

# 학습 목표

---

- 1. Thymeleaf의 특징을 알아본다.
  - 2. 서버에서 가공한 데이터를 Thymeleaf라는 템플릿 엔진을 활용하여 클라이언트에 렌더링하는 방법을 학습한다.
  - 3. 예제를 통해 Thymeleaf의 기본적인 문법을 학습한다.
-

## 3.1 Thymeleaf 소개

---

1. 서버 사이드 렌더링 : 미리 정의된 템플릿(Template)을 만들고 동적으로 HTML 페이지를 만들어서 클라이언트에 전달하는 방식  
요청이 올 때마다 서버에서 새로운 HTML 페이지를 만들어 주기 때문에 서버 사이드 렌더링 방식이라고 함.

2. Thymeleaf : 서버 사이드 템플릿 엔진의 한 종류

---

## 3.1 Thymeleaf 소개

---

- Thymeleaf의 가장 큰 장점은 'natural templates'
- Thymeleaf를 사용할 때 Thymeleaf 문법을 포함하고 있는 html 파일을 서버 사이드 렌더링을 하지 않고 브라우저에 띄워도 정상적인 화면을 볼 수 있음
- 스프링에서 권장하는 서버 사이드 템플릿 엔진

## 3.1 Thymeleaf 소개



[함께 해봐요 3-1] 웹 브라우저에서 Thymeleaf 파일 열어보기

thymeleafEx01.html

```
01 <!DOCTYPE html>
02 <html xmlns:th="http://www.thymeleaf.org">
03 <head>
04     <meta charset="UTF-8">
05     <title>Title</title>
06 </head>
07 <body>
08     <p th:text="${data}">Hello Thymeleaf!!</p>
09 </body>
10 </html>
```



[그림 3-1] 웹 브라우저에서 Thymeleaf 문법을 포함한 html 파일 열어보기

th:text="\${data}"  
라는 Thymeleaf 문  
법이 들어갔지만  
html 파일이 깨지  
지 않고 정상적으  
로 출력되는 것을  
확인 가능

## 3.1 Thymeleaf 소개



[함께 해봐요 3-2] Thymeleaf 예제용 컨트롤러 클래스 만들기

com.shop.controller.ThymeleafExController.java

```
01 package com.shop.controller;
02
03 import org.springframework.stereotype.Controller;
04 import org.springframework.ui.Model;
05 import org.springframework.web.bind.annotation.GetMapping;
06 import org.springframework.web.bind.annotation.RequestMapping;
07
08 @Controller
09 @RequestMapping(value="/thymeleaf") ..... ❶
10 public class ThymeleafExController {
11
12     @GetMapping(value = "/ex01")
13     public String thymeleafExample01(Model model){
14         model.addAttribute("data", "타임리프 예제 입니다."); ..... ❷
15         return "thymeleafEx/thymeleafEx01"; ..... ❸
16     }
17
18 }
```

## 3.1 Thymeleaf 소개

---



[함께 해봐요 3-3] 서버용 Thymeleaf 파일

resources/templates/thymeleafEx/thymeleafEx01.html

```
01 <!DOCTYPE html>
02 <html xmlns:th="http://www.thymeleaf.org"> ..... 1
03 <head>
04     <meta charset="UTF-8">
05     <title>Title</title>
06 </head>
07 <body>
08     <p th:text="${data}">Hello Thymeleaf!!</p> ..... 2
09 </body>
10 </html>
```

## 3.1 Thymeleaf 소개

---

- 애플리케이션 실행 후 해당 url로 접근 시 "Hello Thymeleaf!!" 대신 "타임리프 예제 입니다." 라는 문구가 나타나는 것을 볼 수 있음
- 디자이너 또는 퍼블리셔는 자신이 작업한 내용을 html 파일로 바로 열어서 확인할 수 있으며, 개발자는 디자이너 또는 퍼블리셔로부터 html 파일을 받아서 html 태그 안에 Thymeleaf 문법을 추가하는 것만으로 동적으로 html 파일을 생성할 수 있음



[그림 3-3] ShopApplication 실행 결과

## 3.2 Spring Boot Devtools

---

- Spring Boot Devtools는 애플리케이션 개발 시 유용한 기능들을 제공하는 모듈
    - 1. Automatic Restart: classpath에 있는 파일이 변경될 때마다 애플리케이션을 자동으로 재시작
    - 2. Live Reload: 정적 자원(html, css, js) 수정 시 새로 고침 없이 바로 적용
    - 3. Property Defaults: Thymeleaf는 기본적으로 성능을 향상시키기 위해서 캐싱 기능을 사용. 개발하는 과정에서 캐싱 기능을 사용한다면 수정한 소스가 제대로 반영되지 않을 수 있기 때문에 cache의 기본값을 false로 설정가능
-



## 3.2 Spring Boot Devtools

---



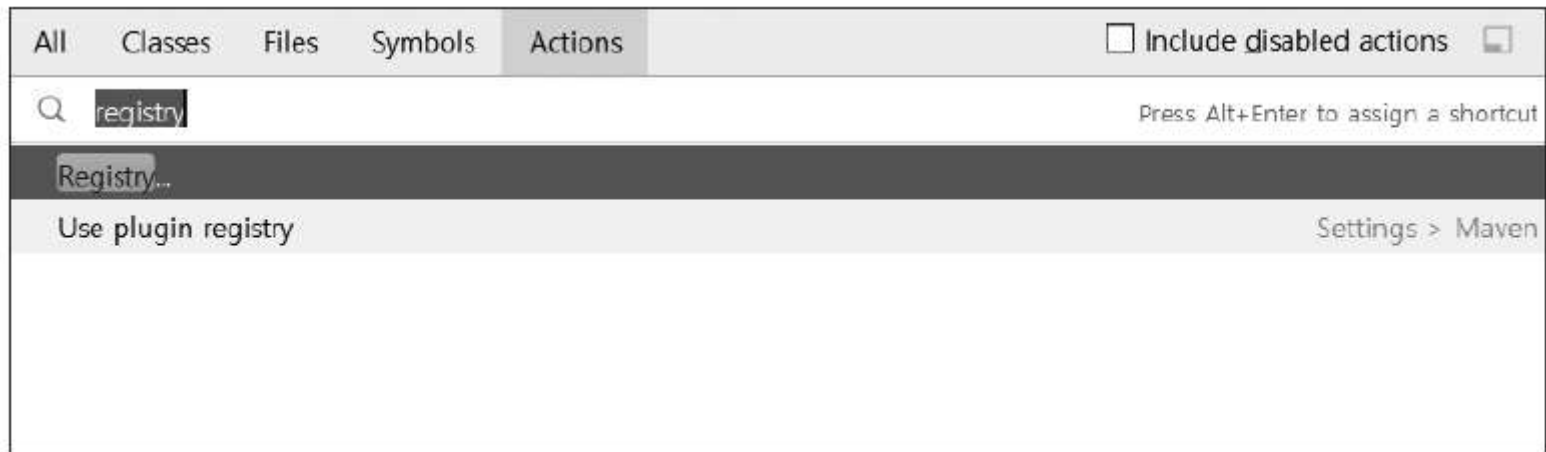
[함께 해봐요 3-4] pom.xml에 의존성 추가하기

```
01 <dependency>
02     <groupId>org.springframework.boot</groupId>
03     <artifactId>spring-boot-devtools</artifactId>
04 </dependency>
```

# Automatic Restart 적용

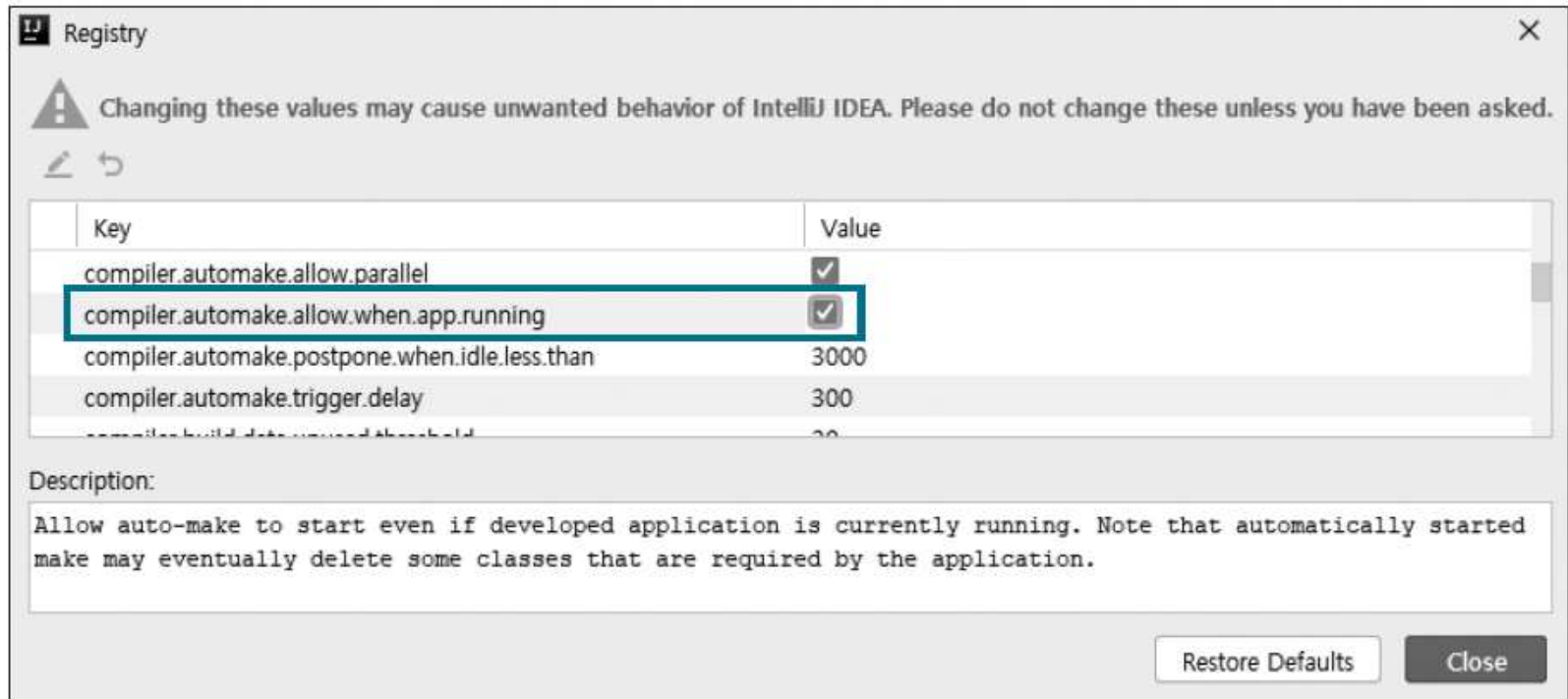
---

- <Shift>키를 연속으로 2번 클릭 -> 검색창
- Actions 탭에서 "registry"를 검색



[그림 3-4] Automatic Restart 적용하기 1단계

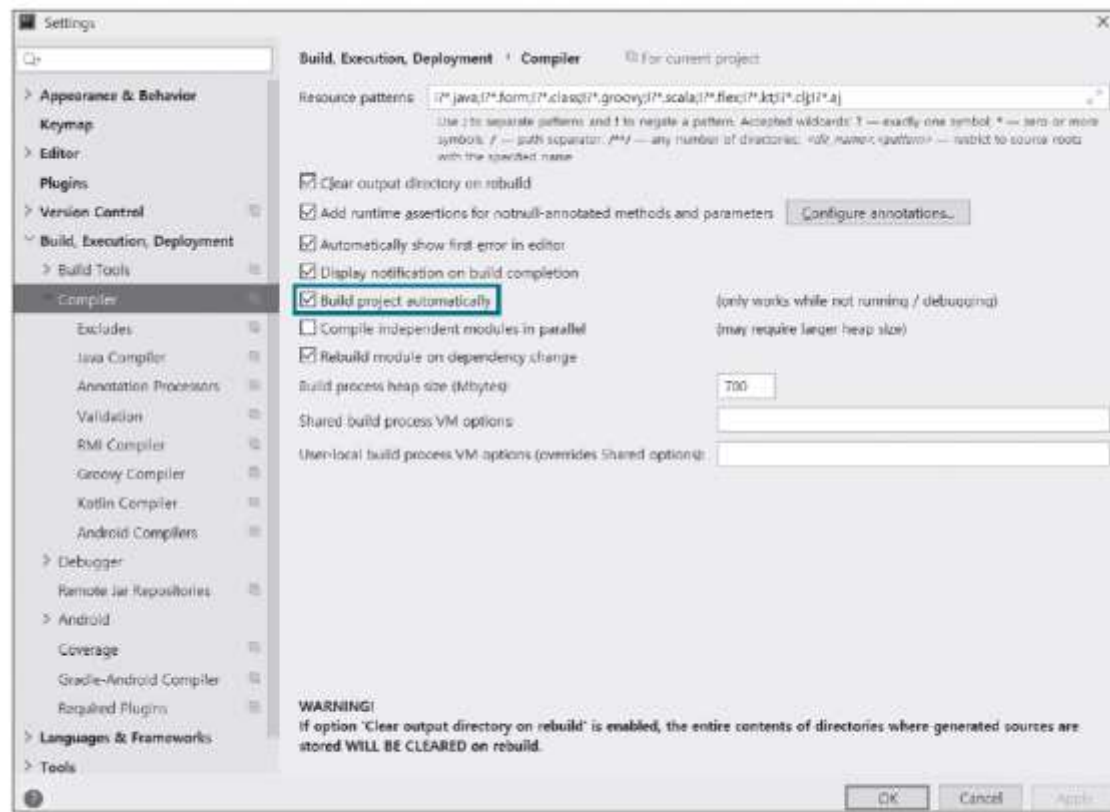
# Automatic Restart 적용



[그림 3-5] Automatic Restart 적용하기 2단계

# Automatic Restart 적용

- [File]–[Settings]–[Build], [Execution], [Deployment]–[Compiler] 메뉴 “Build project automatically” 체크



[그림 3-6] Automatic Restart 적용하기 3단계

# Live Reload 적용

---

- 브라우저의 새로고침을 하지 않아도 변경된 리소스가 웹 브라우저에 반영

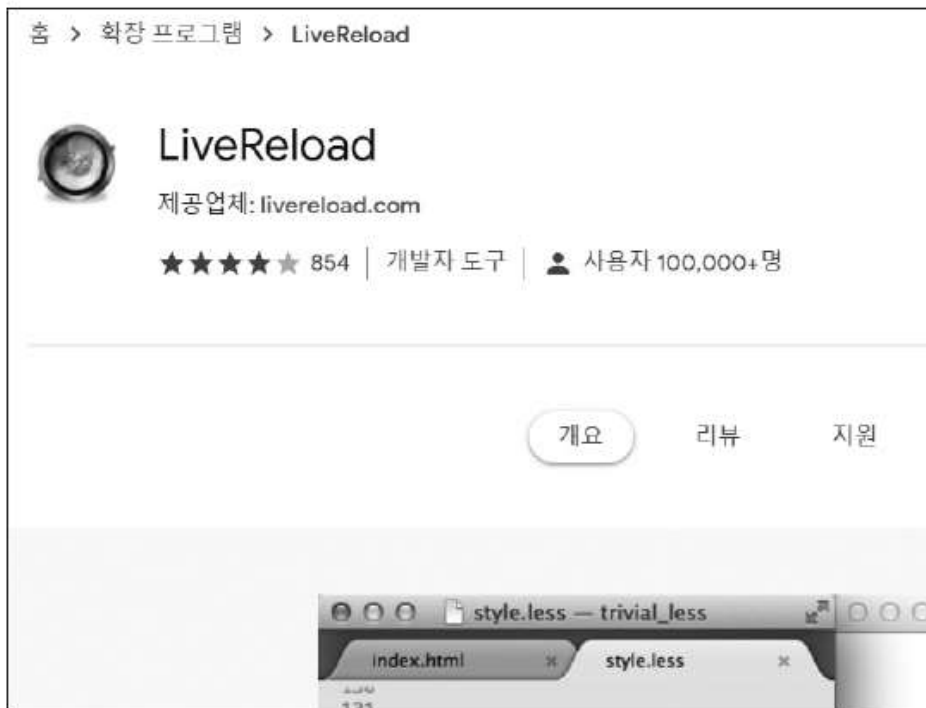


[함께 해봐요 3-5] application.properties Live Reload 적용 설정 추가하기

```
01 #Live Reload 기능 활성화  
02 spring.devtools.livereload.enabled=true
```

# Live Reload 적용

- 구글 크롬 웹 스토어에서 LiveReload를 검색



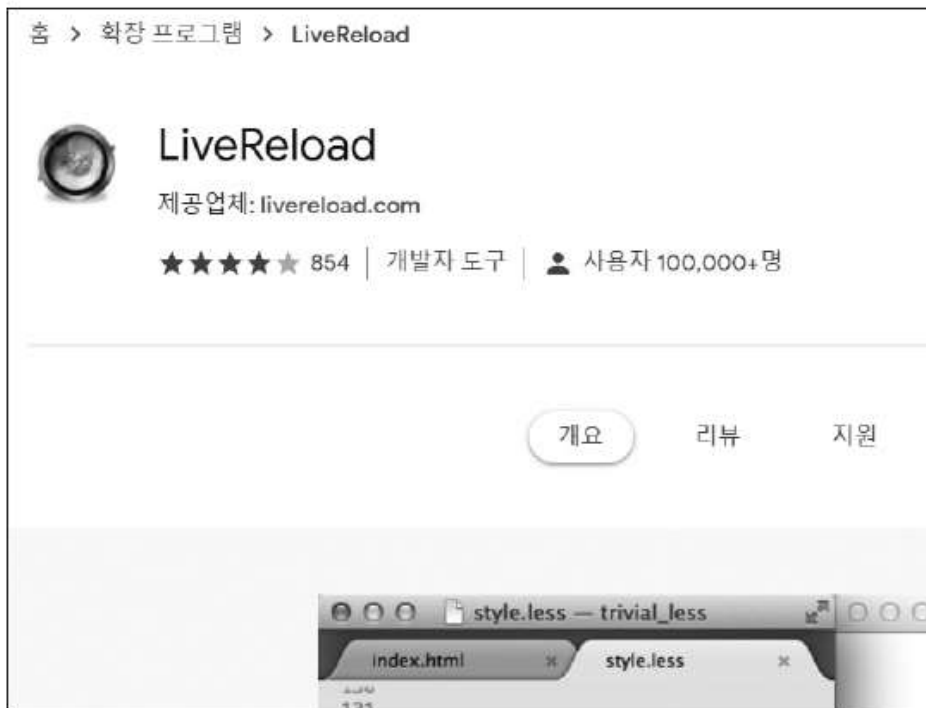
[그림 3-7] Live Reload 적용하기 1단계



[그림 3-8] Live Reload 적용하기 2단계

# Live Reload 적용

- 구글 크롬 웹 스토어에서 LiveReload 검색 및 설치



[그림 3-7] Live Reload 적용하기 1단계



[그림 3-8] Live Reload 적용하기 2단계

# Live Reload 적용



[그림 3-9] Live Reload 적용하기 3단계

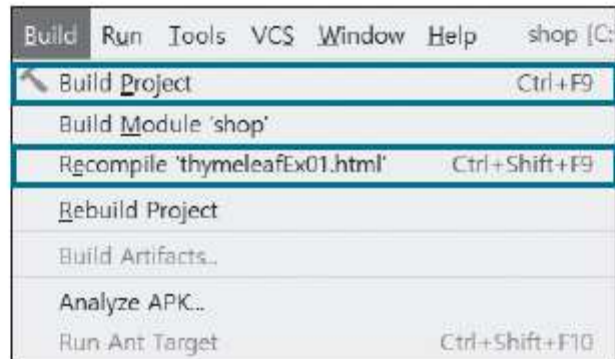


[그림 3-10] Live Reload 활성화 결과



# Live Reload 적용

---



[그림 3-11] Live Reload를 활용하며 애플리케이션 재구동 없이 웹 브라우저에 반영

# Property Defaults적용

---

- Thymeleaf의 캐싱 기능을 false 설정
- application.properties 분리 후 운영환경에서는 캐싱 기능을 사용하고, 개발 환경에서는 캐싱 기능을꺼두는 방법으로 관리



[함께 해봐요 3-6] application.properties Property Defaults 설정 추가하기

```
01 #Thymeleaf cache 사용 중지  
02 spring.thymeleaf.cache = false
```

## 3.3 Thymeleaf 예제 진행 - th:text



[함께 해봐요 3-7] th:text를 이용한 상품 데이터 출력용 Dto 클래스

com.shop.dto.ItemDto

```
01 package com.shop.dto;
02
03 import lombok.Getter;
04 import lombok.Setter;
05
06 import java.time.LocalDateTime;
07
08 @Getter
09 @Setter
10 public class ItemDto {
11
12     private Long id;
13
14     private String itemNm;
15
16     private Integer price;
17
18     private String itemDetail;
19
20     private String sellStatCd;
21
22     private LocalDateTime regTime;
23
24     private LocalDateTime updateTime;
25
26 }
```

\* th:text : 화면에 데이터 출력

\* 뷰 영역에서 사용할  
ItemDto 클래스 생성

## 3.3 Thymeleaf 예제 진행 - th:text



[함께 해보요 3-8] th:text를 이용한 상품 데이터 출력용 컨트롤러 클래스

com.shop.controller.ThymeleafExController.java

```
01 package com.shop.controller;
02
03 .....기존 임포트 생략.....
04
05 import com.shop.dto.ItemDto;
06 import java.time.LocalDateTime;
07
08 @Controller
09 @RequestMapping(value="/thymeleaf")
10 public class ThymeleafExController {
11
12     .....코드 생략.....
13
14     @GetMapping(value = "/ex02")
15     public String thymeleafExample02(Model model){
16         ItemDto itemDto = new ItemDto();
17         itemDto.setItemDetail("상품 상세 설명");
18         itemDto.setItemNm("테스트 상품1");
19         itemDto.setPrice(10000);
20         itemDto.setRegTime(LocalDateTime.now());
21
22         model.addAttribute("itemDto", itemDto);
23         return "thymeleafEx/thymeleafEx02";
24     }
25 }
```

## 3.3 Thymeleaf 예제 진행 - th:text



[함께 해봐요 3-9] th:text를 이용한 상품 데이터 출력용 thymeleaf 파일

resources/templates/thymeleafEx02.html

```
01 <!DOCTYPE html>
02 <html xmlns:th="http://www.thymeleaf.org">
03 <head>
04     <meta charset="UTF-8">
05     <title>Title</title>
06 </head>
07 <body>
08     <h1>상품 데이터 출력 예제</h1>
09     <div>
10         상품명 : <span th:text="${itemDto.itemNm}"></span>
11     </div>
12     <div>
13         상품상세설명 : <span th:text="${itemDto.itemDetail}"></span>
14     </div>
15     <div>
16         상품등록일 : <span th:text="${itemDto.regTime}"></span>
17     </div>
18     <div>
19         상품가격 : <span th:text="${itemDto.price}"></span>
20     </div>
21 </body>
22 </html>
```

전달받은 itemDto 객체를 th:text를 이용하여 화면에 출력

### 3.3 Thymeleaf 예제 진행 - th:text

---



**[그림 3-12]** 상품 데이터 출력 예제 실행 결과

## 3.3 Thymeleaf 예제 진행 - th:each

- th:each : 타임리프에서 반복문 처리



[함께 해봐요 3-10] th:each를 이용한 상품 리스트 출력용 컨트롤러

com.shop.controller.ThymeleafExController.java

```
01 package com.shop.controller;
02
03 .....기존 임포트 생략.....
04
05 import java.util.ArrayList;
06 import java.util.List;
07
08 @Controller
09 @RequestMapping(value="/thymeleaf")
10 public class ThymeleafExController {
11
12     .....코드 생략.....
13 }
```

## 3.3 Thymeleaf 예제 진행 - th:each

```
14  @GetMapping(value = "/ex03")
15  public String thymeleafExample03(Model model){
16
17      List<ItemDto> itemDtoList = new ArrayList<>();
18
19      for(int i=1;i<=10;i++){ ..... ❶
20
21          ItemDto itemDto = new ItemDto();
22          itemDto.setItemDetail("상품 상세 설명"+i);
23          itemDto.setItemNm("테스트 상품" + i);
24          itemDto.setPrice(1000*i);
25          itemDto.setRegTime(LocalDate.now());
26
27          itemDtoList.add(itemDto);
28      }
29
30      model.addAttribute("itemDtoList", itemDtoList); ..... ❷
31      return "thymeleafEx/thymeleafEx03";
32  }
33 }
```



## 3.3 Thymeleaf 예제 진행 - th:each

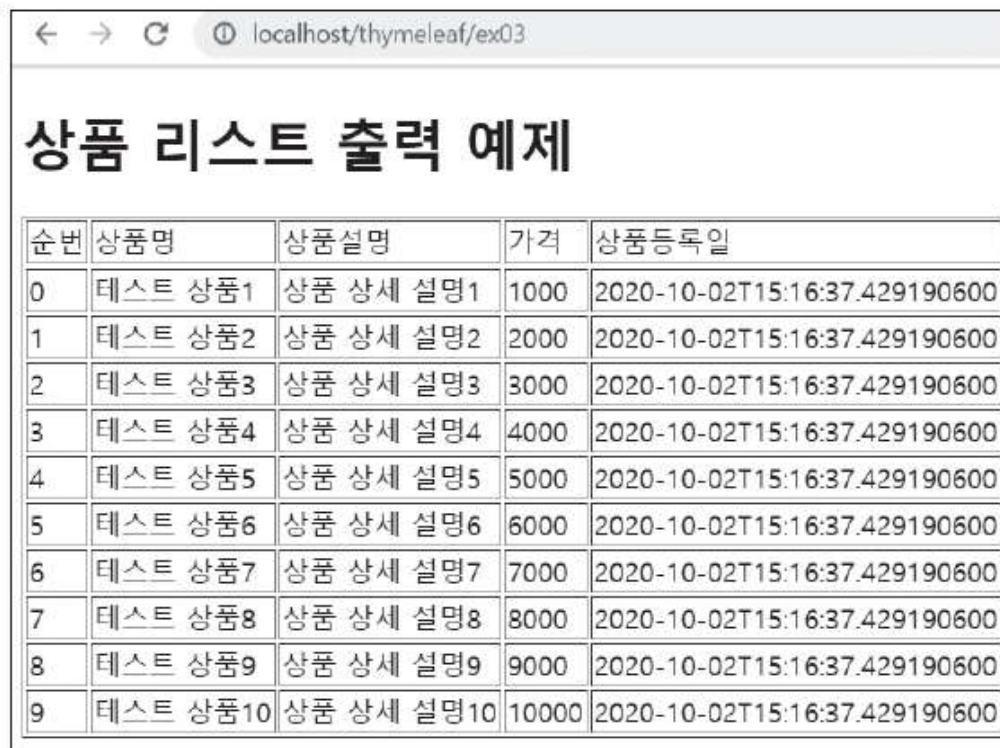


[함께 해봐요 3-11] th:each를 이용한 상품 리스트 출력용 thymeleaf 파일

resources/templates/thymeleafEx03.html

```
01 <!DOCTYPE html>
02 <html xmlns:th="http://www.thymeleaf.org">
03 <head>
04     <meta charset="UTF-8">
05     <title>Title</title>
06 </head>
07 <body>
08
09     <h1>상품 리스트 출력 예제</h1>
10
11     <table border="1">
12         <thead>
13             <tr>
14                 <td>순번</td>
15                 <td>상품명</td>
16                 <td>상품설명</td>
17                 <td>가격</td>
18                 <td>상품등록일</td>
19             </tr>
20         </thead>
21         <tbody>
22             <tr th:each="itemDto, status: ${itemDtoList}"> ..... ❶
23                 <td th:text="${status.index}"></td> ..... ❷
24                 <td th:text="${itemDto.itemNm}"></td>
25                 <td th:text="${itemDto.itemDetail}"></td>
26                 <td th:text="${itemDto.price}"></td>
27                 <td th:text="${itemDto.regTime}"></td>
28             </tr>
29         </tbody>
30     </table>
31
32 </body>
33 </html>
```

### 3.3 Thymeleaf 예제 진행 - th:each



순번	상품명	상품설명	가격	상품등록일
0	테스트 상품1	상품 상세 설명1	1000	2020-10-02T15:16:37.429190600
1	테스트 상품2	상품 상세 설명2	2000	2020-10-02T15:16:37.429190600
2	테스트 상품3	상품 상세 설명3	3000	2020-10-02T15:16:37.429190600
3	테스트 상품4	상품 상세 설명4	4000	2020-10-02T15:16:37.429190600
4	테스트 상품5	상품 상세 설명5	5000	2020-10-02T15:16:37.429190600
5	테스트 상품6	상품 상세 설명6	6000	2020-10-02T15:16:37.429190600
6	테스트 상품7	상품 상세 설명7	7000	2020-10-02T15:16:37.429190600
7	테스트 상품8	상품 상세 설명8	8000	2020-10-02T15:16:37.429190600
8	테스트 상품9	상품 상세 설명9	9000	2020-10-02T15:16:37.429190600
9	테스트 상품10	상품 상세 설명10	10000	2020-10-02T15:16:37.429190600

[그림 3-13] 상품 리스트 출력 예제 실행 결과

## 3.3 Thymeleaf 예제 진행 - th:if, th:unless

- th:if, th:unless : 타임리프에서 조건문 처리
- 순번이 짝수이면 '짝수'를, 출력하고 짝수가 아니라면 '홀수'를 출력 예제



[함께 해봐요 3-12] th:if, th:unless를 이용한 조건문 처리용 컨트롤러 작성하기

com.shop.controller.ThymeleafExController.java

```
01 package com.shop.controller;
02
03 .....기존 임포트 생략.....
04
05 @Controller
06 @RequestMapping(value="/thymeleaf")
07 public class ThymeleafExController {
08
09     .....코드 생략.....
10
11     @GetMapping(value = "/ex04")
12     public String thymeleafExample04(Model model){
13
```

## 3.3 Thymeleaf 예제 진행 - th:if, th:unless

---

```
14      List<ItemDto> itemDtoList = new ArrayList<>();
15
16      for(int i=1;i<=10;i++){
17
18          ItemDto itemDto = new ItemDto();
19          itemDto.setItemDetail("상품 상세 설명"+i);
20          itemDto.setItemNm("테스트 상품" + i);
21          itemDto.setPrice(1000*i);
22          itemDto.setRegTime(LocalDateTime.now());
23
24          itemDtoList.add(itemDto);
25      }
26
27      model.addAttribute("itemDtoList", itemDtoList);
28      return "thymeleafEx/thymeleafEx04";
29  }
30 }
```

## 3.3 Thymeleaf 예제 진행 - th:if, th:unless



[함께 해봐요 3-13] th:if, th:unless를 이용한 조건문 처리용 thymeleaf 파일 만들기

resources/templates/thymeleafEx04.html

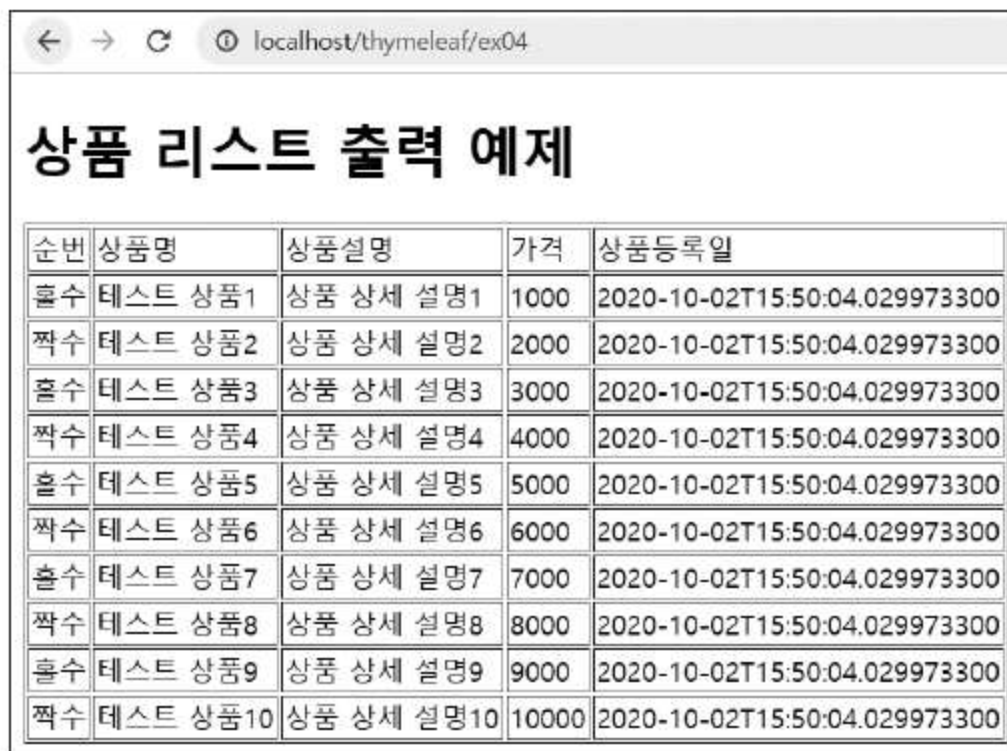
```
01 <!DOCTYPE html>
02 <html xmlns:th="http://www.thymeleaf.org">
03 <head>
04     <meta charset="UTF-8">
05     <title>Title</title>
06 </head>
07 <body>
08
09 <h1>상품 리스트 출력 예제</h1>
10
11 <table border="1">
12     <thead>
13     <tr>
14         <td>순번</td>
15         <td>상품명</td>
16         <td>상품설명</td>
17         <td>가격</td>
18         <td>상품등록일</td>
19     </tr>
```

### 3.3 Thymeleaf 예제 진행 - th:if, th:unless

---

```
20 </thead>
21 <tbody>
22 <tr th:each="itemDto, status: ${itemDtoList}">
23     <td th:if="${status.even}" th:text="짝수"></td> ..... ❶
24     <td th:unless="${status.even}" th:text="홀수"></td> ..... ❷
25     <td th:text="${itemDto.itemNm}"></td>
26     <td th:text="${itemDto.itemDetail}"></td>
27     <td th:text="${itemDto.price}"></td>
28     <td th:text="${itemDto.regTime}"></td>
29 </tr>
30 </tbody>
31 </table>
32
33 </body>
34 </html>
```

### 3.3 Thymeleaf 예제 진행 - th:if, th:unless



순번	상품명	상품설명	가격	상품등록일
홀수	테스트 상품1	상품 상세 설명1	1000	2020-10-02T15:50:04.029973300
짝수	테스트 상품2	상품 상세 설명2	2000	2020-10-02T15:50:04.029973300
홀수	테스트 상품3	상품 상세 설명3	3000	2020-10-02T15:50:04.029973300
짝수	테스트 상품4	상품 상세 설명4	4000	2020-10-02T15:50:04.029973300
홀수	테스트 상품5	상품 상세 설명5	5000	2020-10-02T15:50:04.029973300
짝수	테스트 상품6	상품 상세 설명6	6000	2020-10-02T15:50:04.029973300
홀수	테스트 상품7	상품 상세 설명7	7000	2020-10-02T15:50:04.029973300
짝수	테스트 상품8	상품 상세 설명8	8000	2020-10-02T15:50:04.029973300
홀수	테스트 상품9	상품 상세 설명9	9000	2020-10-02T15:50:04.029973300
짝수	테스트 상품10	상품 상세 설명10	10000	2020-10-02T15:50:04.029973300

[그림 3-14] 상품 리스트 출력 예제 실행 결과

### 3.3 Thymeleaf 예제 진행 - th:switch, th:case

---

- th:switch, th:case : 타임리프에서 조건문 처리
- 여러 개의 조건을 처리할 때 사용



## 3.3 Thymeleaf 예제 진행 - th:switch, th:case



[함께 해봐요 3-14] th:switch, th:case를 이용한 조건문 처리용 thymeleaf 파일

resources/templates/thymeleafEx04.html

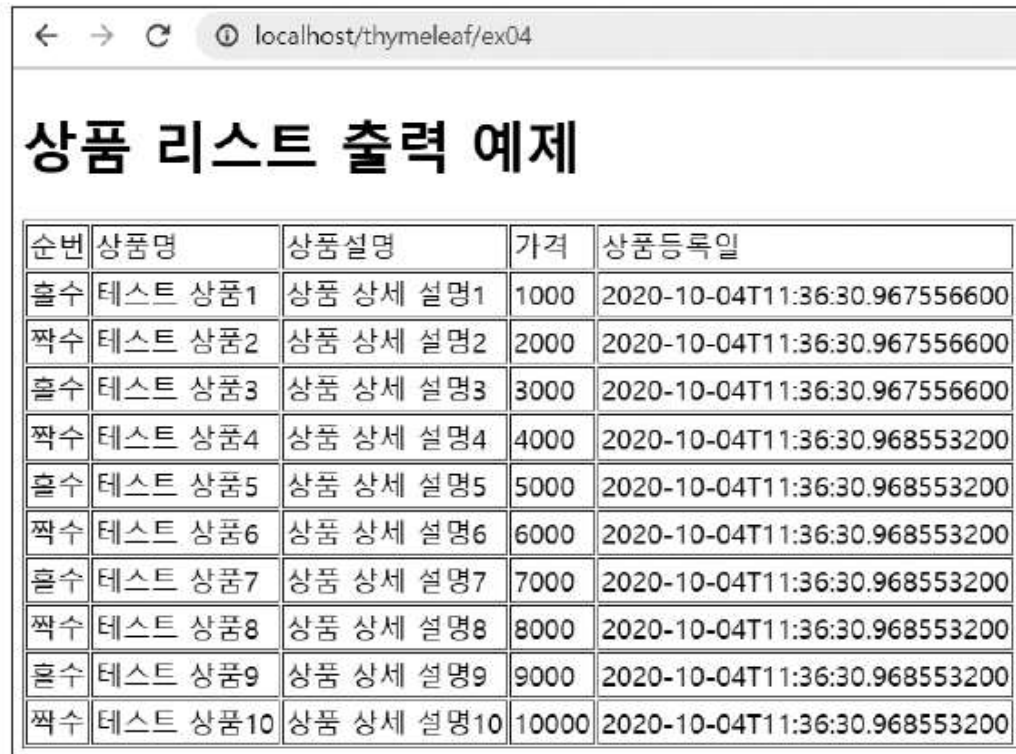
```
01 <!DOCTYPE html>
02 <html xmlns:th="http://www.thymeleaf.org">
03 <head>
04     <meta charset="UTF-8">
05     <title>Title</title>
06 </head>
07 <body>
08
09 <h1>상품 리스트 출력 예제</h1>
10
11 <table border="1">
12     <thead>
13     <tr>
14         <td>순번</td>
15         <td>상품명</td>
16         <td>상품설명</td>
17         <td>가격</td>
18         <td>상품등록일</td>
19     </tr>
20     </thead>
```

## 3.3 Thymeleaf 예제 진행 - th:switch, th:case

```
21 <tbody>
22 <tr th:each="itemDto, status: ${itemDtoList}">
23     <td th:switch="${status.even}"> ..... ❶
24         <span th:case=true>짝수</span>
25         <span th:case=false>홀수</span>
26     </td>
27     <td th:text="${itemDto.itemNm}"></td>
28     <td th:text="${itemDto.itemDetail}"></td>
29     <td th:text="${itemDto.price}"></td>
30     <td th:text="${itemDto.regTime}"></td>
31 </tr>
32 </tbody>
33 </table>
34
35 </body>
36 </html>
```

### 3.3 Thymeleaf 예제 진행 - th:switch, th:case

- 실행 결과는 th:if, th:unless를 사용했을 때와 동일



순번	상품명	상품설명	가격	상품등록일
홀수	테스트 상품1	상품 상세 설명1	1000	2020-10-04T11:36:30.967556600
짝수	테스트 상품2	상품 상세 설명2	2000	2020-10-04T11:36:30.967556600
홀수	테스트 상품3	상품 상세 설명3	3000	2020-10-04T11:36:30.967556600
짝수	테스트 상품4	상품 상세 설명4	4000	2020-10-04T11:36:30.968553200
홀수	테스트 상품5	상품 상세 설명5	5000	2020-10-04T11:36:30.968553200
짝수	테스트 상품6	상품 상세 설명6	6000	2020-10-04T11:36:30.968553200
홀수	테스트 상품7	상품 상세 설명7	7000	2020-10-04T11:36:30.968553200
짝수	테스트 상품8	상품 상세 설명8	8000	2020-10-04T11:36:30.968553200
홀수	테스트 상품9	상품 상세 설명9	9000	2020-10-04T11:36:30.968553200
짝수	테스트 상품10	상품 상세 설명10	10000	2020-10-04T11:36:30.968553200

[그림 3-15] 상품 리스트 출력 예제 실행 결과

### 3.3 Thymeleaf 예제 진행 - th:href

---

- th:href : Thymeleaf에서 링크를 처리하는 문법
- Absolute URL : 'http://' 또는 'https://'로 시작
- Context-relative URL : 가장 많이 사용되는 URL 형식이며 애플리케이션의 서버 내부를 이동하는 방법. 웹 애플리케이션 루트에 상대적인 URL을 입력

## 3.3 Thymeleaf 예제 진행 - th:href



[함께 해봐요 3-15] th:href를 이용한 링크 처리용 컨트롤러

com.shop.controller.ThymeleafExController.java

```
01 package com.shop.controller;
02
03 .....기존 импорт 생략.....
04
05 @Controller
06 @RequestMapping(value="/thymeleaf")
07 public class ThymeleafExController {
08
09     .....코드 생략.....
10
11     @GetMapping(value = "/ex05")
12     public String thymeleafExample05(){
13         return "thymeleafEx/thymeleafEx05";
14     }
15
16 }
```

## 3.3 Thymeleaf 예제 진행 - th:href



[함께 해봐요 3-16] th:href를 이용한 링크 처리용 thymeleaf 파일

resources/templates/thymeleafEx05.html

```
01 <!DOCTYPE html>
02 <html xmlns:th="http://www.thymeleaf.org">
03 <head>
04     <meta charset="UTF-8">
05     <title>Title</title>
06 </head>
07 <body>
08     <h1>Thymeleaf 링크처리 예제 페이지</h1>
09     <div>
10         <a th:href="@{/thymeleaf/ex01}">예제1 페이지 이동</a> ①
11     </div>
12     <div>
13         <a th:href="@{https://www.thymeleaf.org}">thymeleaf 공식 페이지 이동</a> ②
14     </div>
15 </body>
16 </html>
```

### 3.3 Thymeleaf 예제 진행 - th:href

---



[그림 3-16] Thymeleaf 링크 처리 예제 실행 결과

## 3.3 Thymeleaf 예제 진행 - th:href 파라미터 전달

- 링크 이동 시 파라미터를 전달해야하는 경우



[함께 해봐요 3-17] th:href를 이용한 파라미터 데이터 전달용 thymeleaf 파일

resources/templates/thymeleafEx05.html

```
01 <!DOCTYPE html>
02 <html xmlns:th="http://www.thymeleaf.org">
03 <head>
04     <meta charset="UTF-8">
05     <title>Title</title>
06 </head>
07 <body>
08     <h1>Thymeleaf 링크처리 예제 페이지</h1>
09     <div>
10         <a th:href="@{/thymeleaf/ex01}">예제1 페이지 이동</a>
11     </div>
12     <div>
13         <a th:href="@{https://www.thymeleaf.org/}">thymeleaf 공식 페이지 이동</a>
14     </div>
15     <div>
16         <a th:href="@{/thymeleaf/ex06(param1 = '파라미터 데이터1',
17                                     param2 = '파라미터 데이터2')}">thymeleaf 파라미터 전달</a> ❶
18     </div>
19 </body>
20 </html>
```



## 3.3 Thymeleaf 예제 진행 - th:href 파라미터 전달



[함께 해봐요 3-18] th:href를 이용한 파라미터 데이터 전달용 컨트롤러 작성하기

com.shop.controller.ThymeleafExController.java

```
01 package com.shop.controller;
02
03 .....기존 импорт 생략.....
04
05 @Controller
06 @RequestMapping(value="/thymeleaf")
07 public class ThymeleafExController {
08
09     .....코드 생략.....
10
11     @GetMapping(value = "/ex06")
12     public String thymeleafExample06(String param1, String param2, Model model){ ❶
13         model.addAttribute("param1", param1);
14         model.addAttribute("param2", param2);
15         return "thymeleafEx/thymeleafEx06";
16     }
17 }
```

## 3.3 Thymeleaf 예제 진행 - th:href 파라미터 전달



[함께 해봐요 3-19] th:href를 이용한 파라미터 데이터 전달용 thymeleaf 파일

resources/templates/thymeleafEx06.html

```
01 <!DOCTYPE html>
02 <html xmlns:th="http://www.thymeleaf.org">
03 <head>
04     <meta charset="UTF-8">
05     <title>Title</title>
06 </head>
07 <body>
08     <h1>파라미터 전달 예제</h1>
09     <div th:text="${param1}"></div>
10     <div th:text="${param2}"></div>
11 </body>
12 </html>
```

1

2

### 3.3 Thymeleaf 예제 진행 - th:href 파라미터 전달

---



[그림 3-17] Thymeleaf 링크 처리 예제 실행 결과



[그림 3-18] Thymeleaf 파라미터 데이터 전달 예제 실행 결과

## 3.4 Thymeleaf 페이지 레이아웃

---

- header, footer, menu 등 공통적인 페이지 구성 요소들을 페이지 레이아웃 기능을 통해서 1개로 관리

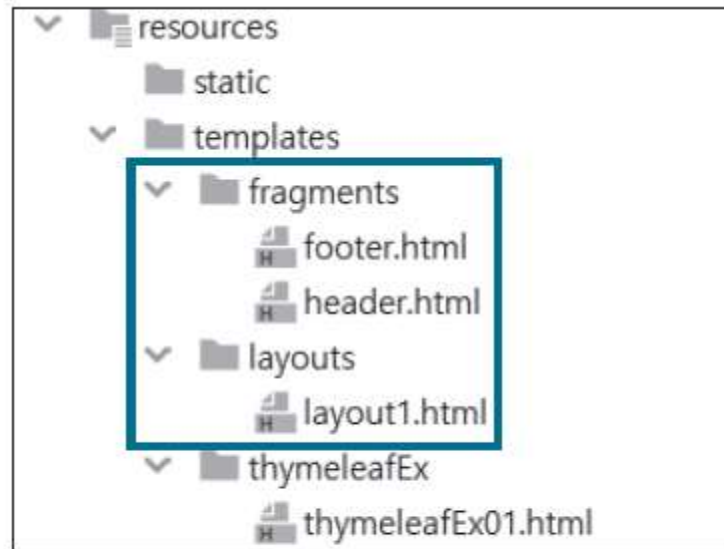


[함께 해봐요 3-20] pom.xml에 Thymeleaf Layout Dialect 의존성 추가하기

```
01 <dependency>
02     <groupId>nz.net.ultraq.thymeleaf</groupId>
03     <artifactId>thymeleaf-layout-dialect</artifactId>
04     <version>2.5.1</version>
05 </dependency>
```

## 3.4 Thymeleaf 페이지 레이아웃

---



[그림 3-19] layouts, fragments 폴더 및 footer, header, layout1.html 파일 생성

## 3.4 Thymeleaf 페이지 레이아웃



[함께 해봐요 3-21] Thymeleaf 페이지 레이아웃 예제: 푸터 만들기

resources/templates/fragments/footer.html

```
01 <!DOCTYPE html>
02 <html xmlns:th="http://www.thymeleaf.org">
03
04     <div th:fragment="footer"> ..... 1
05         footer 영역입니다.
06     </div>
07
08 </html>
```



[함께 해봐요 3-22] Thymeleaf 페이지 레이아웃 예제: 헤더 만들기

resources/templates/fragments/header.html

```
01 <!DOCTYPE html>
02 <html xmlns:th="http://www.thymeleaf.org">
03
04     <div th:fragment="header"> ..... 1
05         header 영역입니다.
06     </div>
07
08 </html>
```

## 3.4 Thymeleaf 페이지 레이아웃



[함께 해봐요 3-23] Thymeleaf 페이지 레이아웃 예제: 본문 레이아웃

resources/templates/layouts/layout1.html

```
01 <!DOCTYPE html>
02 <html xmlns:th="http://www.thymeleaf.org"
03       xmlns:layout="http://www.ultraq.net.nz/thymeleaf/layout"> ❶
04 <head>
05     <meta charset="UTF-8">
06     <title>Title</title>
07
08     <th:block layout:fragment="script"></th:block>
09     <th:block layout:fragment="css"></th:block>
10
11 </head>
12 <body>
13
14     <div th:replace="fragments/header::header"></div> ❷
15
16     <div layout:fragment="content"> ❸
17
18     </div>
19
20     <div th:replace="fragments/footer::footer"></div> ❹
21
22 </body>
23 </html>
```

## 3.4 Thymeleaf 페이지 레이아웃



[함께 해봐요 3-24] Thymeleaf 페이지 레이아웃 예제: thymeleaf 파일 만들기

resources/templates/thymeleafEx07.html

```
01 <!DOCTYPE html>
02 <html xmlns:th="http://www.thymeleaf.org"
03     xmlns:layout="http://www.ultraq.net.nz/thymeleaf/layout
04     layout:decorate="~{layouts/layout1}"> ..... 1
05
06     <div layout:fragment="content"> ..... 2
07         본문 영역 입니다.
08     </div>
09
10 </html>
```



## 3.4 Thymeleaf 페이지 레이아웃



[함께 해봐요 3-25] Thymeleaf 페이지 레이아웃 예제: 컨트롤러 클래스 작성하기

com.shop.controller.ThymeleafExController.java

```
01 package com.shop.controller;
02
03 .....기존 импорт 생략.....
04
05 @Controller
06 @RequestMapping(value="/thymeleaf")
07 public class ThymeleafExController {
08
09     .....코드 생략.....
10
11     @GetMapping(value = "/ex07")
12     public String thymeleafExample07(){
13         return "thymeleafEx/thymeleafEx07";
14     }
15
16 }
```

## 3.4 Thymeleaf 페이지 레이아웃

---



[그림 3-20] Thymeleaf layout 예제 실행 결과

## 3.5 부트스트랩으로 header, footer 영역 수정

---

- 부트스트랩Bootstrap은 웹사이트를 쉽게 만들 수 있게 도와주는 HTML, CSS, JS 프레임워크
- 부트스트랩에서 제공하는 템플릿 및 컴포넌트 등을 이용하면 웹 페이지를 쉽게 꾸밀 수 있음

## 3.5 부트스트랩으로 header, footer 영역 수정

---

- Bootstrap CDN 추가
- CDN(Contents Delivery Network)는 물리적으로 멀리 떨어져 있는 사용자에게 콘텐츠를 좀 더 빠르게 제공하기 위한 서비스
- Bootstrap CDN을 layout1.html의 헤더 영역에 추가하여 해당 리소스를 다운로드해서 사용

## 3.5 부트스트랩으로 header, footer 영역 수정



[함께 해봐요 3-26] 레이아웃에 Bootstrap CDN 추가하기

resources/templates/layouts/layout1.html

```
01 <!DOCTYPE html>
02 <html xmlns:th="http://www.thymeleaf.org"
03     xmlns:layout="http://www.ultraq.net.nz/thymeleaf/layout">
04 <head>
05     <meta charset="UTF-8">
06     <title>Title</title>
07
08     <!-- CSS only -->
09     <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/
10         bootstrap/4.5.2/css/bootstrap.min.css">
11
12     <!-- JS, Popper.js, and jQuery -->
13     <script src="https://code.jquery.com/jquery-3.5.1.min.js"></script>
14     <script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.1/dist
15         /umd/popper.min.js"></script>
16     <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/js
17         /bootstrap.min.js"></script>
18
19     <th:block layout:fragment="script"></th:block>
20     <th:block layout:fragment="css"></th:block>
21
22 </head>
```

## 3.5 부트스트랩으로 header, footer 영역 수정

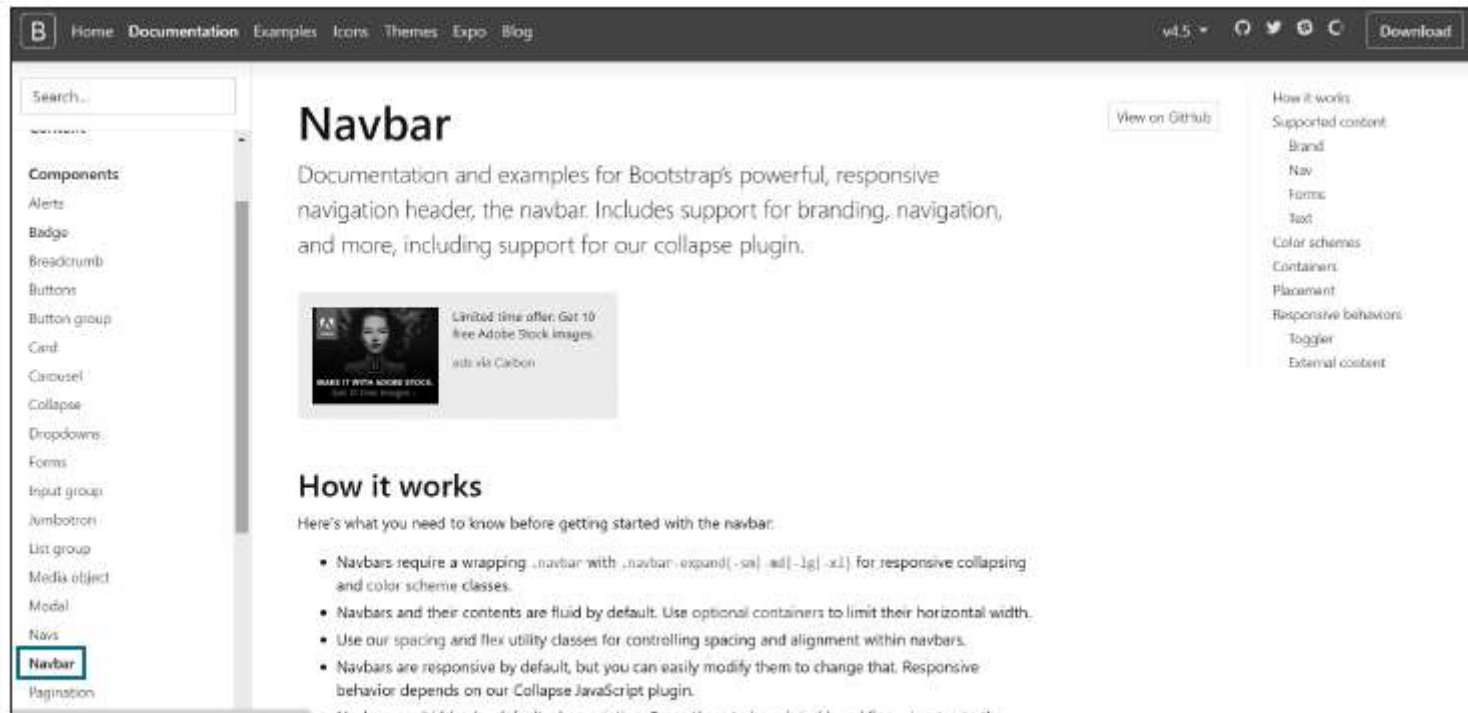
---

```
20 <body>
21
22     <div th:replace="fragments/header::header"></div>
23
24     <div layout:fragment="content">
25
26     </div>
27
28     <div th:replace="fragments/footer::footer"></div>
29
30 </body>
31 </html>
```

---

## 3.5 부트스트랩으로 header, footer 영역 수정

- 'https://getbootstrap.com/' 방문 후 [Documentation] -> [Components] 탭에서 부트스트랩에서 미리 만들어둔 컴포넌트를 사용가능



[그림 3-21] Bootstrap Navbar Component

## 3.5 부트스트랩으로 header, footer 영역 수정



[함께 해봐요 3-27] 헤더 영역에 Navbar 추가하기

resources/templates/fragments/header.html

```
01 <!DOCTYPE html>
02 <html xmlns:th=http://www.thymeleaf.org>
03
04     <div th:fragment="header">
05         <nav class="navbar navbar-expand-sm bg-primary navbar-dark">
06             <button class="navbar-toggler" type="button" data-toggle="collapse"
07                 data-target="#navbarTogglerDemo03"
08                 aria-controls="navbarTogglerDemo03"
09                 aria-expanded="false" aria-label="Toggle navigation">
10                 <span class="navbar-toggler-icon"></span>
11             </button>
12             <a class="navbar-brand" href="/">Shop</a>
13
14             <div class="collapse navbar-collapse" id="navbarTogglerDemo03">
15                 <ul class="navbar-nav mr-auto mt-2 mt-lg-0">
16                     <li class="nav-item">
17                         <a class="nav-link" href="/admin/item/new">상품 등록</a>
18                     </li>
19                     <li class="nav-item">
20                         <a class="nav-link" href="/admin/items">상품 관리</a>
21                     </li>
22                     <li class="nav-item">
23                         <a class="nav-link" href="/cart">장바구니</a>
24                     </li>
25                 </ul>
26             </div>
27         </nav>
28     </div>
```



## 3.5 부트스트랩으로 header, footer 영역 수정

```
24         <li class="nav-item">
25             <a class="nav-link" href="/orders">구매이력</a>
26         </li>
27         <li class="nav-item">
28             <a class="nav-link" href="/members/login">로그인</a>
29         </li>
30         <li class="nav-item">
31             <a class="nav-link" href="/members/logout">로그아웃</a>
32         </li>
33     </ul>
34     <form class="form-inline my-2 my-lg-0" th:action="@{/}"
35           method="get">
36         <input name="searchQuery" class="form-control mr-sm-2"
37               type="search" placeholder="Search" aria-label="Search">
38         <button class="btn btn-outline-success my-2 my-sm-0"
39               type="submit">Search</button>
40     </form>
41 </div>
42 </nav>
43 </div>
44 </html>
```

## 3.5 부트스트랩으로 header, footer 영역 수정

---



[그림 3-23] Navigation bar가 추가된 결과

## 3.5 부트스트랩으로 header, footer 영역 수정



[함께 해봐요 3-28] 푸터 영역 수정하기

resources/templates/fragments/footer.html

```
01 <!DOCTYPE html>
02 <html xmlns:th="http://www.thymeleaf.org">
03
04     <div class="footer" th:fragment="footer">
05         <footer class="page-footer font-small cyan darken-3">
06             <div class="footer-copyright text-center py-3">
07                 2020 Shopping Mall Example WebSite
08             </div>
09         </footer>
10     </div>
11
12 </html>
```

## 3.5 부트스트랩으로 header, footer 영역 수정



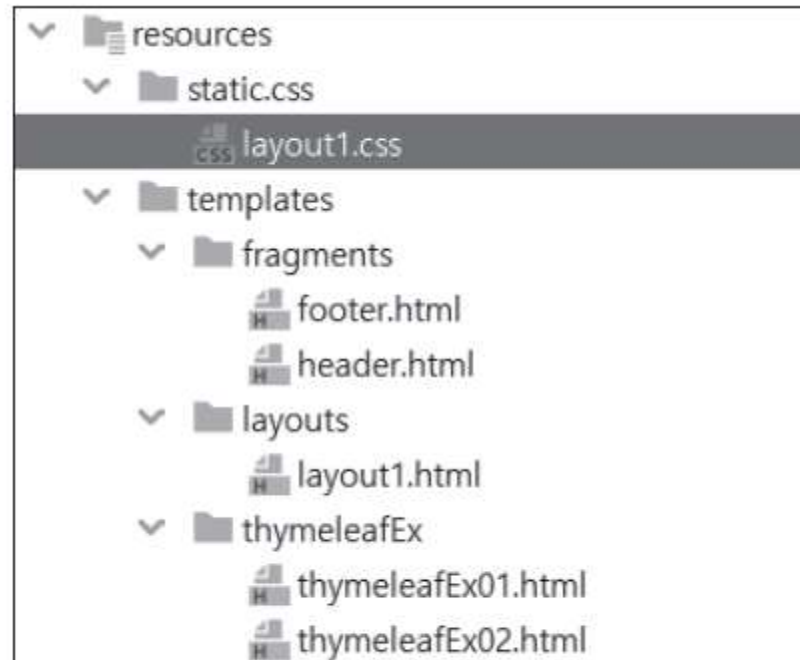
[함께 해봐요 3-28] 푸터 영역 수정하기

resources/templates/fragments/footer.html

```
01 <!DOCTYPE html>
02 <html xmlns:th="http://www.thymeleaf.org">
03
04     <div class="footer" th:fragment="footer">
05         <footer class="page-footer font-small cyan darken-3">
06             <div class="footer-copyright text-center py-3">
07                 2020 Shopping Mall Example WebSite
08             </div>
09         </footer>
10     </div>
11
12 </html>
```

## 3.5 부트스트랩으로 header, footer 영역 수정

---



[그림 3-25] static 폴더 및 css폴더 생성

## 3.5 부트스트랩으로 header, footer 영역 수정



[함께 해봐요 3-29] CSS 적용하기

resources/static/css/layout1.css

```
01 html {  
02     position: relative;  
03     min-height: 100%;  
04     margin: 0;  
05 }  
06 body {  
07     min-height: 100%;  
08 }  
09 .footer {  
10     position: absolute;  
11     left: 0;  
12     right: 0;  
13     bottom: 0;  
  
14     width: 100%;  
15     padding: 15px 0;  
16     text-align: center;  
17 }  
18 .content{  
19     margin-bottom: 100px;  
20     margin-top: 50px;  
21     margin-left: 200px;  
22     margin-right: 200px;  
23 }
```

## 3.5 부트스트랩으로 header, footer 영역 수정



[함께 해봐요 3-30] CSS와 HTML 파일 연결하기

resources/templates/layouts/layout1.html

```
<head>
  <meta charset="UTF-8">
  <title>Title</title>

  <!-- CSS only -->
  <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css" integrity="sha384-JSGKf1Q3gHSPQx24Q0fxTN6R4QC1P9dHz0RxaaDUuIVQahaIwPZhsd0L6adaUQ" crossorigin="anonymous">
  <link th:href="@{/css/layout1.css}" rel="stylesheet">

  <!-- JS, Popper.js, and jQuery -->
  <script src="https://code.jquery.com/jquery-3.5.1.min.js"></script>
  <script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.1/dist/umd/popper.min.js"></script>
  <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>

  <th:block layout:fragment="script"></th:block>
  <th:block layout:fragment="css"></th:block>

</head>
<body>

  <div th:replace="fragments/header::header"></div>

  <div layout:fragment="content" class="content">

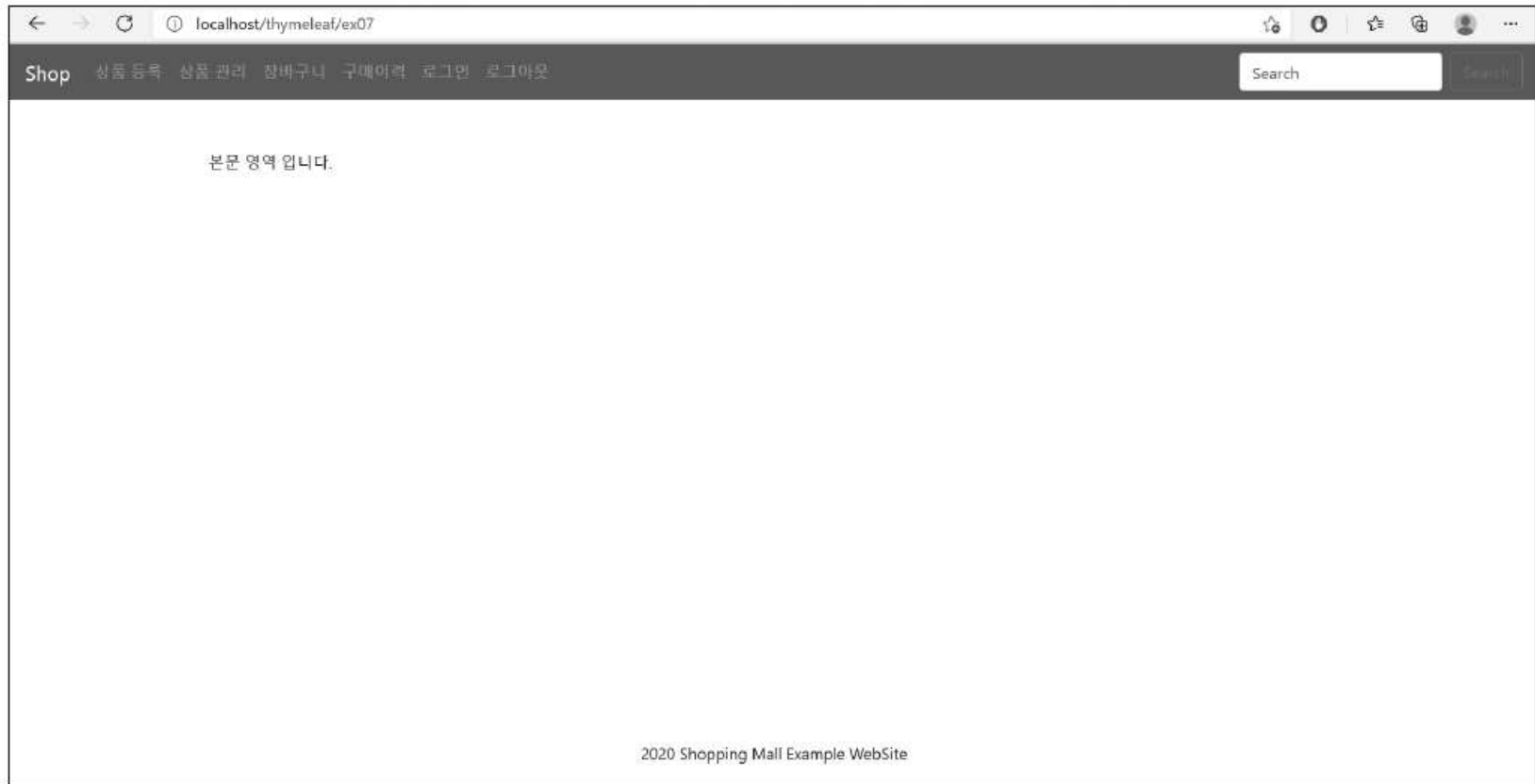
  </div>

  <div th:replace="fragments/footer::footer"></div>
```

[그림 3-26] CSS 적용하기

## 3.5 부트스트랩으로 header, footer 영역 수정

---



[그림 3-27] CSS 적용 결과



# Thank you for your attention

---

© 2021. 변구훈 & 로드북 all rights reserved.

이 콘텐츠의 저작권은 조휘용과 로드북에 있습니다.  
재배포가 가능하지만 저작권자 표시 및 콘텐츠 시작 부분에 나오는 표지를 반드시 실어야 합니다.  
수정하여 재배포할 시에는 수정한 부분을 반드시 명시해야 합니다.