Spring MVC2 04

[타임리프 - 스프링 Form]

스프링 MVC 2편 - 백엔드 웹 개발 활용 기술

김영한

2022.10.19

타임리프와 스프링의 통합 폼

- 스프링과 타임리프가 통합으로 돌아가는 기능

타임리프 메뉴얼

기본 : https://www.thymeleaf.org/doc/tutorials/3.0/usingthymeleaf.html

스프링 통합 : https://www.thymeleaf.org/doc/tutorials/3.0/thymeleafspring.html

기본적으로 추가되는 기능들

- 스프링 SpringEL문법 통합
- 스프링 빈 호출 지원
- 편리한 폼 관리를 위한 추가 속성
- 폼 컴포넌트 기능
- 스프링의 메시지, 국제화 기능의 편리한 통합
- 스프링의 검증, 오류 처리 통합
- 스프링의 변환 서비스 통합

타임리프 폼 기능 사용

등록폼에서 – 우선 빈 객체라도 model에 넘겨줘야한다.

model.addAttribute("item", new Item());

```
<form action="item.html" th:action method="post" th:object="${item}">
   <div>
       <label for="itemName">상품명</label>
```

```
        <input type="text" id="itemName" th:field="*{itemName}" class="form-control"
placeholder="이름을 입력하세요">
    </div>
```

⇒ **th:object="${**객체명**}"** 으로 확인한다**.**

⇒ 사용시에 **${item.itemName}**으로도 사용할 수 있지만 폼에 객체명을 맞춰놓는다면

***{itemName}** 으로 변경해서 사용할 수 있다**.**

th:field="${item.itemName}" ⇒ name="itemName", id="itemName" , value=""를 생성해준다.

value를 뿌려주기에 수정시에 더 간편

편리한 폼 데이터 관리

- 체크 박스, 라디오 박스, 셀렉트 박스, 체크 박스 예시

단일 체크 박스

순수 **HTML**

```
<!-- single checkbox -->
<div>판매 여부</div>
<div>
    <div class="form-check">
        <input type="checkbox" id="open" name="open" class="form-check-input">
        <label for="open" class="form-check-label">판매 오픈</label>
    </div>
</div>
```

체크를 하게되면 on → true 로 변환되고

체크를 안하게 되면 값이 넘어가질 않는다. 로그를 출력해 본다면 null 값이 나온다.

```
log.info("item.open={}",item.getOpen());
```

```
item.open=true
item.open=null
```

서버에서 결과가 null 이기 때문에 문제가 생길 수 있다.

수정시 체크를 빼고 전달하게 되면 값이 전달안되기 때문에 수정이 안될수도있다.

그러므로 스프링 **MVC**는 히든필드를 추가해서 해결하게 해준다.

```
<div class="form-check">
   <input type="checkbox" id="open" name="open" class="form-check-input">
   <input type="hidden" name="_open" value="on"/><!--히든 필드 추가-->
   <label for="open" class="form-check-label">판매 오픈</label>
</div>
```

로그 출력시 false가 제대로 출력된다.

```
item.open=true
item.open=false
```

스프링MVC가 open, _open 이 있다면 open이 없을때 _open의 값을 보고 false로 확인 시킨다.

타임리프 단일 체크 박스

```
<div class="form-check">
   <input type="checkbox" id="open" th:field="${item.open}"
class="form-check-input">
   <label for="open" class="form-check-label">판매 오픈</label>
</div>
```

히든필드 부분도 자동으로 생성해주기 때문에 스프링MVC가 체크박스를 처리하는 것과 같은 방식으로 동작한다. 값을 가져와서 **checked** 하는 부분 까지 th:field가 해결해준다.

멀티 체크 박스

```
@ModelAttribute("regions")
public Map<String, String> regions(){
   Map<String, String> regions = new LinkedHashMap<>();
   regions.put("SEOUL", "서울");
   regions.put("BUSAN", "부산");
   regions.put("JEJU", "제주");
   return regions;
}
```

선택 박스 선택지를 여러 컨트롤러에 전부 넣어주기 위해 사용

각각 컨트롤러에 model.addAttribute사용하는 것과 같다.

```
<div>
   <div>등록 지역</div>
   <div th:each="region : ${regions}" class="form-check form-check-inline">
       <input type="checkbox" th:field="*{regions}" th:value="${region.key}"
              class="form-check-input">
       <label th:for="${#ids.prev('regions')}"
              th:text="${region.value}" class="form-check-label">서울</label>
   </div>
</div>
```

```
th:for="${#ids.prev('regions')}"
```

소스보기로 확인하면

```
<div class="form-check form-check-inline">
   <input type="checkbox" value="SEOUL"
          class="form-check-input" id="regions1" name="regions">
<input type="hidden" name="_regions" value="on"/>
   <label for="regions1" class="form-check-label">서울</label>
```

id 자동 생성 하므로 1, 2, 3 넣어주게 되고 label 맞춰주기 위해 동적 아이디 매칭

```
th:for="${#ids.prev('regions')}"
```

⇒ name 이 같으면 Parameter로 List를 받는다.

조회 할때도 값이 있다면 체크가 자동으로 **checked** 처리가 된다**.**

라디오 버튼

라디오 버튼은 여러 선택지 중에 하나 선택

ENUM사용해서 만들어본다.

-

```
public enum ItemType {

   BOOK("도서"), FOOD("음식"), ETC("기타");

   private final String description;

   ItemType(String description) {
       this.description = description;
   }
}
```

enum 타입 생성.

```
@ModelAttribute("itemTypes")
public ItemType[] itemTypes(){
   return ItemType.values();
}
```

[BOOK, FOOD, ETC] 형태로 배열로 저장

```
<!-- radio button -->
<div>
   <div>상품 종류</div>
   <div th:each="type : ${itemTypes}" class="form-check form-check-inline">
       <input type="radio" th:field="*{itemType}" th:value="${type.name()}"
              class="form-check-input">
       <label th:for="${#ids.prev('itemType')}" th:text="${type.description}"
              class="form-check-label">
           BOOK
       </label>
   </div>
</div>
```

체크를 안할 시에 null이 넘어가게된다.

라디오박스는 체크하면 수정할 때 무조껀 한곳이라도 체크가 되어야 한다.

```
th:each="type : ${itemTypes}
```

ENUM 같은 경우는 아래의 방식으로 직접 접근이 가능하지만. 패키지가 바뀌거나 하면 곤란하니 권장은 하지 않는다.

```
th:each="type : ${T(hello.itemservice.domain.item.ItemType).values()}"
```

셀렉트 박스

```
@ModelAttribute("deliveryCodes")
public List<DeliveryCode> deliveryCodes(){
   List<DeliveryCode> deliveryCodes = new ArrayList<>();
   deliveryCodes.add(new DeliveryCode("FAST", "빠른 배송"));
   deliveryCodes.add(new DeliveryCode("NORMAL", "일반 배송"));
   deliveryCodes.add(new DeliveryCode("SLOW", "느린 배송"));
   return deliveryCodes;
}
```

자바 객체로 셀렉트 박스 생성해보기

```
<div>배송 방식</div>
<select th:field="*{deliveryCode}" class="form-select">
   <option value="">==배송 방식 선택==</option>
   <option th:each="deliveryCode : ${deliveryCodes}" th:value="${deliveryCode.code}"
           th:text="${deliveryCode.displayName}">FAST</option>
</select>
```