



## <5장> 요청 처리 메서드의 파라미터 유형

[도서 쇼핑몰] 도서 상세 정보 표시하기

# 학습 목표

- 다양한 요청 조건을 포함하는 요청 처리 메서드에 대해 알아봅니다.
- 요청 처리 메서드의 파라미터에 사용하는 애너테이션을 살펴봅니다.
- [도서 쇼핑몰] 도서 목록에서 선택한 도서의 상세 정보를 출력하고 다양한 조건으로 검색한 도서 목록을 출력하는 방법을 알아봅니다.

# 목차

5.1 요청 파라미터와 `@RequestParam`

5.2 경로 변수와 `@PathVariable`

5.3 매트릭스 변수와 `@MatrixVariable`

5.4 [도서 쇼핑몰] 도서 상세 정보 표시하기

01

# 요청 파라미터와 @RequestParam

# 1. 요청 파라미터와 @RequestParam

요청 처리 메서드는 웹에서 들어온 요청 URL에 포함된 파라미터(쿼리문) 값을 전달받아 처리합니다.

## ■ 요청 파라미터

- 요청 파라미터(request parameters)는 일반적인 웹 서버 애플리케이션에서 GET 방식의 쿼리문을 '이름=값(name=value)' 형태의 데이터로 전송함. 웹 요청 URL에 다중 쿼리문을 가지면 color:red&year=2024처럼 &로 구분하여 표현함.

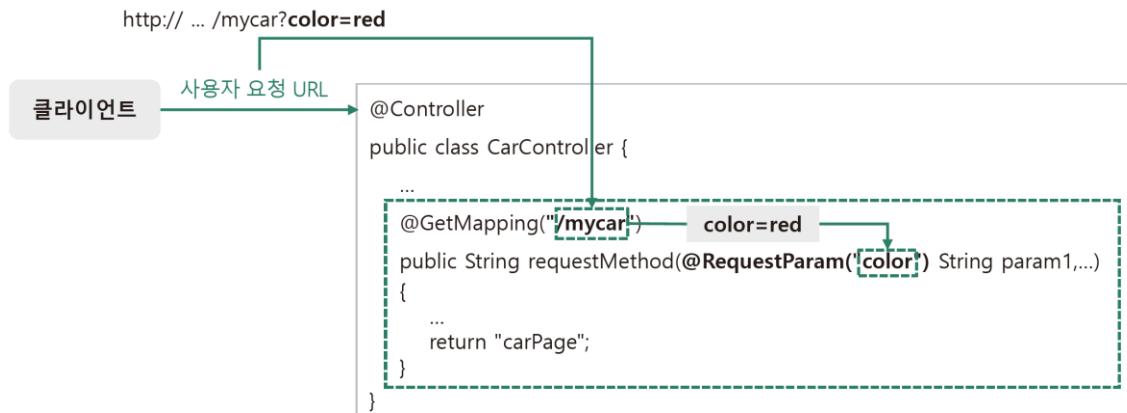


그림 5-1 웹 요청 URL에 포함된 요청 파라미터와 @RequestParam의 매핑

- `@GetMapping`에 설정된 요청 맵핑 경로에 쿼리문이 포함되면 요청 처리 메서드의 매개변수에 `@RequestParam`을 이용해 요청 파라미터 값을 전달받음.
- `@RequestParam`은 컨트롤러의 요청 처리 메서드를 구현할 때 가장 많이 사용하는 애너테이션으로, 요청 파라미터의 수가 많지 않을 때 주로 사용함.

# 1. 요청 파라미터와 @RequestParam

## ■ @RequestParam으로 요청 파라미터 처리

- 일반적으로 @RequestParam은 메서드의 매개변수에 설정하는데, 기본값은 웹 요청 URL로 전송되는 요청 파라미터 이름과 똑같이 설정함. 요청 파라미터의 값은 메서드 매개변수의 데이터 타입에 따라 적절하게 변환됨.
- @RequestParam은 @RequestMapping의 요청 매팅 경로에 포함된 요청 파라미터 값을 요청 처리 메서드의 매개변수로 전달받음.
- @RequestParam의 사용 형식

```
@RequestMapping("요청_경로")
public String 메서드_이름(@RequestParam("요청_파라미터"), ...) {
    ...
}
```

```
@RequestMapping("요청_경로")
public String 메서드_이름(@RequestParam(value="요청_파라미터") 매개변수, ...) {
    ...
}
```

# 1. 요청 파라미터와 @RequestParam

## ■ @RequestParam으로 요청 파라미터 처리

표 5-1 @RequestParam 애너테이션의 속성

속성	타입	설명
defaultValue	String	요청 파라미터가 없으면 기본값으로 대체하여 사용
name	String	요청 파라미터의 이름
required	boolean	요청 파라미터의 필수 여부를 설정
value	String	name의 별칭

# 1. 요청 파라미터와 @RequestParam

요청 처리 메서드의 매개변수에 `@RequestParam`을 이용하여 웹 요청 URL에서 전송되는 요청 파라미터에 접근하는 방식을 예로 살펴보자.

## ■ `@RequestParam`으로 요청 파라미터 처리

[`@RequestParam`을 이용해 요청 파라미터에 접근한 예]

```
Example01Controller.java

package com.springboot.controller;
...
import org.springframework.web.bind.annotation.RequestParam;

@Controller
public class Example01Controller {
    @GetMapping("exam01") ①
    public String requestMethod(@RequestParam("id") String userId, ②
        @RequestParam("passwd") String userPw, Model model) {
        model.addAttribute("data1", "@RequestParam 예제");
        model.addAttribute("data2", "요청 파라미터 id 값 : " + userId + "<br>요청 파라미터 passwd 값" + userPw);
        return "viewPage";
    }
}
```

- ① 웹 요청 URL이 `http://.../exam01?id=admin&passwd=admin1234`이면 요청 파라미터는 ? 다음에 전달되는 id와 passwd가 됨.
- ② `@RequestParam`에 설정된 매개변수 `userId`는 요청 파라미터 `id`에 전달된 값인 `admin`, `userPw`는 요청 파라미터 `passwd`에 전달된 값인 `admin12` 전달받음.

# 1. 요청 파라미터와 @RequestParam

## ■ @RequestParam으로 요청 파라미터 처리

- 웹 브라우저의 응답 페이지로 출력되는 뷰 페이지인 `viewPage.html`

```
viewPage.html

<html>
<head>
    <title>chap05</title>
</head>
<body>
    1 <h3 th:text="${data1}">/>
    <p th:utext="${data2}">/>
2
</body>
</html>
```

- ① 컨트롤러에서 전달된 모델 속성 이름 `data1`에 저장된 값 출력함.
- ② 컨트롤러에서 전달된 모델 속성 이름 `data2`에 저장된 값 출력하는데 HTML 태그가 적용되어 출력됨.

# 1. 요청 파라미터와 @RequestParam

## 타임리브(Thymeleaf)의 th:text와 th:utext의 차이점

**th:text** : HTML 태그 내에 데이터를 출력할 때 사용하며 HTML, 태그 대신 데이터를 직접 출력하고 싶으면 [[...]]를 사용함. 출력하려는 데이터에 HTML 태그가 포함되어 있으면 특수문자인 이스케이프(escape) 형식으로 웹 브라우저에 출력됨.

**th:utext** : th:text 와 같이 HTML 태그 내에 데이터를 출력할 때 사용하며 데이터를 직접 출력하고 싶으면 [((...))]를 사용함. 출력하려는 데이터에 HTML, 태그가 포함되어 있으면 웹 브라우저에 HTML 태그가 적용되어 출력됨.

# 1. 요청 파라미터와 @RequestParam

## ■ @RequestParam으로 요청 파라미터 처리

- 사용자의 웹 요청

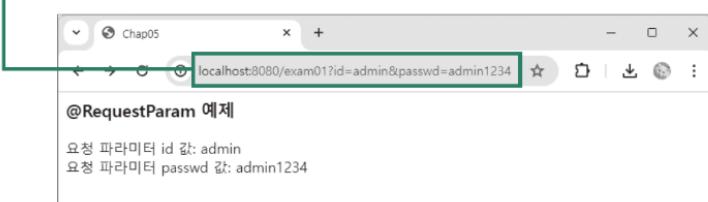
URL(<http://localhost:8080/exam01?id=admin&passwd=admin1234>)에 대한 응답으로 `viewPage` 페이지를 웹 브라우저에 출력하는 과정

```
Example01Controller.java

package com.springboot.controller;
...
import org.springframework.web.bind.annotation.RequestParam;

@Controller
public class Example01Controller {
    @GetMapping("/exam01")
    public String requestMethod(@RequestParam("id") String userId,
                               @RequestParam("passwd") String userPw, Model model) {
        model.addAttribute("data1", "@RequestParam 예제");
        model.addAttribute("data2", "요청 파라미터 id 값 : " + userId + "<br>요청 파라미터 passwd 값" + userPw);
        return "viewPage";
    }
}
```

<http://.../exam01?id=admin&passwd=admin1234>



# 1. 요청 파라미터와 @RequestParam

## ■ @RequestParam으로 요청 파라미터 처리

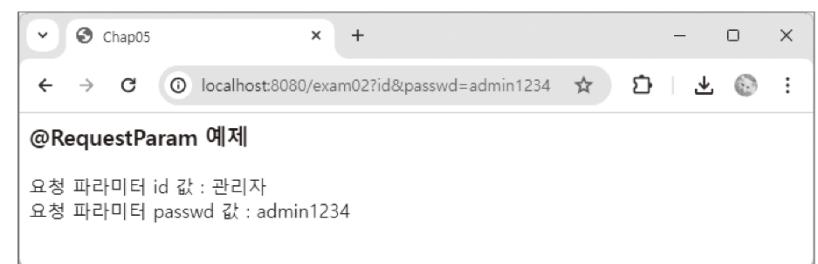
- @RequestParam의 defaultValue 속성을 사용하는 예로, 웹 요청 URL에 요청 파라미터 값이 없으면 기본값으로 설정하는 경우.
- 사용자의 웹 요청 URL이 `http://.../exam02?id&passwd=admin1234`처럼 요청 파라미터의 id 값이 null이면 빈 문자열로 처리.
- `requestMethod()` 메서드에 `defaultValue="관리자"`와 같이 설정되어 있다면 userId는 기본값으로 '관리자'를, userPw 값은 기본값으로 admin1234를 전달받음.

[@RequestParam을 이용해 요청 파라미터에 접근한 예]

```
Example02Controller.java

package com.springboot.controller;
...
import org.springframework.web.bind.annotation.RequestParam;

@Controller
public class Example02Controller {
    @GetMapping("/exam02")
    public String requestMethod(@RequestParam(value="id", defaultValue="관리자") String
        userId, @RequestParam("passwd") String userPw, Model model) {
        model.addAttribute("data1", "@RequestParam 예제");
        model.addAttribute("data2", "요청 파라미터 id 값 : " + userId + "<br>요청 파라미터
            passwd 값" + userPw);
        return "viewPage";
    }
}
```



# 1. 요청 파라미터와 @RequestParam

## ■ @RequestParam으로 요청 파라미터 처리

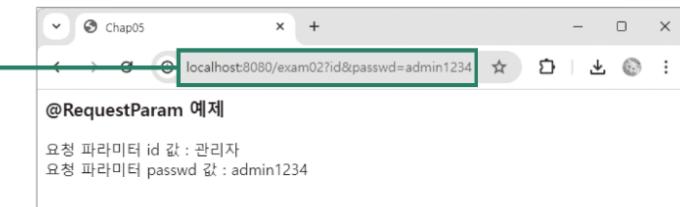
- 사용자의 웹 요청 URL(`http://localhost:8080/exam02?id&passwd=admin1234`)에 대한 응답으로 `viewPage` 페이지를 웹 브라우저에 출력하는 과정

```
Example02Controller.java
```

```
package com.springboot.controller;
...
import org.springframework.web.bind.annotation.RequestParam;

@Controller
public class Example02Controller {
    @GetMapping("/exam02")
    public String requestMethod(@RequestParam(value="id", defaultValue="관리자") String userId,
                               @RequestParam("passwd") String userPw, Model model) {
        model.addAttribute("data1", "@RequestParam 예제");
        model.addAttribute("data2", "요청 파라미터 id 값 : " + userId + "<br>요청 파라미터 passwd 값" + userPw);
        return "viewPage";
    }
}
```

`http://.../exam02?id&passwd=admin1234`



02

경로 변수와 @PathVariable

## 2. 경로 변수와 @PathVariable

웹 요청 URL에 포함된 파라미터 값을 전달받은 경로 변수와 이를 처리하는 요청 처리 메서드의 매개변수에 선언하는 `@PathVariable` 어노테이션에 대해 살펴보자.

### ■ 경로 변수

- 경로 변수(path variables)는 웹 요청 URL에 포함된 파라미터 값을 전달받을 때 사용하는 변수. 매팅 경로 설정하는 `@RequestMapping`(단순화한 `@RequestMapping` 포함)에 중괄호({})를 사용해 웹 요청 URL에 포함된 요청 조건 값을 전달받음. 중괄호 안에 명시된 것이 경로 변수이며, 하나 또는 그 이상의 경로 변수를 포함할 수 있음.

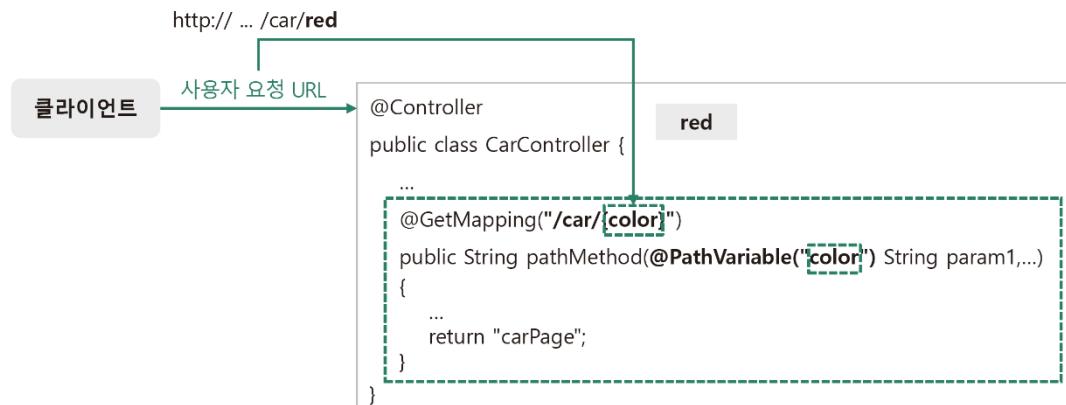


그림 5-2 웹 요청 URL에 포함된 경로 변수와 `@PathVariable`의 매팅

- `@GetMapping`에 설정된 요청 매팅 경로에서 중괄호 안에 명시된 color가 경로 변수에 해당함. 웹 요청 URL이 `http://localhost:8080/car/red`라면 경로 변수 color는 red를 값으로 전달받음.
- `@RequestMapping`에 설정된 요청 매팅 경로가 경로 변수 포함하면 `@PathVariable`로 요청 처리 메서드의 파라미터에 경로 변수 값 전달받을 수 있음.

## 2. 경로 변수와 @PathVariable

### ■ @PathVariable로 경로 변수 처리

- @PathVariable은 @RequestMapping에 설정된 경로 변수 값을 요청 처리 메서드의 매개변수로 전달받음.

```
@RequestMapping(.../{"경로_변수"})
public String 메서드_이름(@PathVariable("경로_변수") 매개변수,...) {
    ...
}

@RequestMapping(.../{"경로_변수"})
public String 메서드_이름(@PathVariable(value="경로_변수") 매개변수,...) {
    ...
}
```

- 사용자의 웹 요청 URL에 포함된 경로 변수 값은 요청 처리 메서드의 매개변수와 데이터 타입(data type)이 일치해야 함. 메서드의 매개변수가 int형이면 해당 경로 변수의 값은 숫자여야 함.

## 2. 경로 변수와 @PathVariable

### ■ @PathVariable로 경로 변수 처리

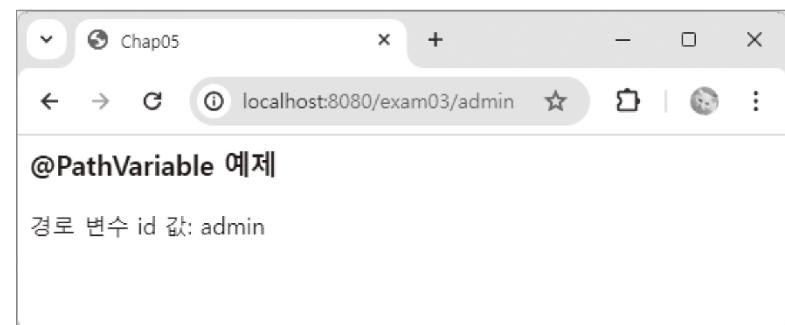
- @PathVariable을 이용하여 경로 변수에 접근하는 예

[@PathVariable을 이용해 경로 변수에 접근한 예]

```
Example03Controller.java

package com.springboot.controller;
...
import org.springframework.web.bind.annotation.PathVariable;

@Controller
public class Example03Controller {
    @GetMapping("/exam03/{id}") ①
    public String requestMethod(@PathVariable("id") String userId, Model model) {
        model.addAttribute("data1", "@PathVariable 예제");
        model.addAttribute("data2", "경로 변수 id 값: " + userId);
        return "viewPage";
    }
}
```



- ① id는 경로 변수로 웹 요청 URL이 http://.../exam03/admin이면 웹 요청 URL로 전송된 파라미터 값 admin을 전달받음.
- ② @PathVariable에 설정된 매개변수 userId는 경로 변수 id에 전달된 값인 admin을 전달받음

## 2. 경로 변수와 @PathVariable

### ■ @PathVariable로 경로 변수 처리

- 사용자의 웹 요청 URL(`http://localhost:8080/exam03/admin`)에 대한 응답으로 `viewPage` 페이지를 웹 브라우저에 출력하는 과정

Example03Controller.java

```
package com.springboot.controller;
...
import org.springframework.web.bind.annotation.PathVariable;

@Controller
public class Example03Controller {
    @GetMapping("/exam03/{id}")
    public String RequestMethod(@PathVariable("id") String userId, Model model) {
        model.addAttribute("data1", "@PathVariable 예제");
        model.addAttribute("data2", "경로 변수 id 값: " + userId);
        return "viewPage";
    }
}
```

http://.../exam03/**admin**

The diagram illustrates the mapping between the URL path and the controller method. A green box highlights the path variable '{id}' in the URL. An arrow points from this box to the corresponding annotation '@PathVariable("id")' in the code. Another arrow points from the code back to the browser window, which displays the resulting page content.

## 2. 경로 변수와 @PathVariable

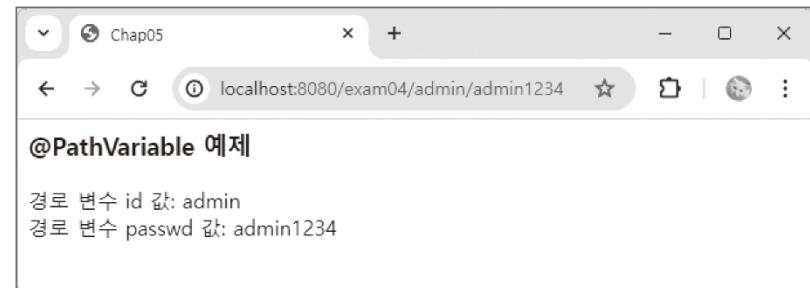
### ■ @PathVariable로 경로 변수 처리

[@PathVariable을 이용해 다중 경로 변수에 접근한 예]

```
Example04Controller.java

package com.springboot.controller;
...
import org.springframework.web.bind.annotation.PathVariable;

@Controller
public class Example04Controller {
    @GetMapping("/exam04/{id}/{passwd}") ①
    public String RequestMethod(@PathVariable("id") String userId,
        ③ @PathVariable("passwd") String userPw, Model model) {
        model.addAttribute("data1", "@PathVariable 예제");
        model.addAttribute("data2", "경로 변수 id 값: " + userId + "<br> 경로 변수 passwd
            값: " + userPw);
        return "viewPage";
    }
}
```



- ① id와 passwd는 경로 변수로, 웹 요청 URL이 http://.../exam04/admin/admin1234 이면 파라미터 값인 admin과 admin1234를 전달받음.
- ② @PathVariable에 설정된 매개변수 userId와 userPw는 경로 변수 id와 passwd에 전달된 값인 admin과 admin1234를 전달받음(3번 또한 같음)

## 2. 경로 변수와 @PathVariable

### ■ @PathVariable로 경로 변수 처리

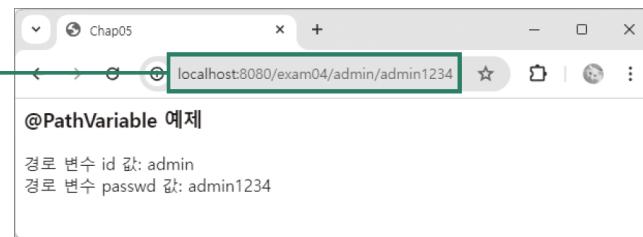
- 사용자의 웹 요청 URL(`http://localhost:8080/exam04/admin/admin1234`)에 대한 응답으로 `viewPage` 페이지를 웹 브라우저에 출력하는 과정

```
Example04Controller.java

package com.springboot.controller;
..
import org.springframework.web.bind.annotation.PathVariable;

@Controller
public class Example04Controller {
    @GetMapping("/exam04/{id}/{passwd}")
    public String requestMethod @PathVariable("id") String userId,
                                @PathVariable("passwd") String userPw, Model model) {
        model.addAttribute("data1", "@ PathVariable 예제");
        model.addAttribute("data2", "경로 변수 id 값: " + userId + "<br> 경로 변수 passwd 값: " + userPw);
        return "viewPage";
    }
}
```

`http://.../exam04/admin/admin1234`



03

# 매트릭스 변수와 @MatrixVariable

### 3. 매트릭스 변수와 @MatrixVariable

웹 요청 URL에 포함된 다중 파라미터 값을 전달받은 매트릭스 변수와 이를 처리하는 요청 처리 메소드의 매개변수에 선언하는 `@MatrixVariable` 애너테이션에 대해 살펴보자.

#### ■ 매트릭스 변수

- 매트릭스 변수(matrix variables)는 경로 변수와 마찬가지로 웹 요청 URL에 포함된 파라미터 값을 전달받는 데 사용되지만, 세미콜론(:)으로 구분해 다중 데이터를 전달받는 것이 다름. `@RequestMapping`의 경로 변수에 '매트릭스 변수=값' 형태로 사용하며, 매트릭스 변수가 여러 개이면 `color:red,green,blue`처럼 콤마(,)로 구분하거나 `color:red;color:green;color:blue`처럼 변수 이름을 반복해서 사용.

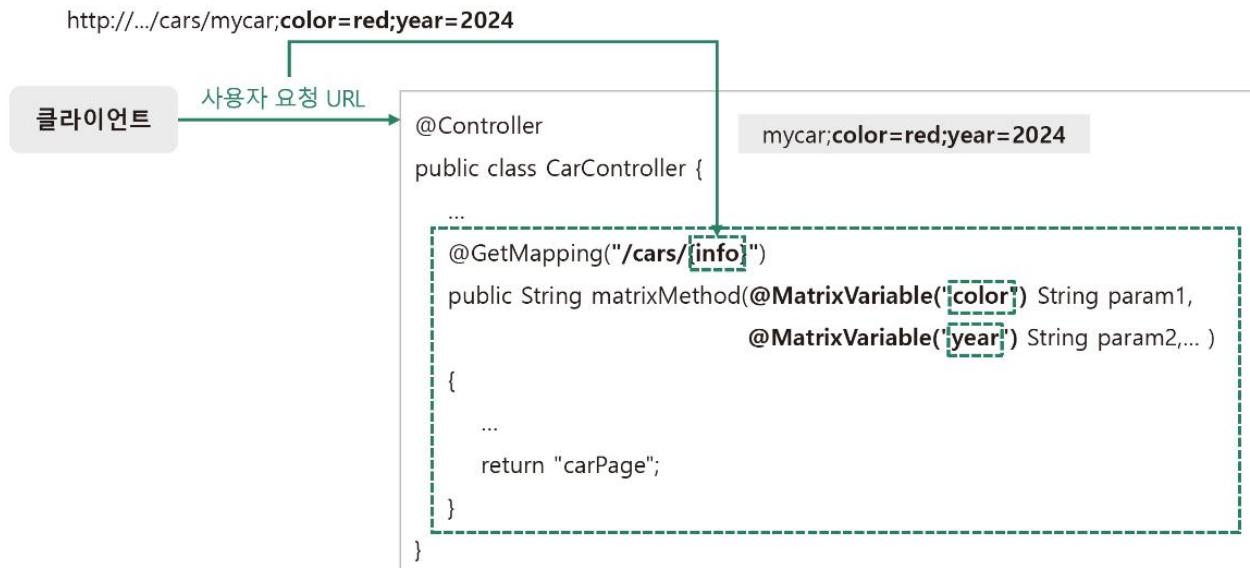


그림 5-3 웹 요청 URL에 포함된 매트릭스 변수와 `@MatrixVariable`의 매핑

- `@RequestMapping`에 설정된 요청 매팅 경로가 경로 변수를 포함하면 `@PathVariable`로 요청 처리 메서드의 파라미터에 경로 변수 값을 전달받을 수 있음.

### 3. 매트릭스 변수와 @MatrixVariable

#### ■ @MatrixVariable로 매트릭스 변수 처리

- @MatrixVariable은 @RequestMapping에 설정된 경로 변수에 포함된 매트릭스 변수 값을 요청 처리 메서드의 매개변수로 전달받음.
- @MatrixVariable의 사용 형식

```
@RequestMapping("{경로_변수}")
public String 메서드_이름(@MatrixVariable("매트릭스_변수") 매개변수,...) {
    ...
}

@RequestMapping("{경로_변수}")
public String 메서드_이름(@MatrixVariable(value="매트릭스_변수") 매개변수,...) {
    ...
}
```

### 3. 매트릭스 변수와 @MatrixVariable

#### ■ @MatrixVariable로 매트릭스 변수 처리

표 5-2 @MatrixVariable 애너테이션의 속성

속성	타입	설명
defaultValue	String	기본값으로 대체하여 사용
name	String	매트릭스 변수의 이름
pathVar	String	매트릭스 변수가 있는 URI 경로 변수의 이름으로, 하나 이상의 경로 구분(/)에서 동일한 이름을 명확하게 구분할 때 사용
required	boolean	매트릭스 변수의 필요 여부 설정
value	String	매트릭스 변수인 name=value에서 name을 가리킴

### 3. 매트릭스 변수와 @MatrixVariable

#### ■ @MatrixVariable로 매트릭스 변수 처리

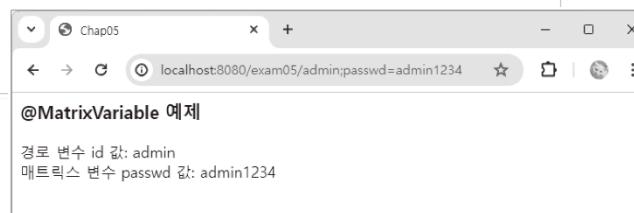
- @MatrixVariable을 이용해 매트릭스 변수에 접근하는 예
- 웹 요청 URL에 포함된 파라미터 값을 경로 변수 이름 id로 전달받고 요청 처리 메서드 requestMethod()에서 매트릭스 변수 passwd로 접근함.

[@MatrixVariable을 이용해 매트릭스 변수에 접근한 예]

```
Example05Controller.java

package com.springboot.controller;
...
import org.springframework.web.bind.annotation.MatrixVariable;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;
import org.springframework.web.servlet.ModelAndView;

@Controller
public class Example05Controller {
    @GetMapping("/exam05/{id}") ①
    public String requestMethod(@PathVariable("id") String userId, ②
        @MatrixVariable("passwd") String userPw, Model model) {
        model.addAttribute("data1", "@MatrixVariable 예제");
        model.addAttribute("data2", "경로 변수 id 값: " + userId +"  
매트릭스 변수
        passwd 값: " + userPw);
        return "viewPage";
    }
}
```



- ① id는 경로 변수로, 웹 요청 URL이 `http://.../exam05/admin;passwd=admin1234`이면 파라미터 값인 admin을 전달받음.
- ② @PathVariable에 설정된 매개변수 userId는 경로 변수 id에 전달된 값인 admin을 전달받음.
- ③ @MatrixVariable에 설정된 매개변수 userPw는 웹 요청 URL로 전송된 파라미터 값에서 매트릭스 변수 passwd 값인 admin1234를 전달받음

### 3. 매트릭스 변수와 @MatrixVariable

#### ■ @MatrixVariable로 매트릭스 변수 처리

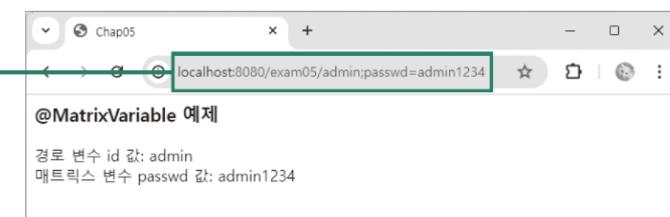
- 사용자의 웹 요청 URL(`http://localhost:8080/exam05/admin;passwd=admin1234`)에 대한 응답으로 `viewPage` 페이지를 웹 브라우저에 출력하는 과정

```
Example05Controller.java

package com.springboot.controller;
...
import org.springframework.web.bind.annotation.MatrixVariable;

@Controller
public class Example05Controller {    admin          passwd=admin1234
    @GetMapping("/exam05/{id}")
    public String RequestMethod(@PathVariable("id") String userId,
                                @MatrixVariable("passwd") String userPw, Model model) {
        model.addAttribute("data1", "@MatrixVariable 예제");
        model.addAttribute("data2", "경로 변수 id 값: " + userId + "<br> 매트릭스 변수 passwd 값: " + userPw);
        return "viewPage";
    }
}
```

`http://.../exam05/admin;passwd=admin1234`



### 3. 매트릭스 변수와 @MatrixVariable

#### ■ @MatrixVariable로 매트릭스 변수 처리

- @MatrixVariable의 value와 pathVar 속성 이용해 웹 요청 URL에 포함된 파라미터를 경로 변수에 맵핑하는 예

[@MatrixVariable에 value 및 pathVar 속성을 사용한 예]

```
Example06Controller.java

package com.springboot.controller;
...
import org.springframework.web.bind.annotation.MatrixVariable;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.GetMapping;

@Controller
public class Example06Controller {
    @GetMapping("/exam06/{id1}/user/{id2}") ①
    public String requestMethod(@PathVariable("id1") String userId1, ②
        @MatrixVariable(value="passwd", pathVar="id1") String userPw1,
        @PathVariable("id2") String userId2,
        @MatrixVariable(value="passwd", pathVar="id2") String userPw2, Model model) {
        model.addAttribute("data1", "@MatrixVariable 예제");
        model.addAttribute("data2", "경로 변수 id1 값: " + userId1 + " 매트릭스 변수
            passwd 값: " + userPw1 + "<br> 경로 변수 id2 값: " + userId2 + " 매트릭스 변수
            passwd 값: " + userPw2);

        return "viewPage";
    }
}
```



- ① id1과 id2는 경로 변수로, 웹 요청 URL이 `http://.../exam06/admin;passwd=admin1234/user/hong;passwd=hong1004` 이면 전송된 파라미터 값 admin은 경로 변수 id1로, hong은 id2로 전달받음.
- ② @PathVariable에 설정된 매개 변수 userId1은 경로 변수 id1에 전달된 값인 admin, userId2는 경로 변수 id2에 전달된 값인 hong을 전달받음(4번 또한 같음).
- ③ @MatrixVariable에 설정된 매개 변수 userPw1은 웹 요청 URL로 전송된 파라미터 값에서 경로 변수 id1에 해당하는 매트릭스 변수 passwd의 값인 admin1234를, userPw2는 웹 요청 URL로 전송된 파라미터 값에서 경로 변수 id2에 해당하는 매트릭스 변수 passwd의 값인 hong1004를 전달받음(5번 또한 같음).

### 3. 매트릭스 변수와 @MatrixVariable

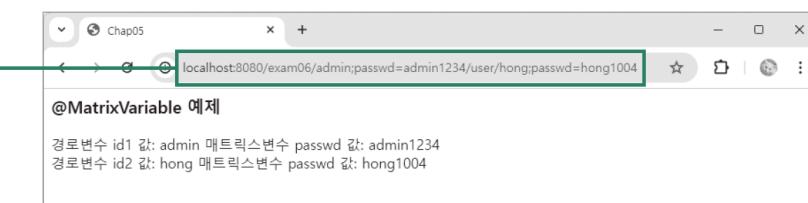
#### ■ @MatrixVariable로 매트릭스 변수 처리

- 사용자의 웹 요청

URL(<http://localhost:8080/exam06/admin;passwd=admin1234/user/hong;passwd=hong1004>)에 대한 응답으로 viewPage 페이지를 웹 브라우저에 출력하는 과정

```
Example06Controller.java  
package com.springboot.controller;  
...  
import org.springframework.web.bind.annotation.MatrixVariable;  
  
@Controller  
public class Example06Controller {  
    @GetMapping("/exam06/{id1}/user/{id2}")  
    public String requestMethod(@PathVariable("id1") String userId1,  
                               @MatrixVariable(value="passwd", pathVar="id1") String userPw1,  
                               @PathVariable("id2") String userId2,  
                               @MatrixVariable(value="passwd", pathVar="id2") String userPw2, Model model) {  
        model.addAttribute("data1", "@MatrixVariable 예제");  
        model.addAttribute("data2", "경로 변수 id1 값: " + userId1 + " 매트릭스 변수 passwd 값: " + userPw1 + "  
        경로 변수 id2 값: " + userId2 + " 매트릭스 변수 passwd 값: " + userPw2);  
        return "viewPage";  
    }  
}
```

<http://.../exam06/admin;passwd=admin1234/user/hong;passwd=hong1004>



### 3. 매트릭스 변수와 @MatrixVariable

#### ■ @MatrixVariable로 매트릭스 변수 처리

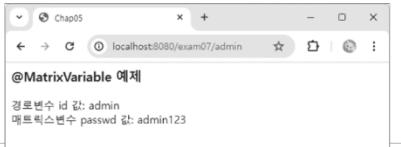
- @MatrixVariable의 defaultValue 속성을 사용해 웹 요청 URL에 매트릭스 변수가 포함되지 않으면 기본값을 설정하는 예
- 사용자의 웹 요청 URL이 http://.../exam07/admin으로 name=value 형식의 매트릭스 변수를 포함하지 않으면 @MatrixVariable의 required 속성 값은 false, defaultValue 속성 값은 매트릭스 변수 값이 되어 userPw 값은 admin123 됨.

[@MatrixVariable에 required 및 defaultValue 속성을 사용한 예]

```
Example07Controller.java

package com.springboot.controller;
...
import org.springframework.web.bind.annotation.MatrixVariable;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.GetMapping;

@Controller
public class Example07Controller {
    @GetMapping("/exam07/{id}①")
    public String requestMethod(@PathVariable("id") String userId, @MatrixVariable
        (value="passwd",defaultValue="admin123") String userPw, Model model) {
        model.addAttribute("data1", "@MatrixVariable 예제");
        model.addAttribute("data2", "경로 변수 id 값: " + userId + "<br> 매트릭스변수
            passwd 값: " + userPw);
        return "viewPage";
    }
}
```



- ① id는 경로 변수로, 웹 요청 URL이 http://.../exam07/admin이면 전송된 파라미터 값인 admin을 전달받음.
- ② @PathVariable에 설정된 매개변수 userId는 경로 변수 id에 전달된 값인 admin을 전달받음.
- ③ @MatrixVariable에 설정된 매개변수 userPw는 웹 요청 URL로 전송된 파라미터 값에서 매트릭스 변수 passwd의 값이 없다면 기본값인 admin123을 전달받음.

### 3. 매트릭스 변수와 @MatrixVariable

#### ■ @MatrixVariable로 매트릭스 변수 처리

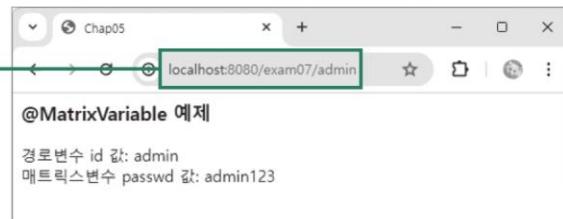
- 사용자의 웹 요청 URL(<http://localhost:8080/exam07/admin>)에 대한 응답으로 `viewPage` 페이지를 웹 브라우저에 출력하는 과정.

```
Example07Controller.java

package com.springboot.controller;
...
import org.springframework.web.bind.annotation.MatrixVariable;

@Controller
public class Example07Controller {
    @GetMapping("/exam07/{id}")
    public String RequestMethod(@PathVariable("id") String userId,
                               @MatrixVariable(value="passwd", defaultValue="admin123") String userPw,
                               Model model) {
        model.addAttribute("data1", "@MatrixVariable 예제");
        model.addAttribute("data2", "경로 변수 id 값: " + userId + "<br> 매트릭스변수 passwd 값: " + userPw);
        return "viewPage";
    }
}
```

<http://.../exam07/admin>



### 3. 매트릭스 변수와 @MatrixVariable

#### ■ @MatrixVariable로 매트릭스 변수 처리

- 웹 요청 URL의 어느 위치에나 포함될 수 있는 매트릭스 변수를 MultiValueMap 객체에 저장하여 처리하는 예

[@MatrixVariable에 Map을 사용한 예]

```
Example08Controller.java

package com.springboot.controller;
...
import org.springframework.util.MultiValueMap;
import org.springframework.web.bind.annotation.MatrixVariable;

@Controller
public class Example08Controller {
    @GetMapping("exam08/{id1}/user/{id2}")
    public String RequestMethod(@MatrixVariable MultiValueMap<String, String> var1,
        @MatrixVariable(pathVar="id2") MultiValueMap<String, String> var2, Model
        model) {
        model.addAttribute("data1", "@MatrixVariable 예제");
        model.addAttribute("data2", var1 +<p>+ var2);

        return "viewPage";
    }
}
```



The browser window shows the URL: localhost:8080/exam08/admin;passwd=admin1234;name=관리자 /user/hong;passwd=hong1004;name=홍길순. The page content displays the variables: {passwd=[admin1234, hong1004], name=[관리자, 홍길순]}.

- ① id1과 id2는 경로 변수로, 웹 요청 URL 이  
`http://.../exam08/admin;passwd=admin1234;name=관리자 /user/hong;passwd=hong1004;name=홍길순`이면 전송된 파라미터값인 admin을 id1이, hong을 id2가 전달받음
- ② @MatrixVariable에 설정된 매개변수 var1은 매트릭스 변수 name과 passwd 를 모두 전달 받으므로  
`{passwd=[admin1234, hong1004], name=[관리자, 홍길순]}`이 됨.
- ③ @MatrixVariable에 설정된 매개변수 var2는 pathVar 옵션 설정으로 경로 변수 id2에 설정된 매트릭스 변수 name과 passwd만 전달받아  
`{passwd=[hong1004], name=[홍길순]}`이 됨

### 3. 매트릭스 변수와 @MatrixVariable

#### ■ @MatrixVariable로 매트릭스 변수 처리

- 사용자의 웹 요청

URL(`http://localhost:8080/exam08/admin;passwd=admin1234;name=관리자 /user/hong;passwd=hong1004;name=홍길순`)에 대한 응답으로 `viewPage` 페이지를 웹 브라우저에 출력하는 과정.

```
Example08Controller.java  
package com.springboot.controller;  
...  
import org.springframework.util.MultiValueMap;  
import org.springframework.web.bind.annotation.MatrixVariable;  
  
@Controller  
public class Example08Controller {  
    @GetMapping("/exam08/{id1}/user/{id2}")  
    public String RequestMethod(@MatrixVariable MultiValueMap<String, String> var1,  
                               @MatrixVariable(pathVar="id2") MultiValueMap<String, String> var2,  
                               Model model) {  
        model.addAttribute("data1", "@ MatrixVariable 예제");  
        model.addAttribute("data2", var1 + "<p>" + var2);  
        return "viewPage";  
    }  
}
```

`http://.../exam08/admin;passwd=admin1234;name=관리자/user/hong;passwd=hong1004;name=홍길순`



04

[도서 쇼핑몰] 도서 상세 정보  
표시하기

# 4. [도서 쇼핑몰] 도서 상세 정보 표시하기

## [미리보기] 도서 쇼핑몰 로드맵



# 4. [도서 쇼핑몰] 도서 상세 정보 표시하기

## [미리보기] 도서 상세 정보 표시

The image shows two screenshots of a BookMarket application interface.

**Left Window (도서 목록):**

- 도서 목록** (BookList)
- 자바스크립트 입문** (JavaScript Introduction)  
작 giả: 조현경  
출판일: 2024/02/20  
가격: 30000원  
상세정보 >
- 파이썬의 정석** (Python's Way)  
작 giả: 조용주, 임화상  
출판일: 2023/01/10  
가격: 29800원  
상세정보 >
- 안드로이드 프로그래밍** (Android Programming)  
작 giả: 송이정  
출판일: 2023/06/30  
가격: 36000원  
상세정보 >

**Right Window (도서 정보):**

- 도서 정보** (Book Information)
- 자바스크립트 입문** (JavaScript Introduction)  
작 giả: 조현경  
출판사: 길벗  
출판일: 2024/02/20  
분류: IT전문서  
재고수: 1000  
가격: 30000원  
도서주문 > 도서 목록 >
- Summary: 자바스크립트의 기초부터 실무까지 핵심 문법을 학습한 후 12가지 프로그램을 만들며 학습한 내용을 확인할 수 있습니다. 몬법 학습과 실습이 적절히 섞여 있어 프로그램을 만드는 방법을 재미있게 익힐 수 있습니다.

## 4. [도서 쇼핑몰] 도서 상세 정보 표시하기

### ■ 실습 1 - @RequestParam으로 ID와 일치하는 도서 상세 정보 출력하기

1. 도서 상세 정보를 가져오는 메서드 정의: BookRepository 인터페이스에 다음과 같이 추가 작성함.

BookMarket/src/main/java/com/springboot/repository/BookRepository.java

```
package com.springboot.repository;

import java.util.List;
import java.util.Map;
import java.util.Set;

import com.springboot.domain.Book;

public interface BookRepository {
    List<Book> getAllBookList();
    Book getBookById(String bookId);
}
```

@RequestParam을 이용해 웹 요청 URL에 포함된 요청 파라미터의 도서 ID와 일치하는 도서의 상세 정보를 출력하는 페이지를 만들어 봅니다.

## 4. [도서 쇼핑몰] 도서 상세 정보 표시하기

### ■ 실습 1 - @RequestParam으로 ID와 일치하는 도서 상세 정보 출력하기

2. 도서 상세 정보를 가져오는 메서드 작성: BookRepositoryImpl 클래스에 다음과 같이 추가 작성함.

BookMarket/src/main/java/com/springboot/repository/BookRepositoryImpl.java

```
package com.springboot.repository;
...
@Repository
public class BookRepositoryImpl implements BookRepository {
    ...
    public Book getBookById(String bookId) {
        Book bookInfo = null;
        for(int i = 0; i<listOfBooks.size(); i++) {
            Book book = listOfBooks.get(i);
            ②
            if(book!=null && book.getBookId()!=null && book.getBookId().equals(bookId)) {
                bookInfo = book;
                break;
            }
        }
        ③
        if(bookInfo == null)
            throw new IllegalArgumentException("도서ID가 "+bookId + "인 해당 도서를 찾을 수
                없습니다.");
        return bookInfo;
    }
}
```

- ① 매개변수 bookId가 도서 ID를 전달받아 일치하는 도서를 검색하는 메서드.
- ② 매개변수 bookId가 저장소 객체에 저장된 도서 목록의 ID와 비교하여 일치하는 도서 정보를 반환.
- ③ 매개변수 bookId와 장소 객체에 저장된 도서 목록의 ID와 일치하는 도서가 없으면 발생하는 예외.

## 4. [도서 쇼핑몰] 도서 상세 정보 표시하기

### ■ 실습 1 - @RequestParam으로 ID와 일치하는 도서 상세 정보 출력하기

3. 도서 상세 정보를 가져오는 메서드 정의: BookService 인터페이스에 다음과 같이 추가 작성

BookMarket/src/main/java/com/springboot/service/BookService.java

```
package com.springboot.service;

import java.util.List;

import com.springboot.domain.Book;

public interface BookService {
    List<Book> getAllBookList();
    Book getBookById(String bookId);
}
```

## 4. [도서 쇼핑몰] 도서 상세 정보 표시하기

### ■ 실습 1 - @RequestParam으로 ID와 일치하는 도서 상세 정보 출력하기

4. 도서 상세 정보를 가져오는 메서드 작성: BookServiceImpl 클래스에 다음과 같이 추가 작성함.

BookMarket/src/main/java/com/springboot/service/BookServiceImpl.java

```
package com.springboot.service;  
...  
@Service  
public class BookServiceImpl implements BookService {  
    ...  
    public Book getBookById(String bookId) {  
        Book bookById = bookRepository.getBookById(bookId);  
        return bookById;  
    }  
}
```

## 4. [도서 쇼핑몰] 도서 상세 정보 표시하기

### ■ 실습 1 - @RequestParam으로 ID와 일치하는 도서 상세 정보 출력하기

5. 도서 상세 정보를 가져오는 요청 처리 메서드 생성: BookController 클래스에 다음과 같이 추가 작성함.

```
BookMarket/src/main/java/com/springboot/controller/BookController.java

package com.springboot.controller;
...
import org.springframework.web.bind.annotation.RequestParam;
...
@Controller
@RequestMapping(value = "/books")
public class BookController {
    ...
    @GetMapping("/book")
    public String requestBookById(@RequestParam("id") String bookId, Model model) {
        Book bookById = bookService.getBookById(bookId);
        model.addAttribute("book", bookById);
        return "book";
    }
}
```

- ① 웹 요청 URL이 `http://.../books/book?id=도서ID`이면 해당 도서 ID의 상세 정보를 가져오는 요청 처리 메서드 `requestBookById()`임. `@GetMapping("/book")`은 `@RequestMapping(value="/book", method=RequestMethod.GET)` 또는 `@RequestMapping("/book")`과 같음
- ② 웹 요청 URL에서 요청 파라미터인 도서 ID를 전달받도록 요청 처리 메서드 `requestBookById( )`의 매개변수에 `@RequestParam("id")` 작성함. 웹 요청 URL이 `http://.../books/book?id=ISBN1234`이면 메서드의 매개변수 `bookId` 값은 `ISBN1234`가 됨.

## 4. [도서 쇼핑몰] 도서 상세 정보 표시하기

### ■ 실습 1 - @RequestParam으로 ID와 일치하는 도서 상세 정보 출력하기

6. 도서 상세 정보 버튼 작성: books.html 파일에 다음과 같이 추가 작성함.

```
BookMarket/src/main/resources/templates/books.html

<html>
...
<div class="row align-items-md-stretch text-center">
    <div class="col-md-4" th:each="book:${bookList}">
        <h3 th:text ="${book.name}"></h3>
        <p th:text = "${book.author}">
        <p th:text = "${book.publisher} + '|' + ${book.releaseDate}">
        <p align="left" th:text = "${book.description}">
        <p th:text = "${book.unitPrice}+'원'">
        <p><a th:href= "'/BookMarket/books/book?id=' + ${book.bookId}" class="btn btn-Secondary" role="button">상세정보 &gt;</a> ①
    </div>
</div>
...

```

- ① [상세정보>>] 버튼 만들고, 버튼을 누르면 요청 경로 및 요청 파라미터로 해당 도서 ID를 전달

## 4. [도서 쇼핑몰] 도서 상세 정보 표시하기

### ■ 실습 1 - @RequestParam으로 ID와 일치하는 도서 상세 정보 출력하기

7. 프로젝트 실행: 웹 브라우저에 <http://localhost:8080/BookMarket/books> 입력하면 [상세정보>>] 버튼이 생성된 실행 결과를 확인할 수 있음.

The screenshot shows a web browser window titled "도서 목록" (Book List) for "BookMarket". The URL in the address bar is "localhost:8080/BookMarket/books". The page displays three book entries:

제목	작성자	출판일	가격	상세정보
자바스크립트 입문	조현영	길벗 2024/02/20	30000원	<a href="#">상세정보 &gt;</a>
파이썬의 정석	조용주, 임좌상	길벗 2023/01/10	29800원	<a href="#">상세정보 &gt;</a>
안드로이드 프로그래밍	송미영	길벗 2023/06/30	36000원	<a href="#">상세정보 &gt;</a>

The "상세정보" button is highlighted with a green background and white text, indicating it is a clickable link.

#### 4. [도서 쇼핑몰] 도서 상세 정보 표시하기

## ■ 실습 1 - @RequestParam으로 ID와 일치하는 도서 상세 정보 출력하기

8. 도서 상세 정보 페이지 작성: book.html 파일 생성해 다음과 같이 작성함:

BookMarket/src/main/resources/templates/book.html

## 4. [도서 쇼핑몰] 도서 상세 정보 표시하기

### ■ 실습 1 - @RequestParam으로 ID와 일치하는 도서 상세 정보 출력하기

```
3.213.492zM8 1.783C7.015.936 5.587.81 4.287.94c-1.514.153-3.042.672-
3.994 1.105A.5.5 0 0 0 2.5v11a.5.5 0 0 0 .707.455c.882-.4 2.303-.881
3.68-1.02 1.409-.142 2.59.087 3.223.877a.5.5 0 0 0 .78 0c.633-.79 1.814-
1.019 3.222-.877 1.378.139 2.8.62 3.681 1.02A.5.5 0 0 0 16 13.5v-11a.5.5
0 0 -.293-.455c-.952-.433-2.48-.952-3.994-1.105C10.413.809 8.985.936 8
1.783"/>
</svg>
<span class="fs-4">BookMarket</span>
</a>
</header>

<div class="p-5 mb-4 bg-body-tertiary rounded-3">
  <div class="container-fluid py-5">
    <h1 class="display-5 fw-bold">도서 정보</h1>
    <p class="col-md-8 fs-4">BookMarket</p>
  </div>
</div>

<div class="row align-items-md-stretch">
  <div class="col-md-12">
    <h3 th:text ="${book.name}"></h3>
    <p th:text ="${book.description}"></p>
    <br>
    <p>도서코드 : <b><span class="badge text-bg-info" th:text ="${book.bookId}"></span></b></p>
    <p>저자</b> : <span th:text ="${book.author}"></span>
    <p>출판사</b> : <span th:text ="${book.publisher}"></span>
    <p>출판일</b> : <span th:text ="${book.releaseDate}"></span>
    <p>분류</b> : <span th:text ="${book.category}"></span>
    <p>재고수</b> : <span th:text ="${book.unitsInStock}"></span>
    <h4 th:text ="${book.unitPrice}+원"></h4>
    <br>
    <p><a href="#" class="btn btn-primary">도서주문 &raquo;</a>
      <a href="/BookMarket/books" class="btn btn-secondary">도서 목록&raquo;</a></p>
  </div>
</div>
```

①

②

## 4. [도서 쇼핑몰] 도서 상세 정보 표시하기

### ■ 실습 1 - @RequestParam으로 ID와 일치하는 도서 상세 정보 출력하기

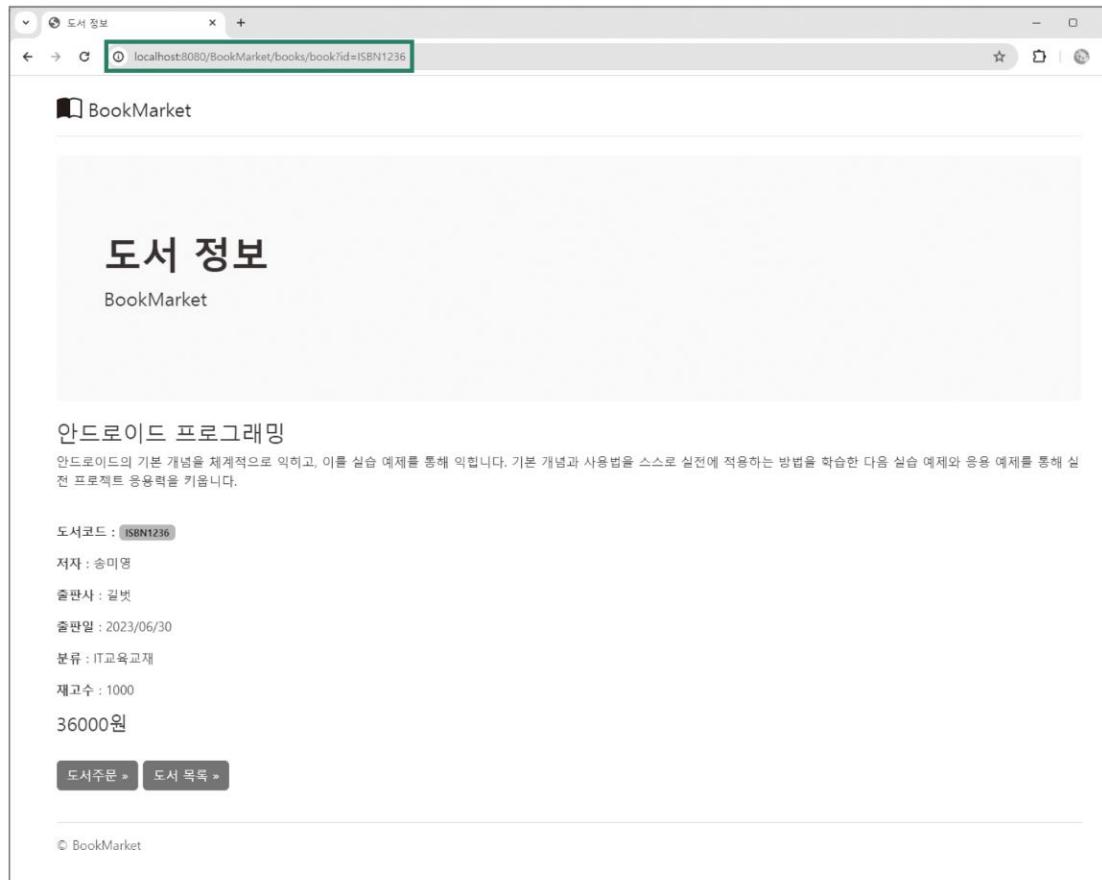
```
<footer class="pt-3 mt-4 text-body-secondary border-top">
    <span class="text-body-secondary">&copy; BookMarket</span>
</footer>
</div>
</body>
</html>
```

- ① 컨트롤러 BookController의 요청 처리 메서드 requestBookById( )에서 모델 데이터 book에 저장된 도서 상세 정보를 출력함.
- ② [도서 목록>>] 버튼 누르면 전체 도서 목록으로 이동함.

## 4. [도서 쇼핑몰] 도서 상세 정보 표시하기

### ■ 실습 1 - @RequestParam으로 ID와 일치하는 도서 상세 정보 출력하기

9. 프로젝트 실행: 웹 브라우저에 `http://localhost:8080/BookMarket/books` 입력하여 실행 후 [상세정보>>] 버튼을 누르거나 `http://localhost:8080/BookMarket/books/book?id=도서ID` 입력하여 실행 결과를 확인함.



## 4. [도서 쇼핑몰] 도서 상세 정보 표시하기

### ■ 실습 2 - @PathVariable로 분야 일치하는 도서 목록 출력하기

1. 도서 분야를 가져오는 메서드 정의: BookRepository 인터페이스에 다음과 같이 추가 작성함

BookMarket/src/main/java/com/springboot/repository/BookRepository.java

```
package com.springboot.repository;  
...  
public interface BookRepository {  
    List<Book> getAllBookList();  
    Book getBookById(String bookId);  
    List<Book> getBookListByCategory(String category);  
}
```

@PathVariable을 이용해 웹 요청 URL에 전송된 도서 분야(category) 값을 경로 변수 category로 전달받아 도서 목록에서 일치하는 도서를 검색한 뒤 목록을 출력합니다.

## 4. [도서 쇼핑몰] 도서 상세 정보 표시하기

### ■ 실습 2 - @PathVariable로 분야 일치하는 도서 목록 출력하기

2. 도서 분야를 가져오는 메서드 작성: BookRepositoryImpl 클래스에 다음과 같이 추가 작성함. com.springboot.repository 패키지의 BookRepositoryImpl 클래스에 getBookListByCategory( ) 메서드를 구현함. getBookListByCategory( ) 메서드는 웹 요청 URL로 전송된 경로 변수 값과 도서 목록의 도서 분야(category 필드 값) 비교하여 일치하는 도서 정보를 저장한 후 반환함.

BookMarket/src/main/java/com/springboot/repository/BookRepositoryImpl.java

```
package com.springboot.repository;  
...  
@Repository  
public class BookRepositoryImpl implements BookRepository {  
    ...  
    public Book getBookById(String bookId) {  
        Book bookInfo = null;  
        for(int i = 0; i<listOfBooks.size(); i++) {  
            Book book = listOfBooks.get(i);  
            if(book!= null && book.getBookId()!= null && book.getBookId().equals(bookId)) {  
                bookInfo = book;  
                break;  
            }  
        }  
        if(bookInfo == null)  
            throw new IllegalArgumentException("도서ID가 "+bookId + "인 해당 도서를 찾을 수 없습니다.");  
    }
```

## 4. [도서 쇼핑몰] 도서 상세 정보 표시하기

### ■ 실습 2 - @PathVariable로 분야 일치하는 도서 목록 출력하기

```
        return bookInfo;
    }

    public List<Book> getBookListByCategory(String category) {
        List<Book> booksByCategory = new ArrayList<Book>();
        for(int i = 0; i < listOfBooks.size(); i++) {
            Book book = listOfBooks.get(i);
            if(category.equalsIgnoreCase(book.getCategory()))
                booksByCategory.add(book);
        }
        return booksByCategory;
    }
}
```

- ① 분야가 일치하는 도서 목록 가져오는 요청 처리 메서드
- ② 전달받은 매개변수 category와 저장소 객체에 저장된 전체 도서 목록에서 분야가 일치하는 도서 정보를 저장함.

## 4. [도서 쇼핑몰] 도서 상세 정보 표시하기

### ■ 실습 2 - @PathVariable로 분야 일치하는 도서 목록 출력하기

3. 도서 분야를 가져오는 메서드 정의: BookService 인터페이스에 다음과 같이 추가 작성함.

BookMarket/src/main/java/com/springboot/service/BookService.java

```
package com.springboot.service;  
...  
public interface BookService {  
    List<Book> getAllBookList();  
    Book getBookById(String bookId);  
    List<Book> getBookListByCategory(String category);  
}
```

## 4. [도서 쇼핑몰] 도서 상세 정보 표시하기

### ■ 실습 2 - @PathVariable로 분야 일치하는 도서 목록 출력하기

4. 도서 분야를 가져오는 메서드 작성: BookServiceImpl 클래스에 다음과 같이 추가 작성함.

BookMarket/src/main/java/com/springboot/service/BookServiceImpl.java

```
package com.springboot.service;  
...  
  
@Service  
public class BookServiceImpl implements BookService {  
    ...  
    public Book getBookById(String bookId) {  
        Book bookById = bookRepository.getBookById(bookId);  
        return bookById;  
    }  
    1  
    public List<Book> getBookListByCategory(String category) {  
        List<Book> booksByCategory = bookRepository.getBookListByCategory(category);  
        return booksByCategory;  
    }  
}
```

① 매개변수 category와 도서 분야가 일치하는 목록을 저장소 객체에서 가져오는 메서드

# 4. [도서 쇼핑몰] 도서 상세 정보 표시하기

## ■ 실습 2 - @PathVariable로 분야 일치하는 도서 목록 출력하기

5. 도서 분야를 가져오는 요청 처리 메서드 작성: BookController 클래스에 다음과 같이 추가 작성함.

BookMarket/src/main/java/com/springboot/controller/BookController.java

```
package com.springboot.controller;  
...  
import org.springframework.web.bind.annotation.PathVariable;  
...  
@Controller  
@RequestMapping(value = "/books")  
public class BookController {  
    @Autowired  
    private BookService bookService;  
    ...  
    @GetMapping("/book")  
    public String requestBookById(@RequestParam("id") String bookId, Model model) {  
        Book bookById = bookService.getBookById(bookId);  
        model.addAttribute("book", bookById);  
        return "book";  
    }  
    @GetMapping("/{category}")  
    public String requestBooksByCategory (  
        ② @PathVariable("category") String bookCategory, Model model) {  
        List<Book> booksByCategory=bookService.getBookListByCategory(bookCategory);  
        model.addAttribute("bookList", booksByCategory);  
        return "books"; ④  
    }  
}
```

- ① 웹 요청 URL이 `http://.../books/{도서 분야}`이면 해당 도서의 정보를 가져오는 요청 처리 메서드 `requestBooksByCategory()`를 작성함. 이 코드는 `@RequestMapping(value = "/{category}", method=RequestMethod.GET)` 또는 `@RequestMapping("/{category}")`와 같고 `{category}`는 경로 변수임.
- ② 웹 요청 URL에서 경로 변수 `category`(도서 분야)를 전달받도록 요청 처리 메서드 `requestBooksByCategory()`의 매개변수에 `@PathVariable("category")`를 작성함. 웹 요청 URL이 `http://.../books/IT전문서`라면 `bookCategory` 값은 IT전문서가 됨.
- ③ 경로 변수 `category`(도서 분야)와 일치하는 도서 목록을 가져오는 서비스 객체의 메서드를 호출해 모델 데이터에 저장함.
- ④ 도서 목록 페이지를 출력하도록 뷰 이름 `books`로 작성함.

## 4. [도서 쇼핑몰] 도서 상세 정보 표시하기

### ■ 실습 2 - @PathVariable로 분야 일치하는 도서 목록 출력하기

#### 6. 도서 분야를 가져오는 뷰 페이지 작성: books.html 파일 다음과 같이 추가 작성

```
BookMarket/src/main/resources/templates/books.html

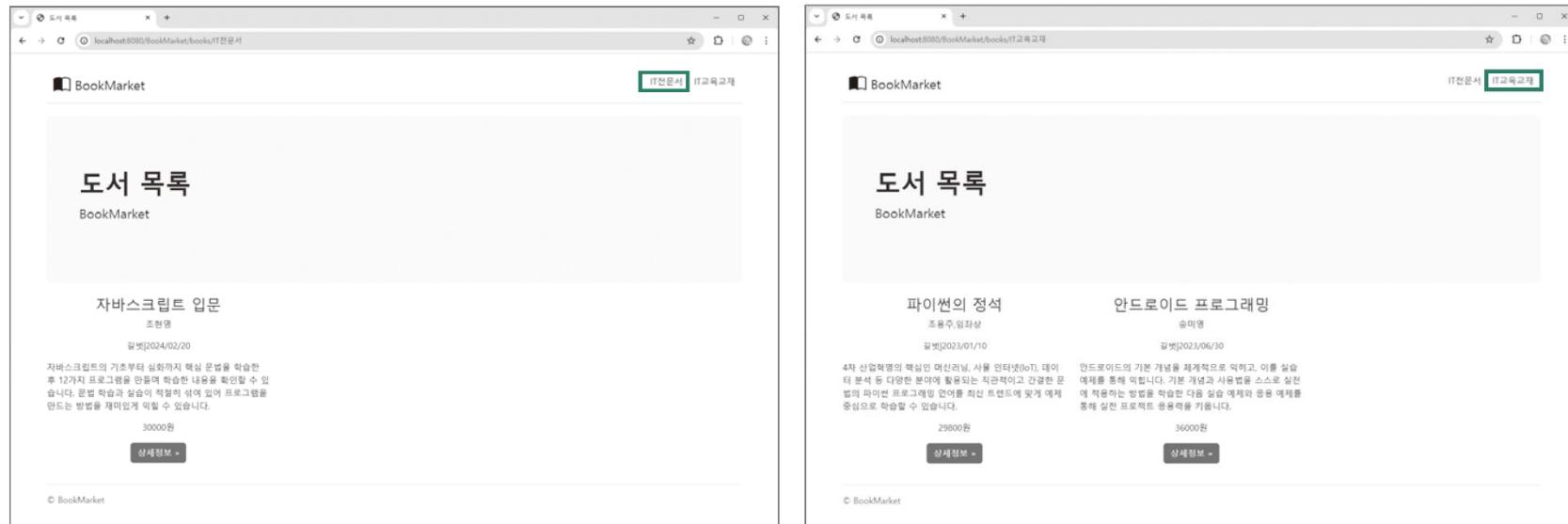
<html>
...
<div class="container py-4">
<header class="py-3 mb-4 border-bottom">
<div class="container d-flex flex-wrap justify-content-center">
<a href="/BookMarket/home" class="d-flex align-items-center link-body-emphasis text-decoration-none mb-3 mb-lg-0 me-lg-auto">
<img alt="Book icon" data-bbox="115 455 395 755" />
BookMarket
</a>
</a>
<a href="/BookMarket/books/IT전문서" class="nav-link">IT전문서</a> &ampnbsp &ampnbsp
<a href="/BookMarket/books/IT교육교재" class="nav-link">IT교육교재</a>
</div>
</header>
...
```

- ① 도서 목록에서 분야가 IT전문서 또는 IT교육교재인 도서를 검색하여 가져옴.

# 4. [도서 쇼핑몰] 도서 상세 정보 표시하기

## ■ 실습 2 - @PathVariable로 분야 일치하는 도서 목록 출력하기

7. 프로젝트 실행: 웹 브라우저에 `http://localhost:8080/BookMarket/books` 입력하여 실행하고 화면 오른쪽 상단에 IT전문서 또는 IT교육교재 선택하여 결과 확인함.



## 4. [도서 쇼핑몰] 도서 상세 정보 표시하기

### ■ 실습 3 - MatrixVariable로 분야와 출판사가 일치하는 도서 목록 출력하기

1. 도서 분야와 출판사를 가져오는 메서드 정의: BookRepository 인터페이스에 다음과 같이 추가 작성함.

BookMarket/src/main/java/com/springboot/repository/BookRepository.java

```
package com.springboot.repository;  
...  
import java.util.Map;  
import java.util.Set;  
  
public interface BookRepository {  
    ...  
    List<Book> getBookListByCategory(String category);  
    Set<Book> getBookListByFilter(Map<String, List<String>> filter);  
}
```

@MatrixVariable을 이용해 웹 요청 URL에 포함된 도서 분야(category)와 출판사(publisher)를 전달받아 도서 목록에서 매트릭스 변수 값이 일치하는 도서를 검색한 뒤 목록을 출력해 봅시다.

## 4. [도서 쇼핑몰] 도서 상세 정보 표시하기

### ■ 실습 3 - MatrixVariable로 분야와 출판사가 일치하는 도서 목록 출력하기

2. 도서 분야와 출판사를 가져오는 메서드 작성: BookRepositoryImpl 클래스에 다음과 같이 추가 작성함.

BookMarket/src/main/java/com/springboot/repository/BookRepositoryImpl.java

```
package com.springboot.repository;  
...  
import java.util.HashSet;  
  
@Repository  
public class BookRepositoryImpl implements BookRepository {
```

# 4. [도서 쇼핑몰] 도서 상세 정보 표시하기

## ■ 실습 3 - MatrixVariable로 분야와 출판사가 일치하는 도서 목록 출력하기

```
...  
public List<Book> getBookListByCategory(String category) {  
    List<Book> booksByCategory = new ArrayList<Book>();  
    for(int i = 0; i < listOfBooks.size(); i++) {  
        Book book = listOfBooks.get(i);  
        if(category.equalsIgnoreCase(book.getCategory()))  
            booksByCategory.add(book);  
    }  
    return booksByCategory;  
}  
  
public Set<Book> getBookListByFilter(Map<String, List<String>> filter) {  
    Set<Book> booksByPublisher = new HashSet<Book>();  
    Set<Book> booksByCategory = new HashSet<Book>();  
    Set<String> booksByFilter = filter.keySet();  
    if(booksByFilter.contains("publisher")) {  
        for(int j = 0; j < filter.get("publisher").size(); j++) {  
            String publisherName = filter.get("publisher").get(j);  
            for (int i = 0; i < listOfBooks.size(); i++) {  
                Book book = listOfBooks.get(i);  
                if (publisherName.equalsIgnoreCase(book.getPublisher()))  
                    booksByPublisher.add(book);  
            }  
        }  
    }  
  
    if(booksByFilter.contains("category")) {  
        for(int i = 0; i < filter.get("category").size(); i++) {  
            String category = filter.get("category").get(i);  
            List<Book> list = getBookListByCategory(category);  
            booksByCategory.addAll(list);  
        }  
    }  
  
    booksByCategory.removeAll(booksByPublisher);  
    return booksByCategory;  
}
```

- ① 웹 요청 URL에 매트릭스 변수 publisher 포함되어 있으면 전체 도서 목록에서 출판사(publisher)가 일치하는 도서를 검색하여 booksByPublisher 객체에 저장함.
- ② 웹 요청 URL에 매트릭스 변수 category가 포함되어 있으면 전체 도서 목록에서 도서 분야(category)가 일치하는 도서를 검색하여 booksByCategory 객체에 저장함.
- ③ booksByCategory 객체와 booksByPublisher에 등록된 도서 중 중복되는 것만 booksByCategory 객체에 저장한 뒤 반환함.

## 4. [도서 쇼핑몰] 도서 상세 정보 표시하기

### ■ 실습 3 - MatrixVariable로 분야와 출판사가 일치하는 도서 목록 출력하기

3. 도서 분야와 출판사를 가져오는 메서드 정의: BookService 인터페이스에 다음과 같이 추가 작성함.

BookMarket/src/main/java/com/springboot/BookService/BookService.java

```
package com.springboot.service;  
...  
import java.util.Map;  
import java.util.Set;  
  
public interface BookService {  
    ...  
    List<Book> getBookListByCategory(String category);  
    Set<Book> getBookListByFilter(Map<String, List<String>> filter);  
}
```

## 4. [도서 쇼핑몰] 도서 상세 정보 표시하기

### ■ 실습 3 - MatrixVariable로 분야와 출판사가 일치하는 도서 목록 출력하기

4. 도서 분야와 출판사를 가져오는 메서드 작성: BookServiceImpl 클래스에 다음과 같이 추가 작성함.

```
BookMarket/src/main/java/com/springboot/service/BookService/BookServiceImpl.java

package com.springboot.service;
...
import java.util.Map;
import java.util.Set;
...
@Service
public class BookServiceImpl implements BookService {
    ...
    public List<Book> getBookListByCategory(String category) {
        List<Book> booksByCategory = bookRepository.getBookListByCategory(category);
        return booksByCategory;
    }
    ...
    public Set<Book> getBookListByFilter(Map<String, List<String>> filter) {
        Set<Book> booksByFilter = bookRepository.getBookListByFilter(filter);
        return booksByFilter;
    }
}
```

①

- ① 매개변수 filter와 도서 분야 및 출판사가 일치하는 도서 목록 저장소 객체에서 가져오는 메서드

## 4. [도서 쇼핑몰] 도서 상세 정보 표시하기

### ■ 실습 3 - MatrixVariable로 분야와 출판사가 일치하는 도서 목록 출력하기

5. 도서 분야와 출판사를 가져오는 요청 처리 메서드 작성: BookController 클래스에 다음과 같이 추가 작성함.

BookMarket/src/main/java/com/springboot/controller/BookController.java

```
package com.springboot.controller;

import java.util.Map;
import java.util.Set;
import org.springframework.web.bind.annotation.MatrixVariable;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.RestController;
import org.springframework.web.servlet.ModelAndView;
import com.springboot.service.BookService;
import com.springboot.vo.Book;

@RestController
@RequestMapping(value = "/books")
public class BookController {

    @GetMapping("/{category}")
    public String requestBooksByCategory (
        @PathVariable("category") String bookCategory, Model model) {
        List<Book> booksByCategory = bookService.getBookListByCategory(bookCategory);
        model.addAttribute("bookList", booksByCategory);
        return "books";
    }

    @GetMapping("/filter/{bookFilter}")
    public String requestBooksByFilter (
        @MatrixVariable(pathVar = "bookFilter") Map<String, List<String>> bookFilter,
        Model model) { ②
        Set<Book> booksByFiter = bookService.getBookListByFilter(bookFilter);
        model.addAttribute("bookList", booksByFiter);
        return "books"; ④
    }
}
```

①

웹 요청 URL이

http://.../books/filter/bookFilter;publisher=길벗  
;category=IT전문서라면 출판사가 길벗, 도서 분야가 IT  
전문서에 해당하는 도서의 정보를 가져오는 요청 처리  
메서드 requestBooksByFilter( )를 작성함.

②

웹 요청 URL에서 매트릭스 변수를 전달받는 요청 처리  
메서드 requestBooksByFilter( )의 매개변수에  
@MatrixVariable(pathVar="bookFilter")를 작성함.

③

매트릭스 변수 publisher(출판사), category(도서 분야)  
와 일치하는 도서 목록을 가져오는 서비스 객체의 메서드를  
호출하여 모델 데이터에 저장함.

④

도서 목록 페이지를 출력하도록 뷰 이름을 books로  
작성함.

①

②

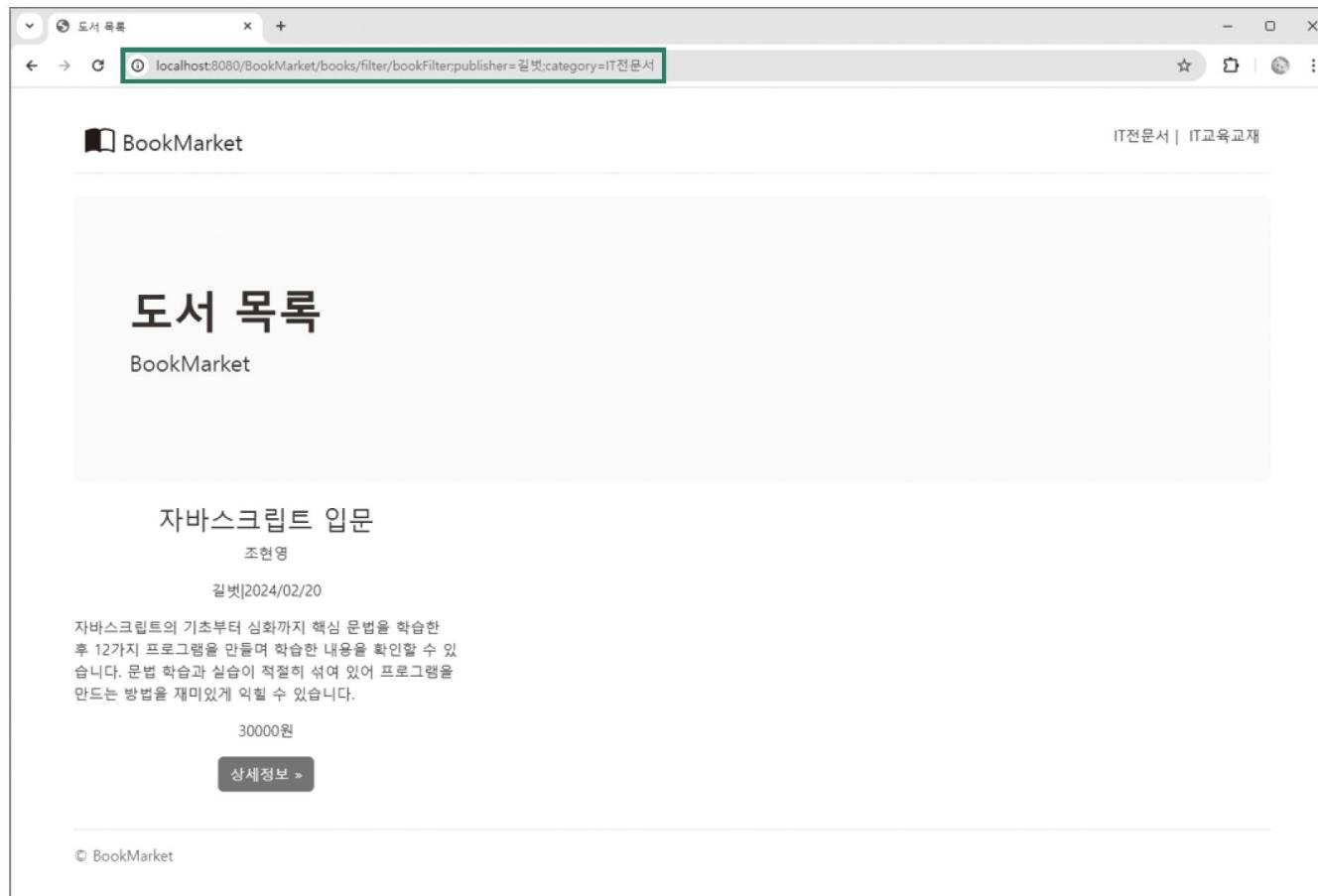
③

## 4. [도서 쇼핑몰] 도서 상세 정보 표시하기

### ■ 실습 3 - MatrixVariable로 분야와 출판사가 일치하는 도서 목록 출력하기

#### 6. 프로젝트 실행: 웹 브라우저에

`http://localhost:8080/BookMarket/books/filter/bookFilter;publisher=길벗  
;category=IT전문서` 입력하고 실행 결과를 확인함.



## 4. [도서 쇼핑몰] 도서 상세 정보 표시하기

### ■ 실습 3 - MatrixVariable로 분야와 출판사가 일치하는 도서 목록 출력하기

- 이번에는 `http://localhost:8080/BookMarket/books/filter/bookFilter;publisher=길벗;category=IT전문서,IT교육교재`를 입력함.
- BookController 컨트롤러의 `requestBooksByFilter()` 메서드에서 매트릭스 변수 `bookFilter`값은 `{publisher : [길벗]}, {category : [IT전문서, IT교육교재]}`가 된 것을 확인 할 수 있음

도서 목록

BookMarket

도서 목록

BookMarket

자바스크립트 입문  
조현영  
길벗|2024/02/20

안드로이드 프로그래밍  
송미영  
길벗|2023/06/30

파이썬의 정석  
조용주, 임좌상  
길벗|2023/01/10

© BookMarket

# Q&A