

스프링 MVC

- 데이터 바인딩 -

폼관련 커스텀 태그

- ❖ 커맨드 객체의 내용을 폼 입력 요소로 바인딩
- ❖ 스프링 커스텀 폼 태그 라이브러리
 - taglib 디렉티브

```
<%@ taglib prefix="form"
    uri="http://www.springframework.org/tags/form" %>
```

폼관련 커스텀 태그

❖ <form:form>

- <form> 태그를 생성
 - action, method 속성을 지정하지 않은 경우
 - action : 현재 요청 url
 - method : post
 - form의 id : 디폴트값 command

```
<form:form>  
...  
</form:form>
```



```
<form id="command" action="/bind/join" method="post">  
...  
</form>
```

폼관련 커스텀 태그

❖ <form:form>

- 커맨드 객체를 폼에 바인딩 시키기
 - `commandName` 속성에 커맨드 객체의 모델 이름을 지정
 - `form`의 `id`가 `commandName`의 값으로 설정됨

```
<form:form commandName="member">  
...  
</form:form>
```



```
<form id="member" action="/bind/join" method="post">  
...  
</form>
```

폼관련 커스텀 태그

❖ 커맨드 객체의 프로퍼티와 폼 입력 요소간의 바인딩

- 요소별 커스텀 태그를 지정
- path 속성으로 바인딩할 커맨드 객체의 프로퍼티명 지정

❖ 단순 텍스트 바인딩

- <form:input>, <form:hidden>, <form:password>, <form:label>
- path 속성에 바인딩할 커맨드 객체의 프로퍼티명 지정
- id, name 속성값으로 바인딩된 프로퍼티명이 지정됨
- value 속성은 프로퍼티의 값으로 바인딩

```
<form:input path="프로퍼티명"/>
```


- GET으로 폼 페이지 이동시에도 모델 객체가 필수로 지정되어야 함.

```
@RequestMapping(method=RequestMethod.GET)
public String form(Member member) {
    return "join";
}
```

폼관련 커스텀 태그

❖ 단순 텍스트 바인딩

```
<form:form commandName="member">  
  사용자 ID : <form:input path="userId"/> <br>  
  패스워드 : <form:password path="password"/><br>  
  <input type="submit" value="확인">  
</form:form>
```



```
<form id="login" action="/bind/join" method="post">  
  사용자 ID :  
  <input id="userId" name="userId" type="text" value=""/> <br>  
  패스워드 :  
  <input id="password" name="password" type="password"  
    value=""/><br>  
  <input type="submit" value="확인">  
</form>
```

폼관련 커스텀 태그

❖ BindingResult 결과 메시지 출력

- 모델 객체에 에러 메시지 지정 - 어노테이션의 message 속성

```
public class Member {  
    @NotEmpty(message="사용자 ID는 필수 항목입니다.")  
    @Length(min=4, message="사용자 ID는 4글자 이상이어야 합니다.")  
    private String userId;  
  
    @NotEmpty(message="이름은 필수 항목입니다.")  
    private String name;  
  
    @NotEmpty(message="비밀번호는 필수 항목입니다.")  
    @Length(min=4, message="비밀번호는 4글자 이상이어야 합니다.")  
    private String password;  
  
    @NotEmpty(message="email은 필수 항목입니다.")  
    @Email(message="email 양식에 맞아야 합니다.")  
    private String email;  
  
    @DateTimeFormat(pattern="yyyy-MM-dd")  
    private Date date;
```

폼관련 커스텀 태그

❖ BindingResult 결과 메시지 출력

- 에러가 있는 경우 폼에 에러 출력
- `<form:errors path="프로퍼티명" element="에러메시지 엘리먼트" cssClass="클래스명" />`

```
<form:errors path="userId" element="div" cssClass="error"/>
```



```
<div id="userId.errors" class="error">  
사용자 ID는 필수 항목입니다..  
사용자 ID는 4글자 이상어야 합니다.  
</div>
```


폼관련 커스텀 태그

❖ BindingResult 결과 메시지 출력

- 에러가 있는 경우 폼에 에러

```
<form:form commandName="member">
  <p>
    사용자 ID :
    <form:input path="userId"/>
    <form:errors path="userId" element="div" cssClass="error"/>
  </p>
  <p>
    이름 :
    <form:input path="name"/>
    <form:errors path="name" element="div" cssClass="error"/>
  </p>
  <p>
    비밀번호 :
    <form:password path="password"/>
    <form:errors path="password" element="div" cssClass="error"/>
  </p>
```

폼관련 커스텀 태그

❖ BindingResult 결과 메시지 출력

- 에러가 있는 경우 폼에 에러

```
<p>
  email :
  <form:input path="email"/>
  <form:errors path="email" element="div" cssClass="error"/>
</p>
```

```
<p>
  생일 :
  <input type="text" name="date" />
</p>
```

```
<p>
  <input type="submit" >
</p>
```

```
</form:form>
```

폼관련 커스텀 태그

❖ <form:select>, <form:options>, <form:option>

- 모델 객체의 loginType 프로퍼티 추가

```
public class Login {  
    private String userId;  
    private String password;  
    private String loginType;  
    ...  
    public String getLoginType() {  
        return loginType;  
    }  
  
    public void setLoginType(String loginType) {  
        this.loginType = loginType;  
    }  
    ...  
}
```

폼관련 커스텀 태그

❖ <form:select>, <form:options>, <form:option>


- <form:select>
 - <select> 태그 생성, <option>을 구성할 컬렉션 사용 가능
- <form:options>
 - 지정한 컬렉션 객체를 이용하여 <option> 태그 생성
- <form:option>
 - 한 개의 <option>태그를 생성
- <option> 구성을 위한 모델 객체가 필요
 - @ModelAttribute 메서드로 컬렉션을 반환

```
@ModelAttribute("loginTypes")
List<String> loginType() {
    List<String> loginTypes = new ArrayList<String>();
    loginTypes.add("일반회원");
    loginTypes.add("기업회원");
    loginTypes.add("헤드헌터회원");
    return loginTypes;
}
```

폼관련 커스텀 태그

❖ <form:select>

```
<form:form commandName="login">
  사용자 ID : <form:input path="userId"/> <br>
  비밀번호 : <form:password path="password"/><br>
  회원 타입 : <form:select path="loginType" items="${loginTypes}"/><br>
  <input type="submit" value="확인">
</form:form>
```



```
<select id="loginType" name="loginType">
  <option value="일반회원">일반회원</option>
  <option value="기업회원">기업회원</option>
  <option value="헤드헌터회원">헤드헌터회원</option>
</select><br>
```

폼관련 커스텀 태그

❖ <form:select>

- 커맨드객체 프로퍼티의 값과 일치하는 항목이 있는 경우
 - 해당 <option> 태그에 selected 속성이 배정


```
<select id="loginType" name="loginType">
  <option value="일반회원">일반회원</option>
  <option value="기업회원">기업회원</option>
  <option value="헤드헌터회원" selected="selected">헤드헌터회원</option>
</select>
```

폼관련 커스텀 태그

❖ <form:options>

- 컬렉션을 기반으로 <option>태그를 생성
- 컬렉션에 없는 <option>을 추가할 때 사용

```
<form:select path="loginType">
  <option value="">--선택하세요--</option>
  <form:options items="${loginTypes}" />
</form:select>
```




```
<select id="loginType" name="loginType">
  <option value="">--선택하세요--</option>
  <option value="일반회원">일반회원</option>
  <option value="기업회원">기업회원</option>
  <option value="헤드헌터회원">헤드헌터회원</option>
</select>
```

폼관련 커스텀 태그

❖ <form:option>

- 한 개의 <option> 태그 생성
- 개별적으로 <option>을 생성시킬 때 사용

```
<form:select path="loginType">  
  <form:option value="일반회원"/>  
  <form:option value="기업회원">기업회원</form:option>  
  <form:option value="헤드헌터회원" label="헤드헌터회원"/>  
</form:select>
```



```
<select id="loginType" name="loginType">  
  <option value="일반회원">일반회원</option>  
  <option value="기업회원">기업회원</option>  
  <option value="헤드헌터회원">헤드헌터회원</option>  
</select>
```


폼관련 커스텀 태그

❖ <form:select>

- value와 label을 다른 값으로 설정해야 하는 경우
- option을 나타내는 객체를 정의하여 각각 value와 label을 따로 지정

```
public class Code {  
    private String code;  
    private String label;  
    ...  
}
```

```
@ModelAttribute("loginTypes")  
List<Code> loginTypes() {  
    List<Code> loginTypes = new ArrayList<Code>();  
    loginTypes.add(new Code("0", "일반회원"));  
    loginTypes.add(new Code("1", "기업회원"));  
    loginTypes.add(new Code("2", "헤드헌터회원"));  
    return loginTypes;  
}
```

폼관련 커스텀 태그

❖ <form:select>

```
<form:select path="loginType" items="${loginTypes}"  
            itemValue="code" itemLabel="label"/>
```



```
<select id="loginType" name="loginType">  
    <option value="0">일반회원</option>  
    <option value="1">기업회원</option>  
    <option value="2">헤드헌터회원</option>  
</select><br>
```

폼관련 커스텀 태그

❖ <form:checkboxes>, <form:checkbox>

- <input type="checkbox"> 태그를 생성

```
<form:checkboxes path="favorites" items=${"favoritesItems"}/>
```

```
<form:checkboxes path="favorites" items=${"favoritesItems"}  
    itemValue="code" itemLabel="label"/>
```

```
<form:checkbox path="contractAgreement" label="약관에 동의합니다"/>
```

폼관련 커스텀 태그

❖ <form:radiobuttons>, <form:radiobutton>

- <input type="radio"> 태그를 생성

```
<form:checkboxes path="tool" items=${"tools"}/>
```

```
<form:checkboxes path="tool" items=${"tools"}  
    itemValue="code" itemLabel="label"/>
```

폼관련 커스텀 태그

❖ <form:textarea>

- <textarea>태그를 생성

```
<form:textarea path="etc" rows="5" cols="20"/>
```

```
<textarea id="etc" name="etc" rows="5" cols="20"></textarea>
```