JSP 4-2

JSP 4-1

🌐 커넥션 풀(DBCP), EL, JSTL

🔄 커넥션 풀 (Connection Pool, DBCP)

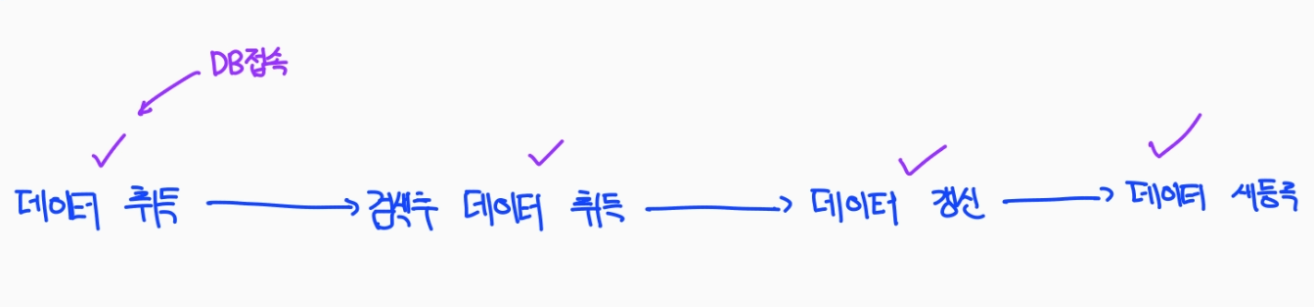
> 자원을 효율적으로 사용하기 위해 DB 연결을 미리 생성해두고 필요 시 꺼내 쓰는 방식

🧩 개념

- 웹 서버가 DB에 매번 연결하면 과부하 발생
- **미리 Connection 객체를 생성** → 필요 시 사용하고 반납
- 톰캣 컨테이너에서 관리

✅ 사용 이유

- 웹사이트 사용자가 많아질수록 DB 접속 수 증가 → 성능 저하
- 커넥션 풀 사용 시 접속 비용 감소, 성능 향상



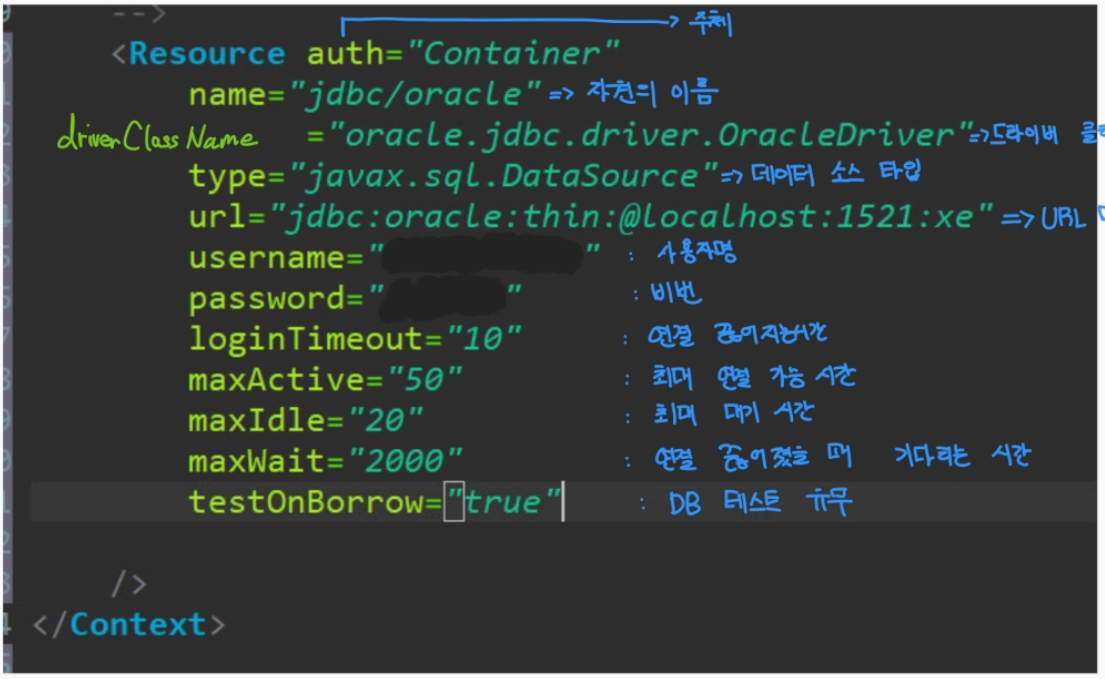
🔧 설정 예시

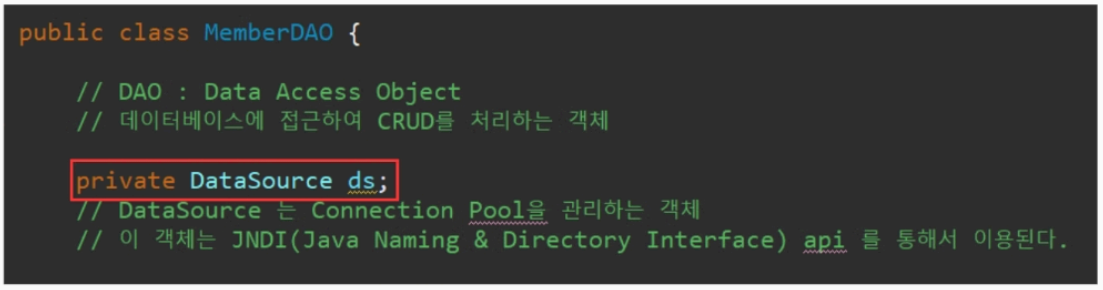
- context.xml에 Resource 등록
- web.xml에서 JNDI 이름 설정
- DAO 클래스에서 DataSource를 통해 Connection 획득

작성

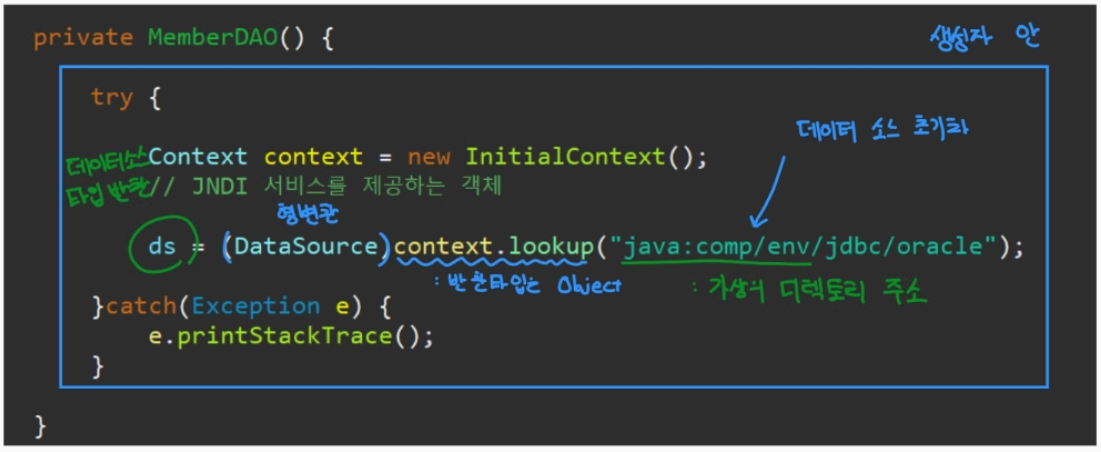
Servers → context.xml → 작성 →

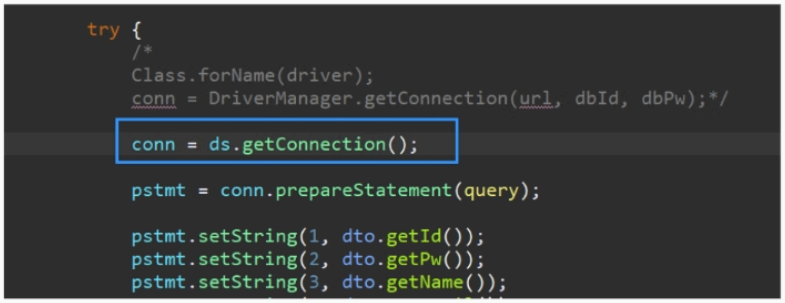
```
    -->
    <Resource auth="Container"
        name="jdbc/oracle"
driverClassName="oracle.jdbc.driver.OracleDriver"
        type="javax.sql.DataSource"
        url="jdbc:oracle:thin:@localhost:1521:xe"
        username="[redacted]"
        password="[redacted]"
        loginTimeout="10"
        maxActive="50"
        maxIdle="20"
        maxWait="2000"
        testOnBorrow="true"

    />
</Context>
```





: 이름 가지고 데이터베이스 정보 얻는다





💬 EL (Expression Language)

> JSP에서 Java 코드를 작성하지 않고 **데이터 출력 / 변수 접근**을 간단히 처리하는 언어

사용 예시

```
${user.name}         // request.getAttribute("user").getName()
${param.id}          // request.getParameter("id")
${sessionScope.user} // 세션의 user 객체
```

EL 표현식 => ${ }

변수, 메소드 호출, 연산 등의 표현식.

JSP 실행될 때 즉시 반영

객체 프로퍼티 값 가져올 때 주로 사용

EL 내장객체

- pageScope
- requestScope
- sessionScope
- applicationScope
- param

```
body>

    <%--
        서블릿 보관소에 저장된 데이터를 사용할 수 있다.
     --%>

    <%
        application.setAttribute("appliName", "appliValue");
        session.setAttribute("sessionName", "sessionValue");
        request.setAttribute("requestName", "requestValue");
        pageContext.setAttribute("pageName", "pageValue");
    %>

    어플리케이션 : ${applicationScope.appliName }<br>
    세션 : ${sessionScope.sessionName }<br>
    리퀘스트 : ${requestScope.requestName }<br>
    페이지 : ${pageScope.pageName }<br>

    <hr>

    어플리케이션 : ${appliName }<br>
    세션 : ${sessionName }<br>
    리퀘스트 : ${requestName }<br>
    페이지 : ${pageName }<br>
```

: 값 얻을 때

=

```
<body>

    <form action="el03.jsp" method="post">

        아이디 : <input type="text" name="id"><br>
        비밀번호 : <input type="text" name="pw"><br>

        <input type="submit" value="로그인">

    </form>

    <%--
        EL 내장객체의 param 객체로 넘어온 파라미터 값을 사용할 수 있다.
     --%>

    아이디 : ${param.id}<br>
    비밀번호 : ${param.pw }<br>
```

아이디 :
비밀번호 :
로그인
아이디 :
비밀번호 :

아이디 : dddd
비밀번호 : aaaa
로그인
아이디 :
비밀번호 :

아이디 :
비밀번호 :
로그인
아이디 : dddd
비밀번호 : aaaa

```jsp
<body>

    <%--
    EL(Expression Language)은
    정수, 실수, Boolean, 문자열, null 을 표현할 수 있다.
    ${null}은 빈 문자열("")로 출력된다.
     --%>

    정수 : ${100}<br>
    실수 : ${3.14}<br>
    논리 : ${true }<br>
    문자열 : ${'Hello World' }<br>
    널 : ${null }<br>

    <hr>
    <%--
        EL에서 사용하는 연산자는
        산술, 비교, 논리, 조건(삼항) 연산자가 있다.
     --%>

    <h2>연산자</h2>

    ${1 + 2}<br>
    ${1 < 2 }<br>
    ${(1 < 2) && (2 < 1) }<br>
    ${(1 < 2) || (2 < 1) }<br>
    ${(1 < 2) ? '참 입니다.' : '거짓 입니다.' }<br>
```

```java
package com.el;

public class ELMember {

    private String id;
    private String pw;
    private String name;
    private int age;

    public ELMember() {

    }

    public String getId() {
        return id;
    }

    public void setId(String id) {
        this.id = id;
    }

    public String getPw() {
        return pw;
    }

    public void setPw(String pw) {
        this.pw = pw;
    }

    public String getName() {
```

```jsp
<%--
    EL은 자바빈의 프로퍼티를 jsp 표현식이나
    액션태그를 사용하는 것보다 쉽고 간결하게 사용할 수 있다.
 --%>

<jsp:useBean id="member" class="com.el.ELMember" />

<jsp:setProperty name="member" property="id" value="hong" />
<jsp:setProperty name="member" property="pw" value="1234" />
<jsp:setProperty name="member" property="name" value="홍길동" />
<jsp:setProperty name="member" property="age" value="20" />

아이디 : <jsp:getProperty name="member" property="id" /><br>
비밀번호 : <jsp:getProperty name="member" property="pw" /><br>
이름 : <jsp:getProperty name="member" property="name" /><br>
나이 : <jsp:getProperty name="member" property="age" /><br>

<hr>

아이디 : ${member.id }<br>
비밀번호 : ${member.pw }<br>
이름 : ${member.name }<br>
나이 : ${member.age }<br>
```

🧱 JSTL (JSP Standard Tag Library)

> JSP의 가독성 문제 보완을 위해 **표준 태그 라이브러리** 제공

🔧 특징

- Java 코드 없이 반복/조건문 구현 가능
- 코드 재사용성 향상
- 스크립틀릿 없이 구조화된 코드 작성 가능

🛠️ 사용을 위한 설정

- JSTL 라이브러리 JAR 파일 설치 필요 (톰캣 기본 미포함)
- `<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>` 선언

예시 태그

```
<c:if test="${score >= 90}">우수</c:if>
<c:forEach var="item" items="${list}">${item}</c:forEach>
```

mvnrepository.com

mvnrepository.com

2. JSTL

javax.servlet » jstl

JSTL

Last Release on Jun 23, 2011

Relocated → javax.servlet.jsp.jstl » jstl

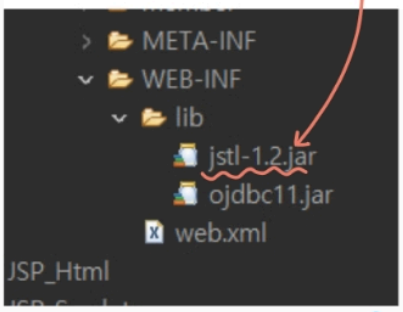
2,183 usages

GPL CDDL

Central (11) | Less is More (1) | EmergyaPub (1)

Version	Vulnerabilities	Repository	Usages	Date
1.2.x 1.2		Central	1,660	Jun 23, 2011

License	CDDL GPL 2.0
Categories	Java Specifications
Tags	standard servlet javax jstl specs
Date	Jun 23, 2011
Files	pom (356 bytes) jar (404 KB) View All
Repositories	Central GlareMasters Pub Gradle Plugins Less is More OneBusAway Pub Softmotions WSO2 Dist
Ranking	#247 in MvnRepository (See Top Artifacts) #17 in Java Specifications
Used By	2,183 artifacts



core

Ctrl + space 누르고 '선택'

```
3
4 <%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
5 <%--
6     자바에서 import 문을 선언하듯 jsp에서 jstl 확장태그를 사용하려면
7     taglib 지시자로 라이브러리를 선언해야 한다.
8  --%>
9 <!DOCTYPE html>
```

```jsp
<body>

    <%--
    일반적으로 JSTL이란 JSTL + EL의 조합을 말한다.
    HTML 코드 내에 java 코드인 스크립틀릿 코드를 대체하여 사용한다.
    스크립틀릿을 사용하면 가독성이 떨어지고, 뷰와 비즈니스로직의 분리로 인해
    현재는 JSTL을 많이 사용하는 추세이다.
     --%>

    <%--
        set : 변수 선언
        out : 변수값 출력
        remove : 변수값 제거

        var : 변수의 이름 지정
        value : 변수의 값 설정
        scope : 범위지정 (page, request, session, application)
     --%>
    변수선언   변수 이름   변수 값   범위지정
    <c:set var="name" value="홍길동" scope="page" />

    이름 : <c:out value="${name }" />
    <hr>
    출력
    삭제
    <c:remove var="name" scope="page"/>

    이름 : <c:out value="${name }" />

</body>
```

이름 : 홍길동

이름 :

```jsp
<%--
    if : if문 (else문 없음)
    test="조건식"
--%>

<c:if test="${1+2 == 3 }">
    1더하기 2는 3 입니다<br>
</c:if>

<hr>

<c:if test="${1+2 == 5 }">
    1더하기 2는 5 입니다<br>
</c:if>
```

조건식

false 면 넘어감 실행 X

```jsp
<%--
    choose 태그는 자바의 switch문과 비슷한 역할을 한다.
    case 와 같은 역할을 하는것은 choose 안의 when 태그이다.
    default 와 같은 역할을 하는 것은 choose 안의 otherwise 태그이다.
--%>

<c:set var="a" value="70" />

<c:choose>
    <c:when test="${a == 10 }">
        a는 10 입니다.<br>
    </c:when>

    <c:when test="${a == 20 }">
        a는 20 입니다.<br>
    </c:when>

    <c:when test="${a == 30 }">
        a는 30 입니다.<br>
    </c:when>

    <c:otherwise>
        a는 10, 20, 30이 아닙니다.<br>
    </c:otherwise>
</c:choose>
```

선언

조건

case 역할

default 와 같은 역할

```jsp
<%--
    forEach : for문
--%>

<c:forEach var="i" begin="0" end="30" step="3" >
    ${i}  
</c:forEach>

<hr>

<%
    ArrayList<String> list = new ArrayList<String>();
    list.add("홍길동");
    list.add("성춘향");
    list.add("이순신");

    request.setAttribute("members", list);
%>

<c:forEach var="member" items="${members}">
    ${member}  
</c:forEach>
```

변수설정 / 시작 / 끝 / 증가 값

members 이름으로 ArrayList 객체 넣어줌

```jsp
<%--
    items 속성에 Collection 객체를 넣어주면
    순서대로 반복이 된다.
--%>
```

localhost:8060/JSP_Database/el_jstl/jstl(

1더하기 2는 3 입니다

a는 10, 20, 30이 아닙니다.

0 3 6 9 12 15 18 21 24 27 30

홍길동 성춘향 이순신