JSP Day - 5

1. JSP 내장객체

JSP 페이지에서 명시적으로 객체를 생성하지 않고도 바로 사용할 수 있는 객체를 말한다. 자바는 객체를 참조하는 참조변수를 선언하고 그 참조변수에 객체를 생성해 참조값을 할당 해야 비로소 참조 변수를 통해서 그 객체에 접근할 수 있는데 JSP에서 사용하는 내장객체처럼 명시적으로 객체를 생성하지 않고 바로 사용할 수 있는 이유는 JSP 페이지가 톰캣에 의해서 서블릿 클래스로 변환될 때 JSP에서 사용하는 내장객체가 자동으로 생성되도록 코드가 만들어지기 때문이다.

1) request 내장객체 : 클라이언트에 대한 요청 정보를 저장하고 관리하는 객체가 바로 request 내장객체이다. request 객체는 HttpServletRequest 타입으로 클라이언트에서 서버로 전송되는 요청 정보에 접근할 수 있는 다양한 메서드를 제공하고 있다.

2) response 내장객체 : 클라이언트의 요청에 대한 응답 처리를 위해 필요한 객체가 바로 response 내장객체이다. response 객체는 요청에 대한 처리 결과를 클라이언트로 보낼 때 사용하는 객체이며 응답 데이터의 헤더 설정이나 리다이렉트 기능을 제공하는 메서드를 제공하고 있다.

3) out 내장객체 : 서버에서 클라이언트로 연결된 출력 스트림으로 클라이언트로 응답할 데이터를 출력하는 역할을 하는 내장객체이다. 서블릿 클래스에서는 PrintWriter 타입의 출력 스트림을 사용해 클라이언트로 응답할 데이터를 만들었지만 JSP 페이지에서는 JSPWriter 타입으로 out 내장객체가 선언되어 있다. PrintWriter 클래스나 JSPWriter 클래스는 Writer 클래스를 상속한 출력 스트림 이 다.

4) 웹 애플리케이션 정보

웹 애플리케이션 컨텍스트 패스 : <%= application.getContextPath() %>
웹 애플리케이션 이름 : <%= application.getServletContextName() %>
웹 애플리케이션 파일의 절대경로 : <%= application.getRealPath("applicationMethods.jsp") %>

- C:\Users\kmh\Desktop\MH\GITAS\JSPStudy\.metadata\.plugins\org.eclipse.wst.server.core\tmp0\work\Catalina\localhost

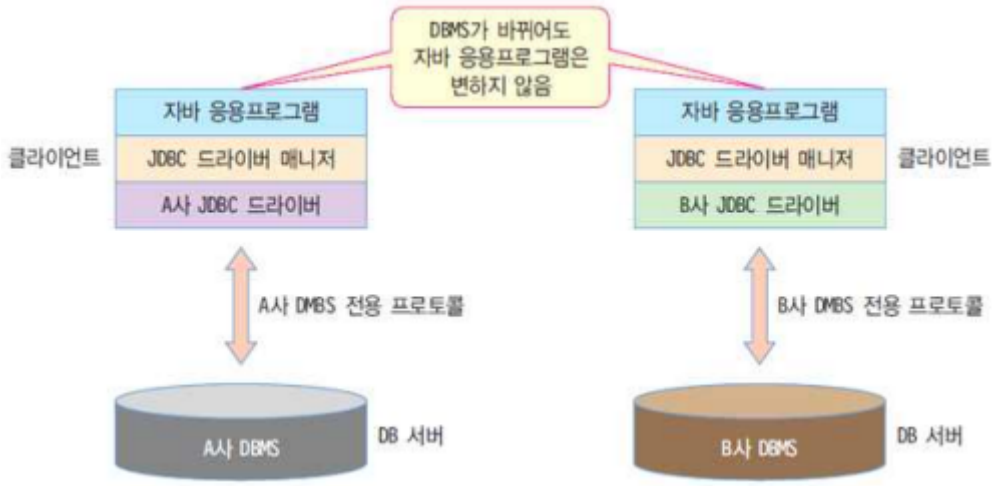
5) 데이터를 저장할 수 있는 4가지 내장객체

데이터를 저장할 수 있도록 속성을 제공하는 내장객체 : pageContext, request, session, application

- pageContext : 내장객체의 영역은 한 번의 요청을 처리하는 같은 JSP 페이지 안에서 유효하다.
- request : 내장객체는 한 번의 요청을 처리하는 과정 안에서 유효하다.
- session : 하나의 브라우저 접속(한 명의 사용자 접속을 의미하며 이를 세션이라고 한다.) 안에서 유효한데 주로 한 사용자(같은 세션)와 관련된 정보(로그인 정보, 장바구니 등등)를 여러 JSP 페이지(웹 컴포넌트)가 공유하기 위해 서 사용한다.
- application : 하나의 웹 애플리케이션 안에서 유효하며 웹 애플리케이션을 사용하는 모든 사용자와 관련해서 파일을 업로드 하는 서버내의 폴더 정보나 웹 애플리케이션의 설정 정보 등을 공유하기 위해서 사용한다.
    - 동일한 페이지에서만 공유되어 페이지를 벗어나면 소멸된다. → pageContext 사용
    - 하나의 요청에 의해 호출된 페이지와 포워드된 페이지까지 공유 → request 사용
    - 클라이언트가 처음 접속한 후 웹 브라우저를 닫을 때까지 공유된다. → session 사용
    - 한 번 저장되면 웹 애플리케이션이 종료될 때까지 유지된다. → application 사용

2. JSP 데이터베이스 프로그래밍

JDBC(Java Database Connectivity)는 자바 응용프로그램과 RDBMS(Relational Database Manager System) 간의 표준적인 연결 방법을 제공하기 위해 정의해 놓은 자바를 위한 데이터베이스 접속 인터페이스이다. 다시 말해 JDBC는 자바 응용프로그램에서 SQL 명령을 사용해 RDBMS(Relational Database Manager System)에 CRUD(Create, Read, Update, Delete) 작업을 위한 표준적인 프로그래밍 방식을 제공하는 라이브러리 이다. 이 JDBC 라이브러리는 java.sql 패키지에 정의되어 있으 며 RDBMS에 접근 할 수 있는 방법을 제공하는 단일 API(Application Programming Interface) 이다.



오라클의 SYS 에서 제한을 풀어둠

ALTER USER hr IDENTIFIED BY hr ACCOUNT UNLOCK;

WEB-INF > SQL > JSPStudyBBS.sql 을 HR에 복사

- Table, Sequence, Insert Into ~ 마다 ctrl + Enter 해서 자료 입력

BoardListController.java 실행 > 게시판에 자료 들어가는 것 확인

2.1 자바 응용프로그램에서 **JDBC** 프로그램을 작성하는 순서

1) 임포트 : import.java.sql.*

2) 각 DBMS에 맞는 JDBC 드라이버를 로드

JDBC를 이용해 데이터베이스를 연동하는 웹 애플리케이션을 작성하기 위해서는 DBMS에 맞는 접속 드라이버가 있어야 한다. 아래의 각 드라이버 파일을 다운 받은 후 WEB-INF > lib 에 추가하면 된다.

- 오라클 접속 드라이버
  - 다운 경로 : http://www.oracle.com/us/downloads/index.html
  - C:\oraclexe\app\oracle\product\11.2.0\server\jdbc\lib > ojdbc6.jar을 WEB-INF > lib
  - 추가로 impl, spec도 lib에 복사할 것
- MySQL 접속 드라이버
  - 다운 경로 : http://dev.mysql.com/downloads/connector/j/

JDBC 드라이버를 로드 하는 방법은 어떤 메소드를 사용하느냐에 따라서 몇 가지가 있을 수 있으나 우리는 클래스의 정보를 다루기 위해 자바에서 제공하는 Class 클래스의 forName() 메소드를 이용해 JDBC 드라이버를 동적으로 로드하여 사용할 것이다. 이 메소드의 인수로 접속할 DBMS에 맞는 Driver 클래스 이름을 문자열로 지정해야 한다. 드라이버 클래스의 이름을 문자열로 지정할 때는 Driver 클래스가 소속된 패키지를 포함하는 완전한 클래스 이름(Fully Qualified Name)을 기재해야 한다. 자바 응용프로그램이 실행되고 forName() 메소드가 호출되면 Driver 클래스를 동적으로 로드하고 드라이버 클래스의 인스턴스를 생성하여 DriverManager에 등록하게 된다. 다시 말해 Driver 클래스의 객체가 실행 시간에 메모리에 생성 되는 것이다. DBMS에 따라 JDBC 드라이버의 클래스 이름이 다르기 때문에 해당 DBMS의 JDBC 문서를 참조하여 적절한 드라이버 클래스 이름을 지정해야 한다. 만일 지정한 Driver 클래스가 존재하지 않는다면 ClassNotFoundException이 발생하므로 적절한 예외처리 또한 필요하다. 아래는 Oracle과 MySQL의 JDBC 드라이버를 로드하여 객체를 생성하고 DriverManager에 등록하는 코드이다.

Oracle DBMS : Class.forName(“oracle.jdbc.driver.OracleDriver”); MySQL DBMS : Class.forName(“com.mysql.jdbc.Driver”)

3) Connection 객체를 생성

DriverManager 클래스는 자바 응용프로그램을 JDBC 드라이버에 연결시켜 주는 역할을 하는 클래 스이다. DBMS 드라이버가 메모리에 생성되면 실제 DBMS가 설치된 접속 URL을 DriverManager 클래스의 getConnection() 메소드의 인수로 지정하여 호출 한다. getConnection() 메소드가 호출되면 DriverManager에 등록된 여러 DBMS의 Driver 중에서 2단계 에서 지정한 Driver를 식별하고 그와 관련된 DBMS와 연결된 Connection 객체를 반환 한다. getConnection() 메서드를 사용해 DBMS와 접속을 시도할 때 접속 URL의 정보나 사용자ID, 비밀 번호가 일치하지 않으면 SQLException이 발생하므로 적절한 예외 처리가 필요하다. 아래는 Oracle과 MySQL의 접속 URL과 사용자 ID, 비밀번호를 지정하는 예이다. 아래 예와 같이 접속 URL은 “jdbc:” 이후에 URL에 포함되는 형식이 DBMS 마다 다르기 때문에 해당 DBMS의 JDBC 문서를 참조하여 적절한 URL을 지정해야 한다.

- Oracle DBMS : DriverManager.getConnection(“jdbc:orcle:thin:@서버ip:1521:SID”, “사용자ID”, “비밀번호”);
- MySQL DBMS : DriverManager.getConnection(“jdbc:mysql://서버ip:3306/DB명”, “사용자ID”, “비밀번호”);

4) Statement 또는 PreparedStatement 객체를 생성

Statement 객체를 생성하고 DBMS에 요청할 SQL 쿼리를 생성하는 단계이다. - 68 - Statement 객체와 PreparedStatement 객체는 DBMS에 SQL 쿼리를 발행할 수 있도록 도와주는 객체로 DBMS에 연결된 Connection 객체로부터 아래와 같이 얻을 수 있다. 4단계부터 특정 DBMS에 구애받지 않고 동일한 코드로 DB 작업을 할 수 있다.

① Statement stmt = connection.createStatement();
② PreparedStatement stmt = connection.prepareStatement("SQL 쿼리문");

5) 쿼리로 질의한 결과를 ResultSet 객체로 얻어 데이터를 추출

SELECT 문의 경우는 데이터베이스에 질의하고 그 결과로 ResultSet 객체를 반환 받는다. 이 ResultSet 객체는 SELECT 질의에 대한 결과 집합으로 테이블 구조를 가지는 데이터이다. ResultSet 객체는 테이블의 행에 대한 참조를 조작하기 위해 커서(Cursor)를 가지고 있으며 이 커서를 이용해 첫 번째 행이나 마지막 행 또는 바로 이전행과 같이 각 행을 이동할 수도 있고 순차적 으로 이동하며 데이터를 읽어 올 수도 있다. SELECT 쿼리가 실행되고 데이터를 읽어오면 커서의 첫 번째 위치는 실제 데이터 행의 위치보다 작은 –1이 된다. 즉 맨 처음 ResultSet 객체를 얻어오면 커서는 첫 번째 행 바로 이전을 가리키고 있는 형태가 된다. 아래 코드는 ResultSet 객체의 메소드를 사용해 순차적으로 데이터를 읽어오는 예이다.

```
while(rs.next()) {
 int no = rs.getInt(1);
 String name = rs.getString("writer");
 String id = rs.getString(3);
}
```

6) DB 작업이 끝나면 작업한 객체를 생성한 역순으로 닫아 준다.

```
try { if(rs != null) rs.close();
 if(pstmt != null) pstmt.close();
 if(conn != null) conn.close();
 } catch(SQLException e) { e.printStackTrace(); }
```

//복습 및 정리

Eclipse 에서의 JDBC 프로그램 연결 작성
컴파일러의 진행 순서 : 이클립스->forName()->DriverManager

- 임포트
- JDBC 드라이버 설치
- forName() 메소드로 동적 연결
    - Class.forName("oracle.jdbc.driver.OracleDriver");
- Connection 객체 생성
    - conn = DriverManager.getConnection("jdbc:oracle:thin:@서버ip:1521:SID", "사용자ID", "비밀번호");
- Statement & PreparedStatement 객체 생성(DBMS에 SQL 쿼리 발행을 도와주는객체)
    - pstmt = conn.prepareStatement("SELECT * FROM jspbbs ORDER BY no DESC");
- 쿼리로 질의한 결과를 ResultSet 객체로 얻어 데이터를 추출
    - while(rs.next()) {String name = rs.getString("writer"), ;}
- 작업한 객체 역순으로 클로즈
    - try { if(rs != null) rs.close() }

2.2 데이터베이스에 접속해 게시 글 리스트 읽어오기

데이터베이스 커넥션 풀(Database Connection Pool) - 커넥션 풀에 사용되는 라이브러리 파일을 https://commons.apache.org/ 에서 다운

- https://commons.apache.org/proper/commons-dbcp/download_dbcp.cgi
- commons-dbcp2-2.11.0-bin.zip

- https://commons.apache.org/pool/download_pool.cgi
- commons-pool2-2.12.0-bin.zip

- https://commons.apache.org/logging/download_logging.cgi
- commons-logging-1.3.0-bin.zip