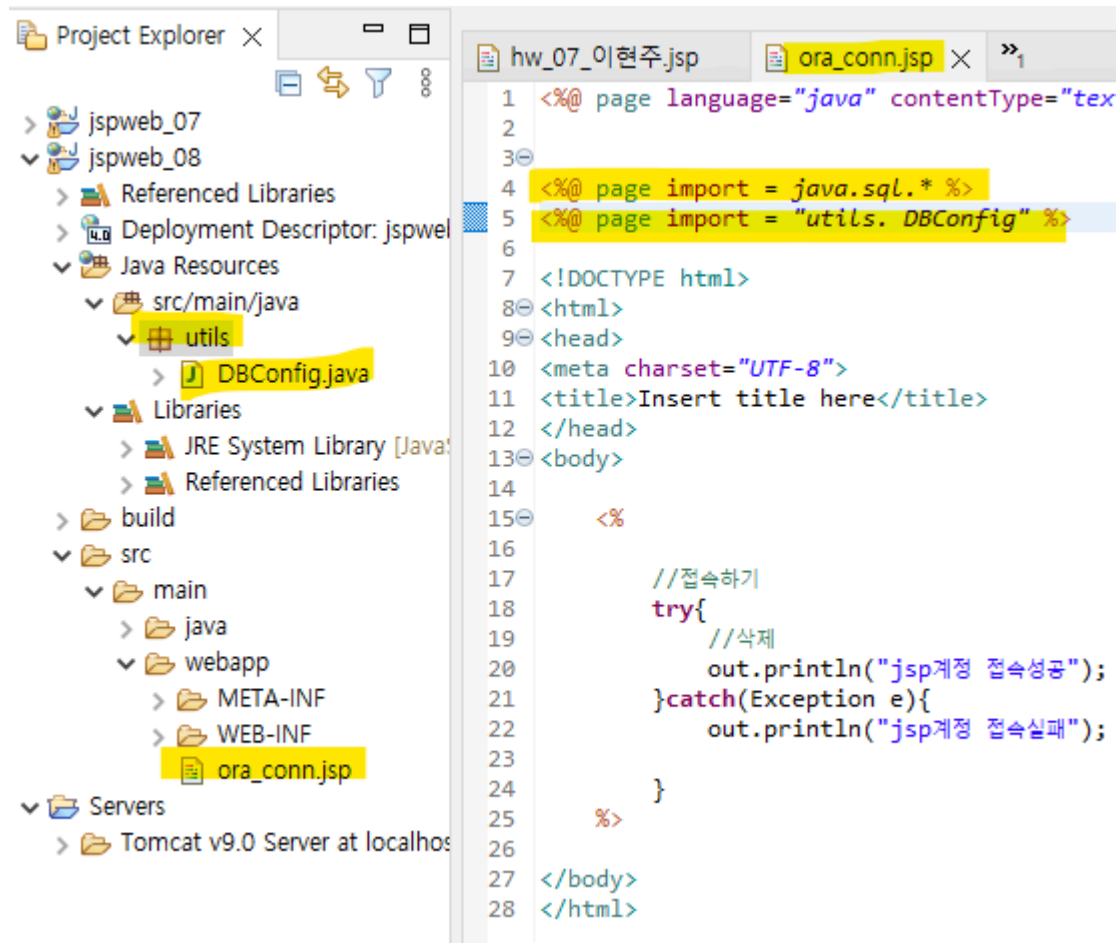
Java Build Path

Source Projects Libraries Order and Export Module Dependencies

JARs and class folders on the build path:

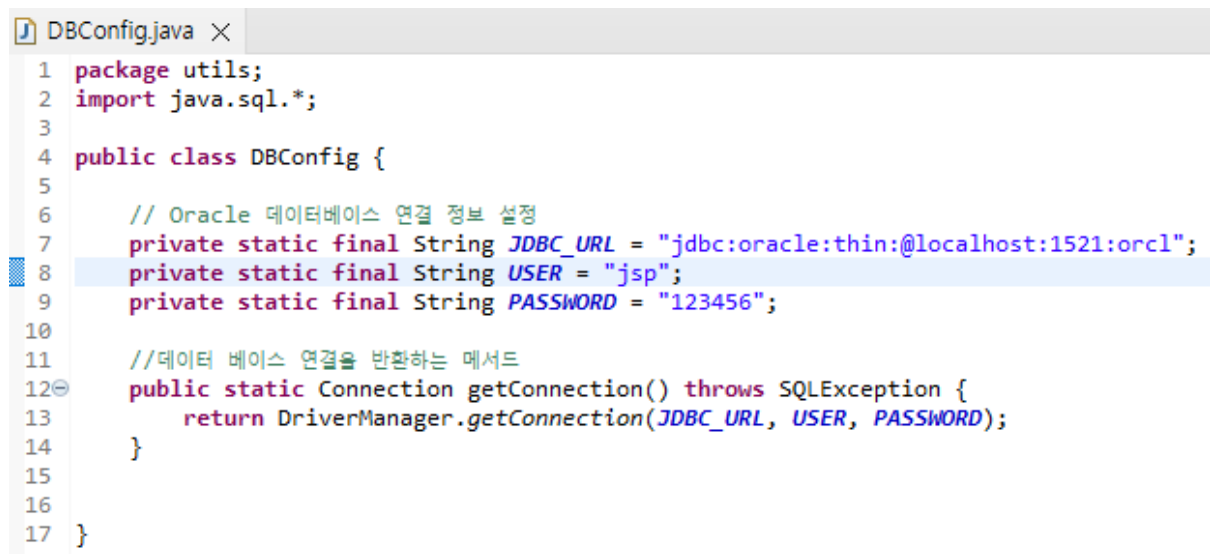
- ojdbc8.jar - C:\tomcat\apache-tomcat-9.0.85-windows-x64\apache-tomcat-9.0.85\lib
- servlet-api.jar - C:\tomcat\apache-tomcat-9.0.85-windows-x64\apache-tomcat-9.0.85\lib
- EAR Libraries
- JRE System Library [JavaSE-1.8]
- Web App Libraries

### 3. 패키지 **utils**, 클래스 **DBConfig.java** 그리고 **ora\_conn.jsp** 만들어준다



### 4. **ora\_conn.jsp** 만들때

**<%@ page import = "java.util.\*" %>** 넣고 빼주기



```
// Oracle 데이터베이스 연결 정보 설정
private static final String JDBC_URL = "jdbc:oracle:thin:@localhost:1521:orcl";
private static final String USER = "jsp";
private static final String PASSWORD = "123456";

//데이터 베이스 연결을 반환하는 메서드
public static Connection getConnection() throws SQLException {
    return DriverManager.getConnection(JDBC_URL, USER, PASSWORD);
}
```

## 오라클 DB 입력

```
-- 관리자 계정 ora_user
-- 사용자 2번 jsp 계정
```

```
-- EMPLOYEES 테이블 생성
CREATE TABLE EMPLOYEES (
    EMPLOYEE_ID NUMBER PRIMARY KEY,
    FIRST_NAME VARCHAR2(50),
    LAST_NAME VARCHAR2(50),
    EMAIL VARCHAR2(100),
    PHONE_NUMBER VARCHAR2(20),
    HIRE_DATE DATE,
    JOB_ID VARCHAR2(50),
    SALARY NUMBER,
    MANAGER_ID NUMBER,
    DEPARTMENT_ID NUMBER
);
```

```
--EMPLOYEES 테이블 조회
SELECT * FROM EMPLOYEES;
```

```
--테이블에 데이터 추가
--INSERT INTO 테이블명 (컬럼명,...) VALUES (값,...) 맞춰서 써주기
```

```
INSERT INTO EMPLOYEES VALUES (1, 'John', 'Doe', 'john.doe@example.com',
'123-456-7890', TO_DATE('2022-01-01', 'YYYY-MM-DD'), 'MANAGER', 50000, NULL, 10);
INSERT INTO EMPLOYEES VALUES (2, 'Jane', 'Smith', 'jane.smith@example.com',
'987-654-3210', TO_DATE('2022-02-01', 'YYYY-MM-DD'), 'CLERK', 30000, 1, 20);
INSERT INTO EMPLOYEES VALUES (3, 'Bob', 'Johnson', 'bob.johnson@example.com',
'555-123-4567', TO_DATE('2022-03-01', 'YYYY-MM-DD'), 'ANALYST', 60000, 1, 30);
```

```
--테이블 데이터 수정하는 구문
--UPDATE 테이블명 SET 컬럼명=수정값, , WHERE 조건
```



```
-- EMPLOYEE_ID가 2인 직원의 정보를 수정
UPDATE EMPLOYEES
SET FIRST_NAME = 'UpdatedFirstName',
    LAST_NAME = 'UpdatedLastName',
    EMAIL = 'updated.email@example.com',
    PHONE_NUMBER = '999-999-9999',
    HIRE_DATE = TO_DATE('2022-04-01', 'YYYY-MM-DD'),
    JOB_ID = 'UPDATED_JOB',
    SALARY = 40000,
    MANAGER_ID = 1,
    DEPARTMENT_ID = 20
WHERE EMPLOYEE_ID = 2;
```

```
--테이블 삭제
DELETE FROM EMPLOYEES
WHERE EMPLOYEE_ID = 2;
```

```
SELECT count(*) as CNT FROM EMPLOYEES;
```

```
CREATE TABLE DEPARTMENTS(
    DEPART_ID NUMBER PRIMARY KEY,
    DEPART_NAME VARCHAR2(50),
    LOCATION VARCHAR2(100),
    PHONE_NUMBER VARCHAR(20)
);
```

```
INSERT INTO DEPARTMENTS VALUES(10, 'Sales', 'New York', '555-1234');
INSERT INTO DEPARTMENTS VALUES(20, 'Marketing', 'Los Angeles', '555-5678');
INSERT INTO DEPARTMENTS VALUES(30, 'IT', 'San Francisco', '555-9876');
```

```
commit;
```

1. 디비에서 아래를 확인 값이 = 2  
SELECT count(\*) as CNT FROM EMPLOYEES;

## 2. jspweb\_07의 ora\_conn.jsp파일에

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

<%@ page import = "java.sql.*" %>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
    <h2>jsp에서 오라클 접속하는 예제</h2>

    <%
        // 1 라이브러리 드라이버를 호출
        try {
            Class.forName("oracle.jdbc.driver.OracleDriver");
            out.println("라이브러리 로드성공");
        } catch (Exception e) {
            out.println("라이브러리 로드실패");
        }
        // 라이브러리가 추가되면 서버를 보통 재기동

        // 접속관련 정보 저장
        Connection connection = null;
        Statement statement = null;
        ResultSet resultSet = null;

        // 2 접속
        // Oracle 데이터베이스 연결 정보 설정
        final String JDBC_URL = "jdbc:oracle:thin:@localhost:1521:orcl";
        final String USER = "test";
        final String PASSWORD = "1234";
        // 접속하기
        try {
            connection = DriverManager.getConnection(JDBC_URL, USER, PASSWORD);
            out.println("jsp계정 접속성공");
        } catch (Exception e) {
            out.println("jsp계정 접속실패");
        }

        // SQL 쿼리 실행
        try {
            String sqlQuery = "SELECT count(*) as CNT FROM EMPLOYEES";
            statement = connection.createStatement();
            resultSet = statement.executeQuery(sqlQuery);
```

```
        while(resultSet.next()){
            String total = resultSet.getString("CNT");
            out.println("<h2>총갯수: " + total + "</h2>");
        }
    }catch(SQLException e) {

    }

    %>

</body>
</html>
```

## jsp에서 오라클 접속하는 예제

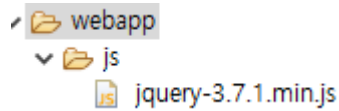
라이브러리 로드성공 jsp계정 접속성공

**총갯수: 2**

0이 나올수도 있는데 오라클에서 커밋을 해줘야 한다

1월 12일 금요일

1. `<script src="js/jquery-3.7.1.min.js"></script>` 를 활용할 수 있도록 **js** 폴더 만들어주고 라이브러리 넣어주기



2. **ora\_conn.jsp** 파일과 **src**의 **utils** 패키지의 **DBConfig.java** 코드 넣기

### ora\_conn.jsp

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

```
<%@ page import = "java.sql.*" %>
<%@ page import = "utils.DBConfig.*" %>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
```

```
    <%
    // 1 라이브러리 드라이버를 호출
    /* try {
        Class.forName("oracle.jdbc.driver.OracleDriver");
        out.println("라이브러리 로드성공");
    }catch(Exception e) {
        out.println("라이브러리 로드실패");
    } */

    out.println("<br>");
    // 라이브러리가 추가되면 서버를 보통 재기동
```

```
    // 접속관련 정보 저장
    Connection connection = null;
    Statement statement = null;
    ResultSet resultSet = null;
```

```
    // 2 접속
    // Oracle 데이터베이스 연결 정보 설정
    final String JDBC_URL = "jdbc:oracle:thin:@localhost:1521:orcl";
    final String USER = "test";
    final String PASSWORD = "1234";
    // 접속하기
    try {
        connection = DriverManager.getConnection(JDBC_URL, USER, PASSWORD);
        out.println("jsp계정 접속성공");
    }catch(Exception e) {
```

```

        out.println("jsp계정 접속실패");

    }

    %>

</body>
</html>

```

### 3. DBConfig.java 만들기

```

package utils;
import java.sql.*;
public class DBConfig {
    // Oracle 데이터베이스 연결 정보 설정
    private static final String JDBC_URL = "jdbc:oracle:thin:@localhost:1521:orcl";
    private static final String USER = "test";
    private static final String PASSWORD = "1234";

    //데이터 베이스 연결을 반환하는 메서드
    public static Connection getConnection() throws SQLException {
        return DriverManager.getConnection(JDBC_URL, USER, PASSWORD);
    }
}

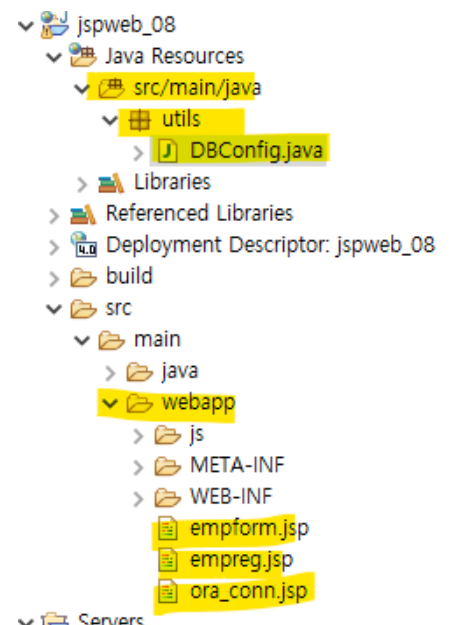
```

### 4. empform.jsp 폴더 만들기. 직원 정보 등록 폼 출력화면

```

<%@ 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>
<style>
    body {
        font-family: Arial, sans-serif;
        background-color: #f2f2f2;
        margin: 0;
        padding: 0;
        display: flex;
        justify-content: center;
        align-items: center;
        height: 100vh;
    }
    .employee-form {

```



```

        background-color: #fff;
        padding: 20px;
        border-radius: 5px;
        box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
    }
    form {
        display: flex;
        flex-direction: column;
    }
    label {
        margin-bottom: 5px;
    }
    input {
        padding: 8px;
        margin-bottom: 10px;
        border: 1px solid #ccc;
        border-radius: 3px;
    }
    button {
        background-color: #3498db;
        color: #fff;
        padding: 10px;
        border: none;
        border-radius: 3px;
        cursor: pointer;
    }
    button:hover {
        background-color: #217dbb;
    }
}
</style>
</head>
<body>

```

```

    <div class="employee-form">
        <h2>직원 정보 등록</h2>

```

<!-- label 은 주로 값을 입력받는 곳에 이름을 적는 태그이며  
 for 를 통해 for 뒤에 나오는 값 과 id 이름 이 같은것을 찾아 연결시키는 역할을 한다.  
 -->

```

<form id="employeeForm" action = "empreg.jsp">
    <label for="empid">사원번호:</label>
    <input type="text" id="empid" name="empid" required>

    <label for="name">이름:</label>
    <input type="text" id="name" name="name" required>

    <label for="email">이메일:</label>
    <input type="text" id="email" name="email" required>

    <label for="contact">연락처:</label>
    <input type="text" id="contact" name="contact" required>

```

```

        <button type="submit">등록</button>
    </form>
</div>

<script src="js/jquery-3.7.1.min.js"></script>

<script>
    $(document).ready(function () {
        // 폼 제출 이벤트 처리
        $('#employeeForm').submit(function (event) {
            //event.preventDefault(); // 동작안되는코드

            // 입력된 값 가져오기
            var empid = $('#empid').val();
            var name = $('#name').val();
            var email = $('#email').val();
            var contact = $('#contact').val();

            // 간단하게 콘솔에 출력 (실제로는 서버에 전송하거나 다른 작업 수행)
            console.log('사원번호:', empid);
            console.log('이름:', name);
            console.log('이메일:', email);
            console.log('연락처:', contact);
        });
    });
</script>
</body>
</html>

```

## 5. empreg.jsp 폴더 만들기 > 직원 정보 등록 처리

```

<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ page import = "java.sql.*" %>
<%@ page import = "utils.DBConfig" %>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>

    <%
        out.println("직원정보처리");
        String empid = request.getParameter("empid");
        String name = request.getParameter("name");
        String email = request.getParameter("email");
        String contact = request.getParameter("contact");
    %>

```

```

//오라클 DB와 접속
Connection connection = null;
PreparedStatement pstmt = null;

// 접속하기
try {
    //connection = DriverManager.getConnection(JDBC_URL, USER, PASSWORD);
    connection = DBConfig.getConnection();
    out.println("jsp계정 접속성공");
} catch (Exception e) {
    out.println("jsp계정 접속실패");
}

//폼 정보에서 받은 변수들 empid, name, email, contact 값
try { // sql문을 실행할 때 발생할 수 있는 예외들을 처리하기 위한 구문
    String sql =
        "INSERT INTO EMPLOYEES(EMPLOYEE_ID, LAST_NAME, EMAIL, PHONE_NUMBER)
VALUES(?,?,?,?)";
    pstmt = connection.prepareStatement(sql); //insert, update, delete의 경우에만 적는다,
    일반적으로 prepareStatement사용
    pstmt.setString(1, empid); //1,2,3,4 물음표 번호순서
    pstmt.setString(2, name);
    pstmt.setString(3, email);
    pstmt.setString(4, contact);

    int result = pstmt.executeUpdate();
    out.println("데이터입력결과" + result);

    //성공되면 직원목록화면 emplist.jsp 성공하면 값이 1이 나온다
    if(result == 1){
        out.println("<script>alert('직원등록성공!!')</script>");
        out.println("<script>location.href='</script>");
    } else { //실패하면 다시 직원등록화면
    }
} catch (SQLException se) {

}

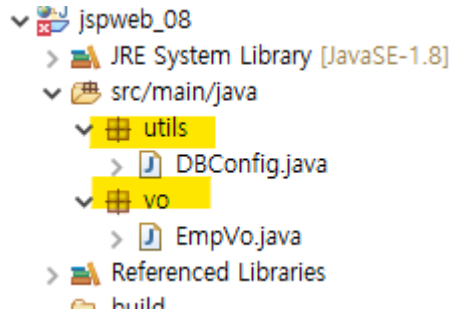
%>

</body>
</html>

```



## 6. EmpVo.java를 vo 패키지 안에 만들기



## 7. emplist.jsp 직원목록 가져오는 파일

화면설계

디비접속

ResultSet rs = excuteQuery: select

int result = excuteUpdate: i, u, d

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

```
<!-- DB 접속! -->
```

```
<%@ page import = "java.sql.*" %>
```

```
<%@ page import = "utils.DBConfig" %>
```

```
<%@ page import = "java.util.*" %>
```

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

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<meta charset="UTF-8">
```

```
<title>Insert title here</title>
```

```
<style>
```

```
/* 직원 목록 */
```

```
body {
```

```
    font-family: Arial, sans-serif;
```

```
    background-color: #f2f2f2;
```

```
    margin: 0;
```

```
    padding: 0;
```

```
    display: flex;
```

```
    justify-content: center;
```

```
    align-items: center;
```

```
    height: 100vh;
```

```
}
```

```
.employee-list {
```

```
    background-color: #fff;
```

```
    padding: 20px;
```

```
    border-radius: 5px;
```

```
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
```

```
}
```

```
table {
```

```
    width: 100%;
```

```

        border-collapse: collapse;
        margin-top: 20px;
    }

    th, td {
        border: 1px solid #ddd;
        padding: 10px;
        text-align: left;
    }

    th {
        background-color: #3498db;
        color: #fff;
    }
</style>
</head>
<body>

    <%      //날는이유? 가짜 웹을 디비랑 연동시켜서 값을 넣어준다
            //오라클 DB와 접속
            Connection connection = null; //

            // 접속하기
            try {
                //connection = DriverManager.getConnection(JDBC_URL, USER,
PASSWORD);???
                connection = DBConfig.getConnection();
                System.out.println("jsp계정 접속성공");
            } catch (Exception e) {
                System.out.println("jsp계정 접속실패");
            }

            //직원 정보 목록을 가져오는 쿼리 실행
            PreparedStatement pstmt = null; //쿼리를 실행하기 위한 객체 > sql
            ResultSet resultSet = null; // 디비에서 select 된 결과를 저장하는 배열 - 여기서
나온데이터가 배열로 나온다

            ArrayList<EmpVo> emplist = new ArrayList<>(); //import java.util.*
            // <>안에 들어가는 타입이 String이 아니라 EmpVo
            //int count = 0;
            try{
                String sql = "SELECT EMPLOYEE_ID, LAST_NAME, JOB_ID,
PHONE_NUMBER FROM EMPLOYEES";
                pstmt = connection.prepareStatement(sql); // ? 물음표가 없는경우에
statement 쓰는게 좋다 > pstmt = connection.createStatement();
                resultSet = pstmt.executeQuery(); // 오라클 디비 결과값
                while(resultSet.next()){ // .next()인덱스 행을 다음 차례로 바꿔준다
                    EmpVo empvo = new EmpVo(); // @ page import = "vo.*"
                    empvo.setEmployee_id(resultSet.getString("EMPLOYEE_ID"));
                    empvo.setLast_name(resultSet.getString("LAST_NAME"));
                    empvo.setJob_id(resultSet.getString("JOB_ID"));
                    empvo.setPhone_number(resultSet.getString("PHONE_NUMBER"));
                }
            }

```

```

        emplist.add(empvo);
        //(안넣어도됨) 각각에 대한 정보를 배열로 만들어 넣는것
        //count++;
    }
    System.out.println("총갯수: " + emplist.size()); //System.out.println 콘솔에
    찍기 out.println웹에 나오기
    }catch(SQLException se){

    }

    %>
<div class="employee-list">
    <h2>직원 목록</h2>
    <button onclick="handleAddButtonClick()">직원 등록</button> <!-- 등록 버튼 추가 -->
    <table>
        <thead>
            <tr>
                <th>사원번호</th>
                <th>이름</th>
                <th>직급</th>
                <th>연락처</th>
                <th>동작</th> <!-- 새로운 열 추가-->

            </tr>
        </thead>
        <tbody>
            <%
            for(int i=0; i<emplist.size(); i++){ //왜 size 써주지?
                EmpVo temp = emplist.get(i);

            %>
            <tr>
                <td><%= temp.getEmployee_id() %></td>          <!-- 1000 -->
                <td><%= temp.getLast_name() %></td>          <!--John Doe -->
                <td><%= temp.getJob_id() %></td>              <!-- 매니저 -->
                <td><%= temp.getPhone_number() %></td>        <!-- 010-1234-5678 -->
                <td><button onclick="handleButtonClick(0)">수정 처리</button> <!-- 버튼 추가 -->
                <button onclick="handleButtonClick(0)">삭제 처리</button><!-- 버튼 추가 -->

                </td> <!-- 상세보기 버튼 추가 ???-->
            </tr>

            <%
            }
            %>
            <!-- 다른 직원들의 정보도 추가할 수 있습니다->
        </tbody>
    </table>
</div>

<script>
function handleAddButtonClick() {
    location.href="empform.jsp";

```

```
}  
</script>
```

```
</body>  
</html>
```

## empform.jsp 질문 Q&A

1. **empform.jsp** 파일에서 **label**에서 사용한 **for** 를 통해 **for** 뒤에 나오는 값 과 **id** 이름 이 같은것을 찾아 연결시키는 역할을 한다.?

→ form에 대한 설명을 label 태그에 넣고 form 과 label의 연결고리는 for을 사용한다.  
for은 id와 같은 값을 넣어주면 같이간다

2. **\$('#employeeForm').submit(function (event) {** 왜 버튼 클릭시 **.click**을넣지 않을까?

→ .click 일반적인 모든버튼에 해당한다

html에 써준 **<button type="submit">등록</button>** 구절은 데이터를 담아서 액션으로 넘겨주는 특수한 submit 버튼이다.

이를 활용하기 위해서는 제이쿼리 function에 .click이 아닌 .submit를 넣어서 사용한다.

3. **deppreg.jsp** 파일에서 **request.getParameter**의 의미

→ .getParameter는 웹에서 전송받은 request값을 읽어오는, 말그대로 html의 데이터를 추출하는데에만 사용됩니다.  
getParameter의 타입은 String다.

you need to parse the parameter as an int. The parameters are just "String" so you need to interpret it on the server as the type you expect.

ex) `studentId = Integer.parseInt(request.getParameter("StudentId"));`

## preventDefault() - JS 코드에 사용됨

```
$('#employeeForm').submit(function (event) {  
    event.preventDefault();  
})
```

- a 태그나 submit 태그는 누르게 되면 href 를 통해 이동하거나 , 창이 새로고침하여 실행되는 것을 막아준다.
- 주로 사용되는 경우

1. **a** 태그를 눌렀을때도 **href** 링크로 이동하지 않게 할 경우
2. **form** 안에 **submit** 역할을 하는 버튼을 눌렀어도 새로 실행하지 않게 하고싶을 경우 (**submit**은 작동됨)

- 해당 함수가 사용되지 않을 경우 해당 창에서 정답을 맞추는 퀴즈 같은 경우 정답여부가 보였다가 사라지게 되는데,
- 해당 함수를 사용할 경우 출력되어 남아있는 모습을 보이며 **submit**과 동시에 새로 이동되는것을 막아주기 때문에
- 페이지에 남아 정답여부를 보여주는 것이라면 해당 함수를 사용한다.

### emplist.jsp 질문Q&A

1. **//connection = DriverManager.getConnection(JDBC\_URL, USER, PASSWORD);**
2. **<th>동작</th> <!-- 새로운 열 추가, 수정과 삭제처리 onclick="handleButtonClick(0) 0**왜 있지?  
→ 임시로 0을 넣은거지만 **Employee\_id()** 값을 넣어 수정처리할 수 있다
3. **<tbody>** 상단 다음 테이블 내용/ 상단제목 **thead**
4. **for(int i=0; i<emplist.size(); i++){ //왜 size 써주지? 640**쪽 이것이 자바다 확인  
ArrayList에서는 **size** - 동적배열 사이즈가  
정적배열에서는 **.length** - 사이즈가 정해져있다
5. 세개 모두 같이 붙어서 쓰거나 따로 써 줄수 있다.  
<https://wonderful1003.tistory.com/2>

Connection connection = null;      DBConfig 파일과 연결  
PreparedStatement pstmt = **null**;;      쿼리를 실행하기 위한 객체 > sql  
ResultSet resultSet = **null**; // 디비에서 **select** 된 결과를 저장하는 배열 - 여기서 나온데이터가 배열로 나온다

<https://api.jquery.com/>

이제 부서파일도 만들어보자

1월 15일 월요일 - 10장 게시판 구현



index.jsp

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3 <!DOCTYPE html>
4 <html>
5 <head>
6   <meta charset="UTF-8">
7   <title>Insert title here</title>
8 </head>
9 <body>
10
11   <h2>기업 정보 시스템 예제</h2>
12
13   <ul>
14     <li><a href = "empform.jsp">직원정보등록화면</a></li>
15     <li><a href = "depform.jsp">직원정보등록화면</a></li>
16   </ul>
17 </body>
18 </html>
```

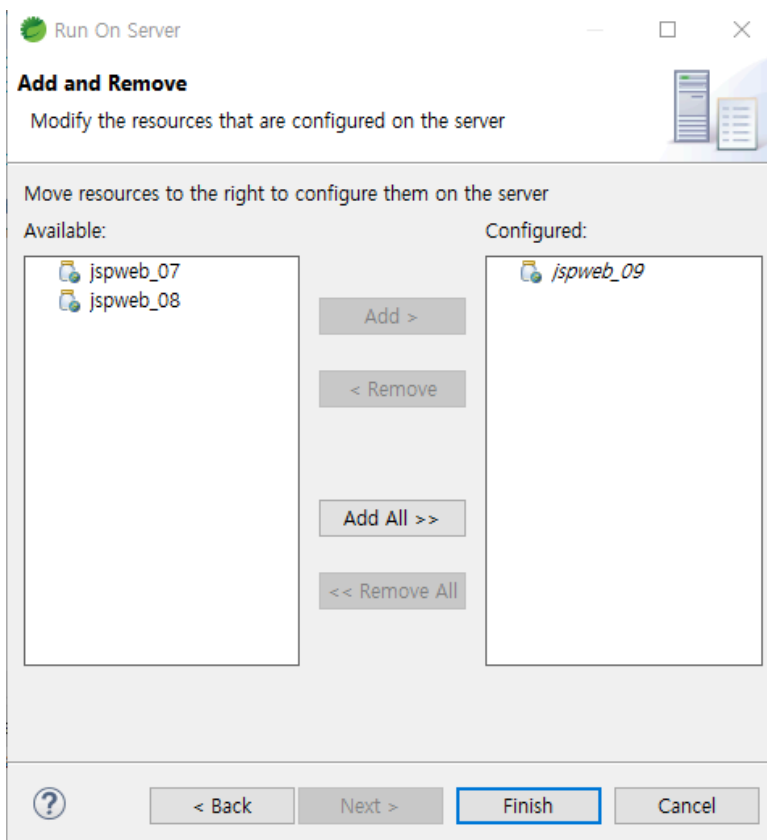
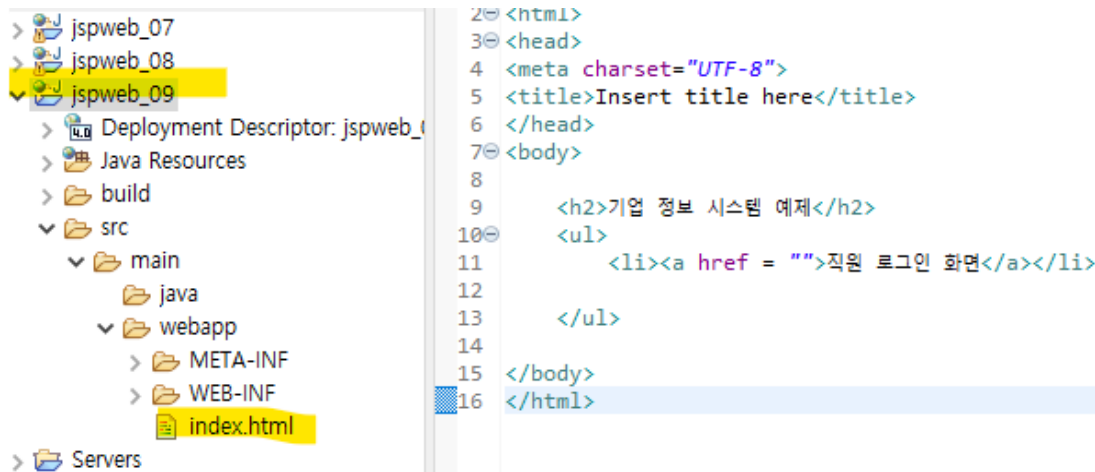
오라클에서 입력

직원 테이블에는 비밀번호 컬럼이 없기 때문에 사원 테이블로 수정을 해줘야 한다.

```
ALTER TABLE EMPLOYEES ADD USER_PW VARCHAR2(50) DEFAULT '1234';
DESC EMPLOYEES;
```

**STS** 에서 **jspweb\_09** 폴더 **dynamic web project** 만들기

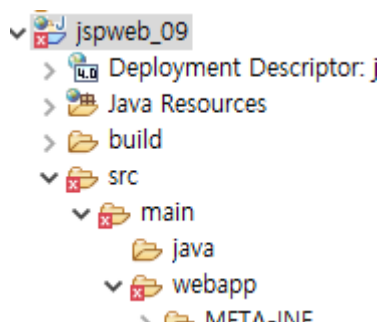
- dynamics web module 을 4.0으로 사용( tomcat 9 연동을위해서)
- index.html 만들기



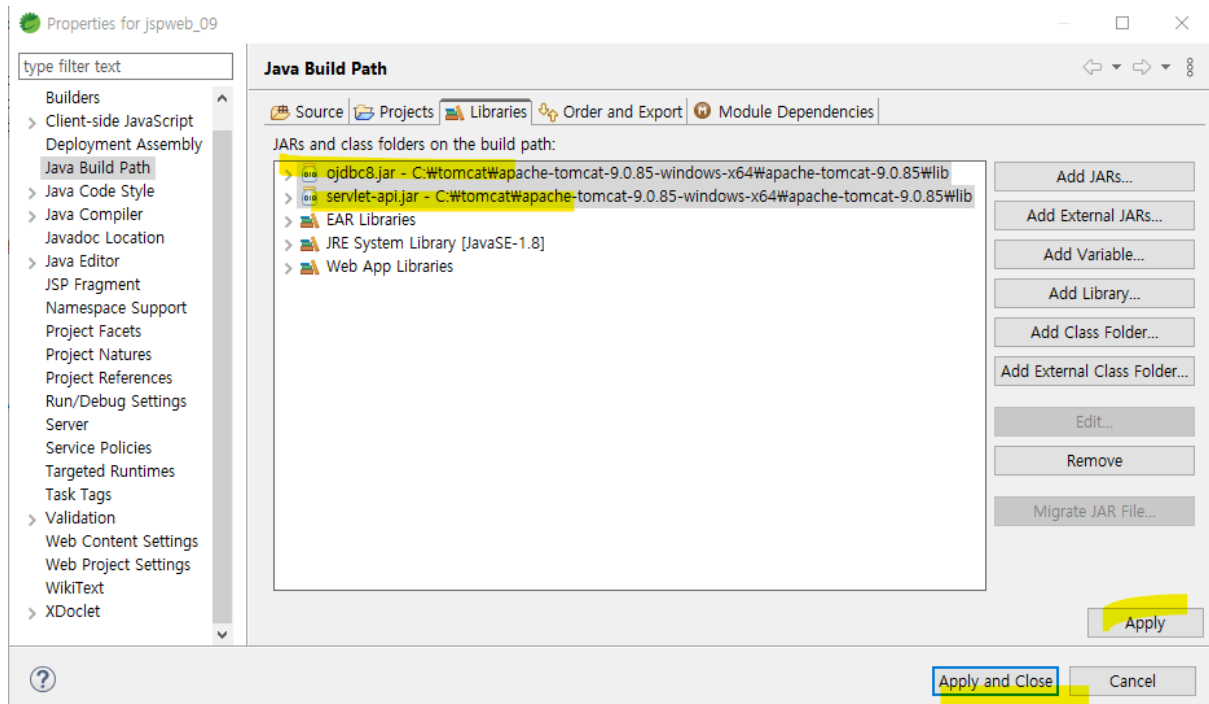
09만 실행해준다

## 기업 정보 시스템 예제

- 직원 로그인 화면



이렇게 오류가 날땐,  
build path로 라이브러리를 넣어준다



다음 `<%@ page import="java.util.*" %>` 넣다 빼주기

**loginform.jsp** 아래 입력

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

```
<%@ page import="java.sql.*" %>
```

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<meta charset="UTF-8">
```

```
<title>Insert title here</title>
```

```
<style>
```

```
    body {
```

```
        font-family: Arial, sans-serif;
```

```
        background-color: #f4f4f4;
```

```
        margin: 0;
```

```
        padding: 0;
```

```
        display: flex;
```

```
        align-items: center;
```

```
        justify-content: center;
```

```
        height: 100vh;
```

```
    }
```

```
    .login-container {
```

```
        background-color: #fff;
```

```
        padding: 20px;
```



```

        border-radius: 8px;
        box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
        width: 300px;
        text-align: center;
    }
    .login-container h2 {
        margin-bottom: 20px;
    }
    .login-form input {
        width: 100%;
        padding: 10px;
        margin-bottom: 15px;
        box-sizing: border-box;
    }
    .login-form button {
        background-color: #4caf50;
        color: #fff;
        padding: 10px;
        border: none;
        border-radius: 4px;
        cursor: pointer;
        width: 100%;
    }
    .login-form button:hover {
        background-color: #45a049;
    }
</style>
</head>
<body>
    <div class="login-container">
        <h2>Login</h2>
        <form class="login-form" action = "loginpro2.jsp">
            <input type="text" id="employeeId" name="employeeId" placeholder="사원번호" required> <!--
required 필수입력사항 -->
            <input type="password" id="password" name="userPw" placeholder="비밀번호" required>
            <button type="submit">로그인</button> <!-- onclick="login()"필요없다 이미 submit로 해결 -->
        </form>
    </div>
</body>
</html>

```

**Q. label, for** 붙이고 안붙이고 **empform.jsp** 와 **loginform.jsp** 차이는 보이는 디자인의 차이다

### 직원 정보 등록

사원번호:

이름:

이메일:

연락처:

등록

### Login

사원번호

비밀번호

로그인

로그인 인증처리

- 로그인 버튼 > **loginpro.jsp**로 입력값을 전달
- **loginpro.jst**.에서 오라클과 접속확인
- **jsp**(자바코드)에서 **DBConfig**(자바)를 호출하려면 **import**
- 아이디와 비번 존재하는지 체크
- 인증성공시 보여 줄 페이지로 이동 > **main.jsp**

- ▼ jspweb\_09
  - > Referenced Libraries
  - > Deployment Descriptor: jspwe
  - ▼ Java Resources
    - ▼ src/main/java
      - ▼ utils
        - > DBConfig.java
      - ▼ vo
        - > BoardVo.java
      - > Libraries
    - > build
    - ▼ src
      - ▼ main
        - > java
        - ▼ webapp
          - > META-INF
          - > WEB-INF
            - bbsread.jsp
            - board.jsp
            - header.jsp
            - index.html
            - loginform.jsp
            - loginpro.jsp
            - loginpro2.jsp
            - main.jsp
            - writeform.jsp
            - writepro.jsp
    - > Servers

워드파일로 확인하기.