Spring Boot Application

스프링 부트 응용

JSP를 이용한 UI 화면 개발
(JSP 기본 문법 및 글목록 개발)

한국기술교육대학교
온라인평생교육원

학습내용

- JSP 기본 문법
- JSP를 이용한 게시글 목록 개발

학습목표

- 웹 애플리케이션 개발에 필요한 기본적인 JSP 문법에 대해 설명할 수 있다.
- JSP를 이용하여 게시글 목록 화면을 개발할 수 있다.

JSP 기본 문법

1 JSP(Java Server Pages) 개발 설정

〈1〉 라이브러리 추가

- 스프링 부트는 타임리프(Thymeleaf) 같은 템플릿 엔진을 기본으로 지원함

1 JSP(Java Server Pages) 개발 설정

〈1〉 라이브러리 추가

- 타임리프가 아닌 JSP를 적용하기 위해서는 JSP 관련 의존성을 pom.xml 파일에 추가해야 함

```
    <dependency>
        <groupId>org.apache.tomcat.embed</groupId>
        <artifactId>tomcat-embed-jasper</artifactId>
    </dependency>

    <dependency>
        <groupId>javax.servlet</groupId>
        <artifactId>jstl</artifactId>
    </dependency>

</dependencies>
```

JSP 기본 문법

1 JSP(Java Server Pages) 개발 설정

⟨1⟩ 라이브러리 추가

pom.xml 파일에 추가된 라이브러리에 대한 정보	
tomcat-embed-jasper	JSP 파일을 Servlet으로 컴파일할 때 필요한 라이브러리
jstl	JSP에 JSTL을 적용할 때 필요한 라이브러리

1 JSP(Java Server Pages) 개발 설정

⟨2⟩ ViewResolver 설정

- 스프링에서 제공하는 ViewResolver를 설정하면 브라우저로부터의 직접적인 JSP 호출을 차단하면서 동시에 JSP 파일의 경로를 효율적으로 관리할 수 있음

JSP 기본 문법

1 JSP(Java Server Pages) 개발 설정

〈2〉 ViewResolver 설정

- application.yml 파일에 ViewResolver 설정을 추가함

```
# ViewResolver 설정
 mvc:
    view:
      prefix: /WEB-INF/board/
      suffix: .jsp
```


1 JSP(Java Server Pages) 개발 설정

〈2〉 ViewResolver 설정

- application.yml 파일에 ViewResolver 설정을 추가함

mvc는 spring 하위에 위치해야 함

```
spring:
    mvc:
        view:
            prefix: /WEB-INF/board/
```

JSP 기본 문법

1 JSP(Java Server Pages) 개발 설정

〈2〉 ViewResolver 설정

ViewResolver가 실제 실행할 뷰 이름을 완성하는 과정

컨트롤러의 메소드가 리턴하는 뷰 이름 + 접두사(prefix)와 접미사(suffix)를 추가

예 뷰 이름이 getBoardList인 경우 :
/WEB-INF/board/getBoardList.jsp

1 JSP(Java Server Pages) 개발 설정

〈3〉 JSP 폴더 생성

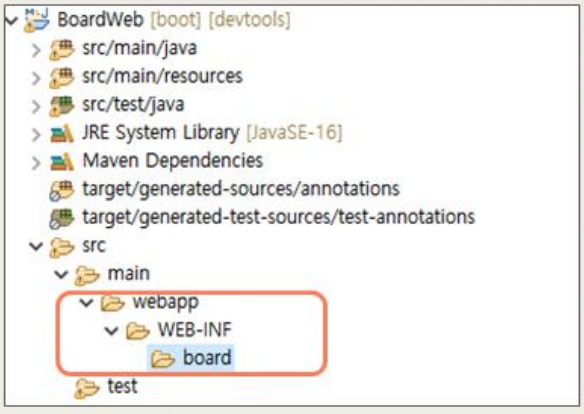
- JSP 파일을 등록할 물리적인 폴더 경로 (webapp/WEB-INF/board)를 생성함

JSP 기본 문법

1 JSP(Java Server Pages) 개발 설정

〈3〉 JSP 폴더 생성

- 웹 애플리케이션의 WEB-INF 폴더는 브라우저가 직접 접근할 수 없는 폴더



1 JSP(Java Server Pages) 개발 설정

〈4〉 Web Developer Tools 설치

- STS가 포함된 Eclipse에는 기본적으로 웹 개발과 관련된 도구들이 제공되지 않음

JSP 기본 문법

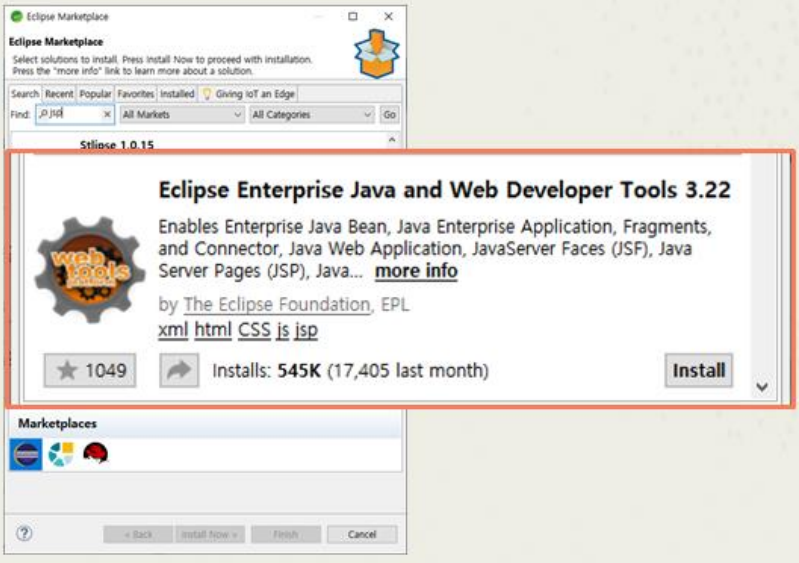
1 JSP(Java Server Pages) 개발 설정

4 Web Developer Tools 설치

- 웹 애플리케이션을 개발하기 위해서는 사용 중인 Eclipse에 Web Developer Tools 플러그인을 설치해야 함

1 JSP(Java Server Pages) 개발 설정

4 Web Developer Tools 설치



JSP 기본 문법

2 JSP 기본 문법

① 스크립트 기반 태그

1 Scriptlet

- Scriptlet은 사용자가 JSP 페이지를 요청할 때마다 수행되는 JAVA 코드가 작성됨

2 JSP 기본 문법

① 스크립트 기반 태그

1 Scriptlet

```
<%@page import="com.mycompany.entity.Member"%>
<%@page import="com.mycompany.service.MemberService"%>
<%@page contentType="text/html; charset=UTF-8"%>

<%
    // 1. 사용자 입력정보 추출
    String id = request.getParameter("id");
    String password = request.getParameter("password");

    // 2. 비즈니스 로직 처리
    Member member = new Member();
    member.setId(id);
    member.setPassword(password);

    MemberService memberService = new MemberService();
    Member credentials = memberService.getMember(member);
%>
```

Scriptlet으로 감싸지 않은 JAVA 코드는 단순한 텍스트로 처리됨

JSP 기본 문법

2 JSP 기본 문법

⟨1⟩ 스크립트 기반 태그

2 Expression

- 주로 Scriptlet에 의해 처리된 동적 데이터를 응답 화면에 포함시키기 위해 사용됨

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
                      "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Insert title here</title>
</head>
<body>
<center>
<h1><%= credentials.getName() %>님 환영합니다.</h1>
</center>
</body>
</html>
```

2 JSP 기본 문법

⟨2⟩ HTTP 요청과 응답 처리

1 HTTP 요청 처리

- 브라우저로부터 전송된 HTTP 요청을 처리하기 위해서는 HttpServletRequest 객체를 사용해야 함

JSP 기본 문법

2 JSP 기본 문법

〈2〉 HTTP 요청과 응답 처리

1 HTTP 요청 처리

- HttpServletRequest 객체가 제공하는 주요 메소드

메소드	기능
String getParameter(String paramName)	사용자가 입력한 파라미터 값을 추출함
void setCharacterEncoding(String enc)	사용자 입력 값에 대한 인코딩을 설정함
HttpSession getSession()	브라우저와 매핑된 세션 객체를 생성하거나 리턴함

2 JSP 기본 문법

〈2〉 HTTP 요청과 응답 처리

2 HTTP 응답 처리

- 서버가 브라우저에 HTTP 응답을 전송하기 위해서는 HttpServletResponse 객체를 사용해야 함

JSP 기본 문법

2 JSP 기본 문법

〈2〉 HTTP 요청과 응답 처리

2 HTTP 응답 처리

- HttpServletResponse 객체가 제공하는 주요 메소드

메소드	기능
void sendRedirect(String url)	응답을 받은 브라우저가 어느 페이지로 Redirect(재요청)할지 지정함

JSP를 이용한 게시글 목록 개발

1 비즈니스 컴포넌트 작성

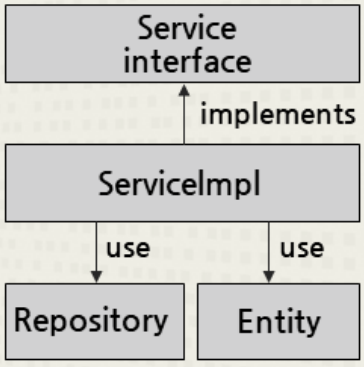
1 비즈니스 컴포넌트 구조

- 비즈니스 컴포넌트는 Entity, Repository, Service, ServiceImpl 클래스로 구성됨

1 비즈니스 컴포넌트 작성

1 비즈니스 컴포넌트 구조

비즈니스 컴포넌트의 각 구성요소	
Entity	테이블의 Row와 매핑할 엔티티 클래스
Repository	엔티티를 기반으로 실질적인 CRUD 기능을 제공하는 인터페이스
Service Interface	비즈니스 컴포넌트 인터페이스(Option)
ServiceImpl	비즈니스 로직 처리를 담당하는 비즈니스 클래스



JSP를 이용한 게시글 목록 개발

1 비즈니스 컴포넌트 작성

〈2〉 Entity 클래스

- JPA가 제공하는 어노테이션을 이용하여 BOARD 테이블과 매핑할 Board 엔티티 클래스를 매핑함

1 비즈니스 컴포넌트 작성

〈2〉 Entity 클래스

```
package com.mycompany.entity;

import java.util.Date;

@Data
@Entity
public class Board {
    @Id
    @GeneratedValue
    private int seq;
    private String title;
    @Column(updatable = false)
    private String writer;
    private String content;
    @Column(updatable = false)
    private Date createDate = new Date();
    @Column(updatable = false)
    private int cnt = 0;
}
```

JSP를 이용한 게시글 목록 개발

1 비즈니스 컴포넌트 작성

〈3〉 Repository 인터페이스

- BOARD 테이블에 대한 단순 CRUD 기능만 제공할 것이므로 CrudRepository 인터페이스를 상속하여 작성함

```
@Repository
public interface BoardRepository extends CrudRepository<Board, Integer> {
}
```

1 비즈니스 컴포넌트 작성

〈4〉 비즈니스 클래스

- 유지 보수 과정에서 비즈니스 클래스를 교체하지 않고 직접 수정할 것이므로 별도의 비즈니스 인터페이스는 작성하지 않음

JSP를 이용한 게시글 목록 개발

1 비즈니스 컴포넌트 작성

④ 비즈니스 클래스

- 스프링 부트는 내부적으로 AOP(Aspect Oriented Programming)를 이용하여 비즈니스 메소드에 대한 트랜잭션을 관리함

```
package com.mycompany.service;

import java.util.List;

@Service("boardService")
public class BoardService {
    @Autowired
    private BoardRepository boardRepositoy;

    public List<Board> getBoardList() {
        return (List<Board>) boardRepositoy.findAll();
    }
}
```

2 게시글 목록 화면 개발

① Controller 클래스

- BoardController 클래스에 사용자의 글목록 검색 요청을 처리할 getBoardList 메소드를 작성함

JSP를 이용한 게시글 목록 개발

2 게시글 목록 화면 개발

① Controller 클래스

```
package com.mycompany.controller;

import org.springframework.beans.factory.annotation.Autowired;

@Controller
public class BoardController {
    @Autowired
    private BoardService boardService;

    @RequestMapping("/board/getBoardList")
    public String getBoardList(Model model) {
        model.addAttribute("boardList", boardService.getBoardList());
        return "getBoardList";
    }
}
```

Controller의 메소드가 "getBoardList" 문자열을 리턴하면 ViewResoler가 동작함

2 게시글 목록 화면 개발

② 글목록 JSP 작성

- webapp/WEB-INF/board 폴더에 getBoardList.jsp 파일을 작성함

```
<%@page import="com.mycompany.entity.Board"%>
<%@page import="java.util.List"%>
<%@page contentType="text/html; charset=UTF-8"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
 "http://www.w3.org/TR/html4/loose.dtd">

<%
    // Controller가 Model에 등록한 검색 결과를 꺼낸다.
    List<Board> boardList = (List) request.getAttribute("boardList");
%>
```

JSP를 이용한 게시글 목록 개발

2 게시글 목록 화면 개발

⟨2⟩ 글목록 JSP 작성

```
<body>
<center>
<h1>게시글 목록</h1>
<table border="1" cellpadding="0" cellspacing="0" width="700">
<tr>
<th bgcolor="orange" width="100">번호</th>
<th bgcolor="orange" width="200">제목</th>
<th bgcolor="orange" width="150">작성자</th>
<th bgcolor="orange" width="150">등록일</th>
<th bgcolor="orange" width="100">조회수</th>
</tr>
<% for(Board board : boardList) { %>
<tr>
<td><%= board.getSeq() %></td>
<td align="left"><%= board.getTitle() %></td>
<td><%= board.getWriter() %></td>
<td><%= board.getCreateDate() %></td>
<td><%= board.getCnt() %></td>
</tr>
<% } %>
</table>
<br>
<a href="insertBoard">새글 등록</a>
</center>
</body>
```

2 게시글 목록 화면 개발

⟨3⟩ JSP 결과 페이지 확인

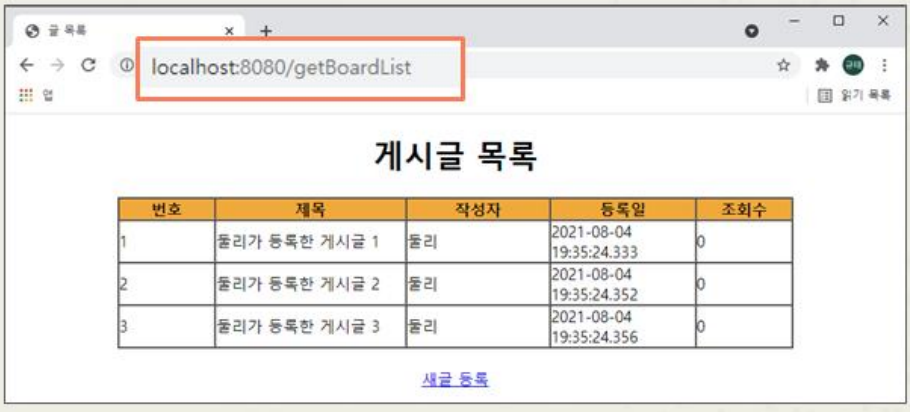
http://localhost:8080/board/getBoardList URL 요청을 전송했을 때

- BoardController.getBoardList 메소드가 실행되고 글목록 화면(getBoardList.jsp)이 출력되는지 확인함

JSP를 이용한 게시글 목록 개발

2 게시글 목록 화면 개발

〈3〉 JSP 결과 페이지 확인



3 EL과 JSTL 적용

〈1〉 EL(Expression Language)

- EL은 일반적으로 내장 객체에 등록된 데이터를 JSP 화면에 출력하는 용도로 사용함

JSP를 이용한 게시글 목록 개발

3 EL과 JSTL 적용

〈1〉 EL(Expression Language)

- EL 표현식은 ${}로 묶고 내부에서 도트(.) 연산자를 사용하여 내장 객체에 등록된 데이터에 접근함

3 EL과 JSTL 적용

〈1〉 EL(Expression Language)

- EL을 통해 접근한 값이 null인 경우에는 값을 출력하지 않을 뿐 에러를 발생시키지 않음

JSP를 이용한 게시글 목록 개발

3 EL과 JSTL 적용

〈2〉 JSTL(JSP Standard Tag Library)

- JSP에서 자주 사용하는 기능
(반복처리, 분기처리, 데이터 포맷 변경, XML 조작,
데이터베이스 엑세스)을 구현해 놓은
Custom Tag Library의 모음

3 EL과 JSTL 적용

〈2〉 JSTL(JSP Standard Tag Library)

- JSTL을 사용하면 Scriptlet에서 사용했던
if, for 같은 JAVA 구분을 제거할 수 있음
- JSTL과 EL을 결합하면 JSP 페이지에서 JAVA 코드를
완벽하게 제거할 수 있음

JSP를 이용한 게시글 목록 개발

3 EL과 JSTL 적용

〈2〉 JSTL(JSP Standard Tag Library)

- JSP는 EL을 기본적으로 지원하지만 JSTL은 별도로 라이브러리를 추가해야 함

```
        <dependency>
            <groupId>org.apache.tomcat.embed</groupId>
            <artifactId>tomcat-embed-jasper</artifactId>
        </dependency>

        <dependency>
            <groupId>javax.servlet</groupId>
            <artifactId>jstl</artifactId>
        </dependency>

    </dependencies>
```

3 EL과 JSTL 적용

〈3〉 EL과 JSTL을 적용한 JSP

- getBoardList.jsp 파일 상단에 두 개의 taglib 지시자를 설정함

core	for 루프나 if를 사용하기 위한 taglib
fmt	날짜나 숫자 관련 포맷을 설정하기 위한 taglib

```
<%@page contentType="text/html; charset=UTF-8"%>
<%@taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<%@taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="fmt" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
                      "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
```

JSP를 이용한 게시글 목록 개발

3 EL과 JSTL 적용

③ EL과 JSTL을 적용한 JSP

- core가 제공하는 <c:forEach> 태그를 이용하여 반복문을 처리함

3 EL과 JSTL 적용

③ EL과 JSTL을 적용한 JSP

- fmt가 제공하는 <fmt:formatDate> 태그를 이용하여 날짜 데이터의 포맷을 지정함

```
<c:forEach var="board" items="${boardList }">
<tr>
<td>${board.seq }</td>
<td align="left">${board.title }</td>
<td>${board.writer }</td>
<td><fmt:formatDate value="${board.createDate }" pattern="yyyy-MM-dd"></fmt:formatDate></td>
<td>${board.cnt }</td>
</tr>
</c:forEach>
```

적용 스프링 부트

실습단계
스프링 부트 프로젝트에 JSP 이용하여 게시글 목록 출력
스프링 부트에서 JSP를 이용한 화면 개발을 위해 2개의 라이브러리 필요
1. JSTL 라이브러리
2. tomcat-embed-jasper 라이브러리
라이브러리 두 개를 pom.xml에 추가
mvnrepository.com 사이트에서 키워드 검색 후 등록 가능
yml 파일 수정
JSP 파일의 경로를 완성해줌
중요! JSP 파일이 /WEB-INF/board 아래에 확장자 .jsp 형태로 작성되어야 함
Board 엔티티의 경우 '다대일 단방향/양방향 매핑' 변수 제거
writer 업데이트에서 제외
디폴트 값 insert
TestCase 적절히 주석 처리
에러 발생 소스들을 주석으로 막아줌

적용 스프링 부트

실습단계
getBoardList( ) 메소드 수정
타이틀에 like 검색이 동작하도록 findByTitleContaining 메소드 작성
BoardController는 get 방식으로 호출됨
중요! 비즈니스 객체를 호출하면 BoardRepository가 동작해 글목록을 SELECT 한 후에 리턴
JSP 파일 작성
src → main → webapp 폴더를 생성해야 함
JSP 위저드가 따로 제공되지 않음
Tip! 화면을 잠깐 멈추고 소스를 타이핑하며 진행할 것
JSTL taglib 지시자 2개 등록
게시글 목록 구성
일련번호, 타이틀, 등록날짜, 조회수 출력
BoardWebApplication 실행, 내장 톰캣 구동
주의! 애플리케이션 실행 시 테이블을 create 함
데이터를 유지하기 위해 update로 변경
H2 console을 열어 insert 작업
BOARD 테이블에 insert 작성
포트 번호 디폴트 : 8080
화면이 이상하게 출력된 이유?
Rest 메소드가 리턴한 문자열을 JSON 형태로 전달하기 때문
일반 @Controller 메소드로 변경
Reloading
브라우저 새로고침
Tip! 프로젝트 실행 시 문제가 생길 경우 1. getBoardList 경로 확인 2. Controller 확인 3. JSP 파일 이름 확인