

웹 프로그래밍

3장. Spring Boot 기초 실습

동아대학교 컴퓨터공학과
양 선

웹 서버에 회원이 접속했을 때

회원님이 로그인하셨을 때

그루트님께서...

카리나님께서...

샘스미스님께서...



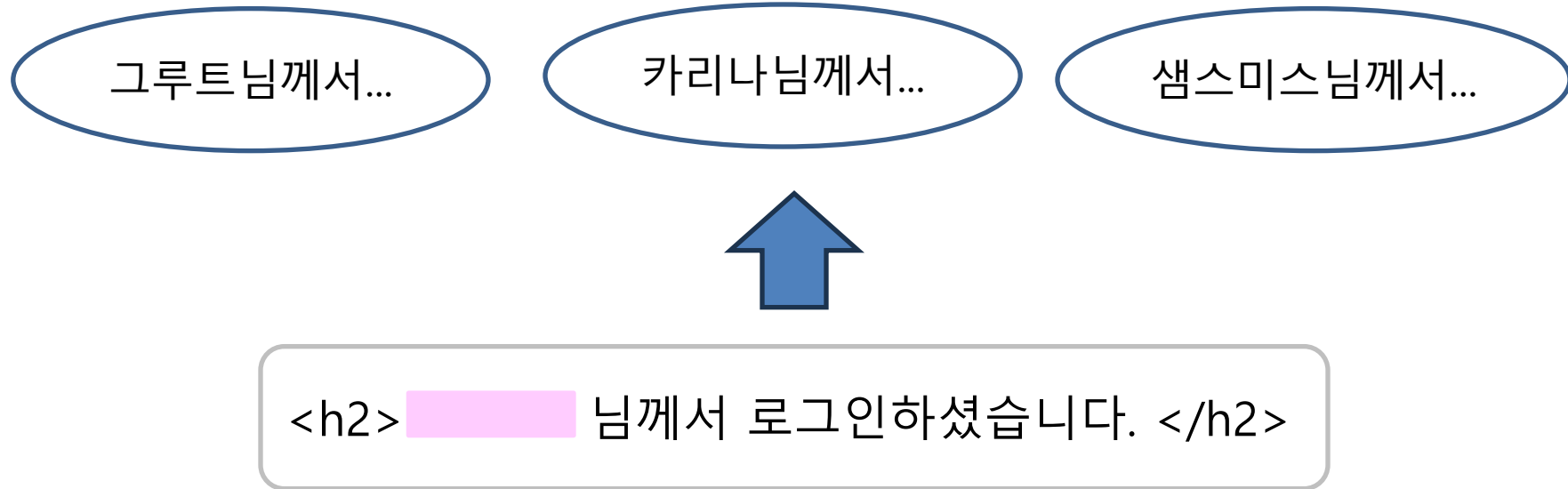
```
<h2>그루트님께서 로그인하셨습니다.</h2>
```

```
<h2>카리나님께서 로그인하셨습니다.</h2>
```

```
<h2>샘스미스님께서 로그인하셨습니다.</h2>
```

서버에 회원 수만큼
미리 html 소스 준비?

소스는 1개



서버에 소스는 1개.
변수가 포함되는 html을 view template 이라고 합니다.
(줄여서 그냥 template이라고도 부름)

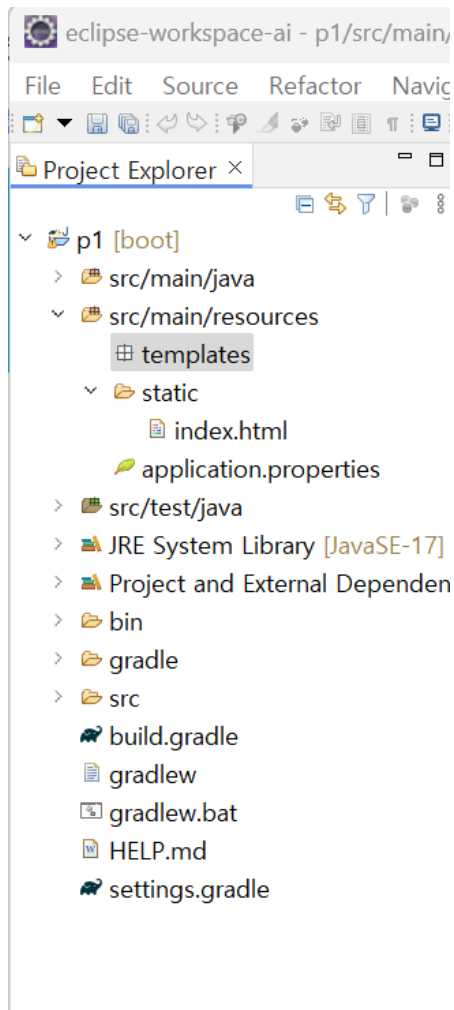
우리가 주로 사용할 Spring Boot 요소

종류	매우 간단하게 요약하자면
@Controller	총 컨트롤 역할. 사용자 요구 감지하고 view 선택
@Service	logic 구현. 컨트롤러를 도와줌 (이번 학기에 여기까지는 안나감)
@Entity	DB의 테이블에 해당 (5장 이후)
@Repository	Entity를 DB에 연결 (소스에 어노테이션 생략)
@RequestParam	앞에서 보낸 파라미터를 받을 때 사용. (생략해도 되는 경우 많지만, 명확히 하기 위해 사용하는 걸로)
html 화면들	변수를 html에 표현해 주기 위해 우리는 thymeleaf 라는 view template 사용 (templates 폴더의 html파일들)

웹 화면과 주소

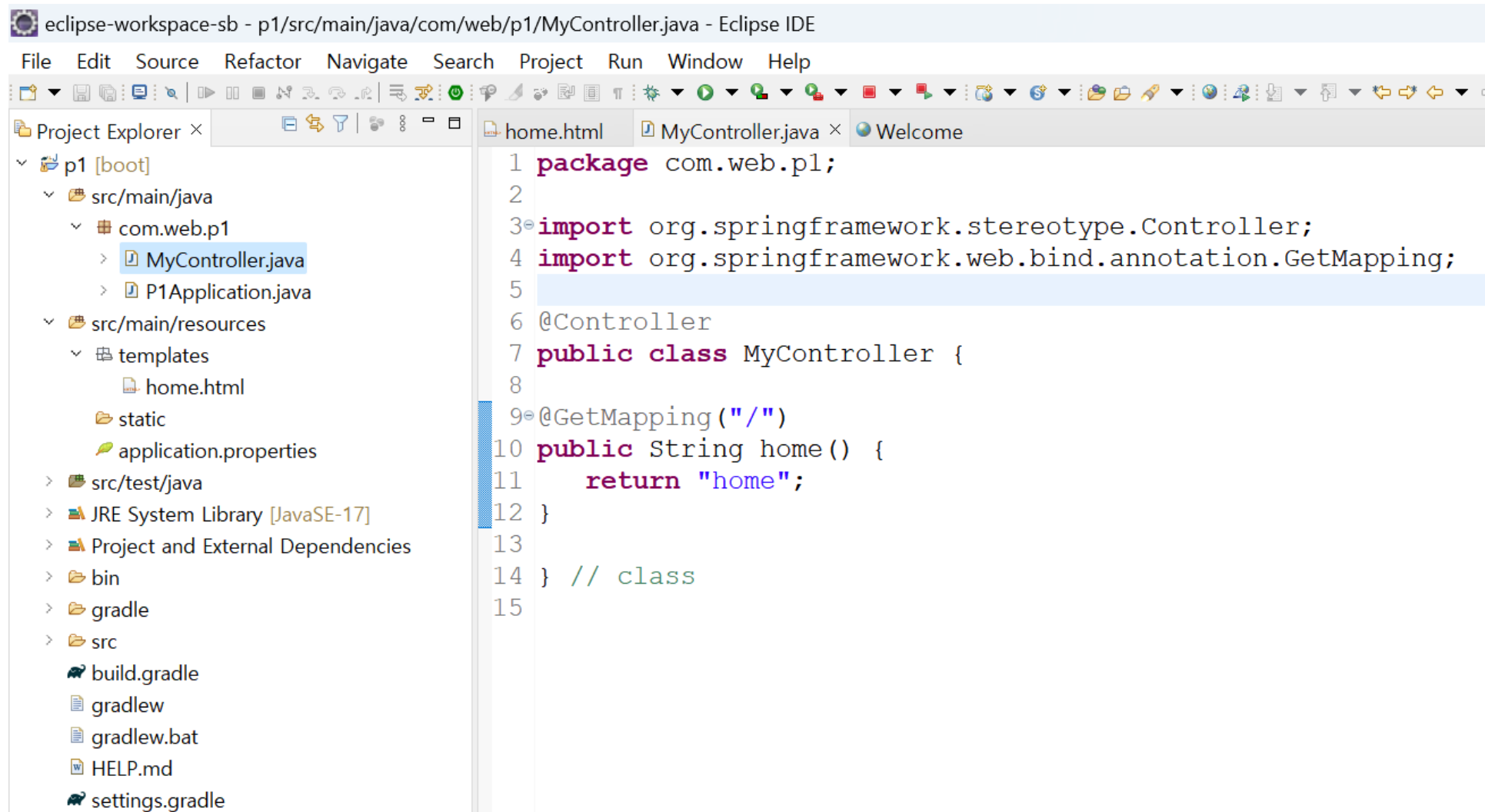
- 예를 들어 네이버 뉴스 주소는 <https://news.naver.com>
- 네이버 뉴스에서 IT/과학 탭 클릭하면 <https://news.naver.com/section/105>
- (대체적으로) 화면 1개당 주소 1개
- **Controller:** 어떤 주소에 어떤 화면(html)을 내보낼 지 결정
- [참고] http, https 차이?
 - localhost:8080은 https 아니고 http

웹 화면과 주소창



- **templates 폴더** 안에 html 파일들이 Controller가 내보낼 view template 에 해당
- 이 폴더에 있는 html은 1장에서 배웠던 순수한 html파일과는 달리 view template에 해당합니다.
 - ▣ 변수도 넣고 반복문/조건문도 넣을 수 있습니다.
- 우리는 이번 학기 **타임리프 (thymeleaf)** 라는 뷰 템플릿을 사용합니다.
 - ▣ 타임리프 뷰 템플릿의 특징: 확장자가 html
- **static폴더** 안에는 index.html (미리 약속한 파일명) 외에 css파일, 그림파일 등을 저장

Controller 작성 시작 (Java 소스)



MyController.java

```
package com.web.p1;
```

```
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.GetMapping;
```

← 자동import 하려면?

```
@Controller
```

← @Controller 없으면?

```
public class MyController {
```

```
@GetMapping("/")
```

```
public String home() {
    return "home";
}
```

← return "home.html";

```
} // class
```

/* @GetMapping("/") 은 서버접속 첫화면을 의미 */

/* 어? 서버 접속 첫 화면은 static폴더의 index.html
였는데? 어떻게 된 걸까요?

답: Controller에서 코딩하면 index.html보다
우선순위 높아요! */

home.html

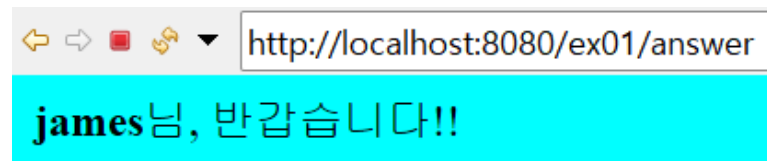
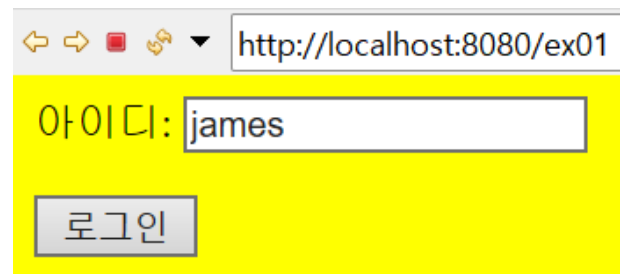
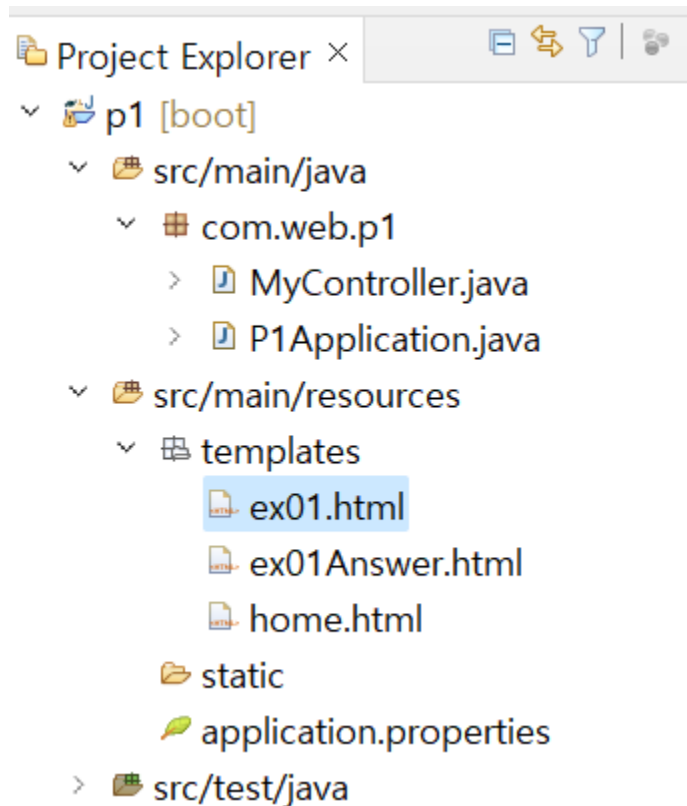
home.html은
index.html과 동일하
게 만드셔도 되고,
다르게 만드셔도 되
고,
여러분 자유!
(저는 동일하게 만든 후
index.html을 삭제하겠습
니다.)

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Welcome</title>
<style>
  body { background:lime; }
  li { line-height: 1.5em; }
  li::marker { color: red; }
</style>
</head>
<body>
<h2>2300111 Dooli가 개설한 서버 홈페이지</h2>
<ul>
  <li><a href="/ex01">예제 1</a>
  <li><a href="/ex02">예제 2</a>
  <li><a href="/ex03">예제 3</a>
  <li><a href="/ex04">예제 4</a>
</ul>
</body>
</html>
```

2300111 Dooli가 개설한 서버 홈페이지

- [예제 1](#)
- [예제 2](#)
- [예제 3](#)
- [예제 4](#)

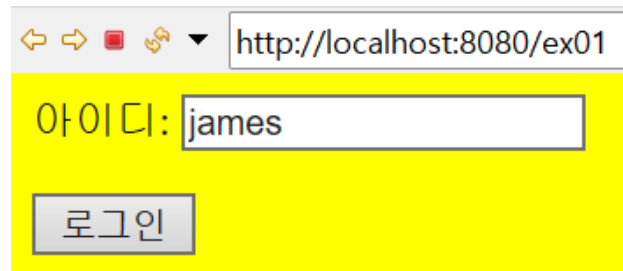
첫 번째 예제: 반갑습니다!



앞으로 화면은 templates 폴더에 저장

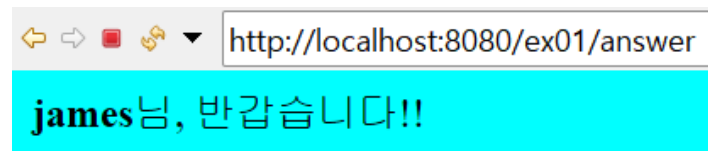
ex01.html

```
<!DOCTYPE html><html>
<head><meta charset="UTF-8">
<title>보내는 쪽</title></head>
<body style="background-color:yellow">
<form method="post" action="/ex01/answer" >
  아이디: <input type="text" name="mid"> <p>
  <input type="submit" value="로그인">
</form>
</body>
</html>
```



ex01Answer.html

```
<!DOCTYPE html><html>
<head><meta charset="UTF-8">
<title>받는 쪽</title></head>
<body style="background-color:aqua">
<strong>james</strong>님, 반갑습니다!!
</body>
</html>
```



html
미완성

MyController.java에 메소드(ex01, ex01Answer) 추가

```
package com.web.pl;  
import org.springframework.stereotype.Controller;  
import org.springframework.web.bind.annotation.GetMapping;  
import org.springframework.web.bind.annotation.PostMapping;
```

```
@Controller  
public class MyController {  
    @GetMapping("/")  
    public String home() {  
        return "home";  
    }  
    @GetMapping("/ex01")  
    public String ex01() {  
        return "ex01";  
    }  
    @PostMapping("/ex01/answer")  
    public String ex01Answer() {  
        return "ex01Answer";  
    }  
}  
  
} // class
```

메소드
미완성

- 기본적으로 @GetMapping
- return 다음에 (view template) html 파일명 (확장자 생략가능)
- 자료를 보내는 쪽 form태그의 method 속성값이 post인 경우 받는 쪽은 @PostMapping
- 주소/메소드명/html파일명 세 가지는 동일할 필요 없습니다. (주소는 주소이고, 메소드명은 메소드명이고, 파일명은 파일명)

그런데, 앞 코딩에서 문제점은?

- 받는 쪽 html 소스에 james라고 하드코딩 되어있음
 - ▣ 즉, 사용자가 아이디 dooli 라고 넣어도 받는 쪽 화면에는 무조건 james 가 뜹니다.
- 따라서, 보내는 쪽 아이디가 받는 쪽 화면에 잘 출력되려면, 받는 쪽 소스에 james라고 하드코딩하면 안 됨!! 변수 처리해야 함.
- 해야 할 일: (1) Controller 수정 (2) 받는 쪽 소스 ex01Answer.html 수정

(1) 수정된 MyController.java

```
package com.web.p1;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PostMapping;
@Controller
public class MyController {
    @GetMapping("/")
    public String home() {
        return "home";
    }
    @GetMapping("/ex01")
    public String ex01() {
        return "ex01";
    }
    @PostMapping("/ex01/answer")
    public String ex01Answer(@RequestParam(name="mid") String mid, Model mo) {
        mo.addAttribute("mid", mid);
        return "ex01Answer";
    }
}

} // class
```

메소드
완성!!

Model 이라는 가방에 넣어야 (template) html에서 볼 수 있다!

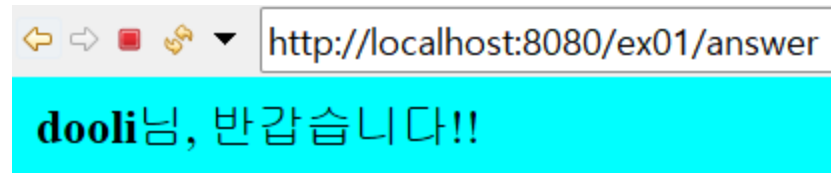
보내는 쪽 form 안에 있는 name속성의 값이 mid인 데이터를 받아서,
메소드의 매개변수 mid에 넣겠다는 의미 (둘 이름 달라도 됩니다.)

```
@PostMapping("/ex01/answer")  
public String ex01Answer(@RequestParam(name="mid") String mid, Model mo) {  
    mo.addAttribute("mid", mid);  
    return "ex01Answer";  
}
```

우측 mid는 매개변수 mid이고
좌측 id는 mo 가방 안에서의 이름입니다.
동일할 필요 없지만, 대부분 동일하게 사용

받은 id를 html 화면에 표시하려면
Model 이라는 가방에 넣어서 html까지
잘 운반해야 합니다.
변수명으로 소문자 model 이 자주
사용되지만, 저는 그냥 mo라고만
하겠습니다.

(2) ex01Answer.html 수정



```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
<meta charset="UTF-8">
<title>받는 쪽</title>
</head>
<body style="background-color:aqua">

<strong th:text="${mid}">james</strong>님, 반갑습니다!!

</body>
</html>
```

타임리프 namespace 입니다.
안적어도 대부분 잘 돌아가지만 적는 게 원칙

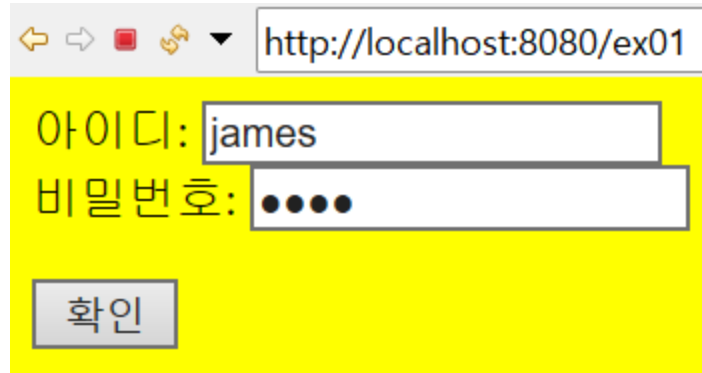
mo 안에 있던 이름

우선순위 면에서 타임리프 text가
원래있던 james 글자보다 우선순위 높음

html
완성!!

[중간점검]

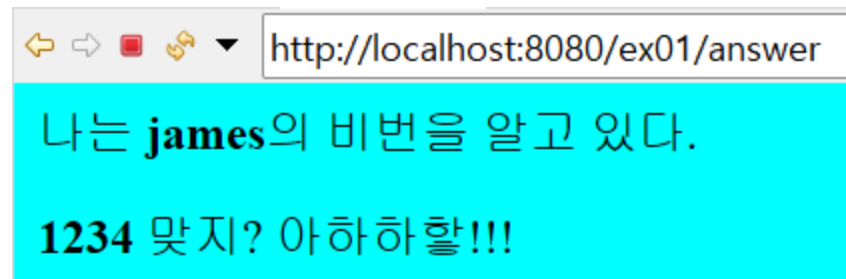
방금 코딩한 화면2개 및 Controller를 다음과 같이 수정해 보세요.



아이디: james

비밀번호:

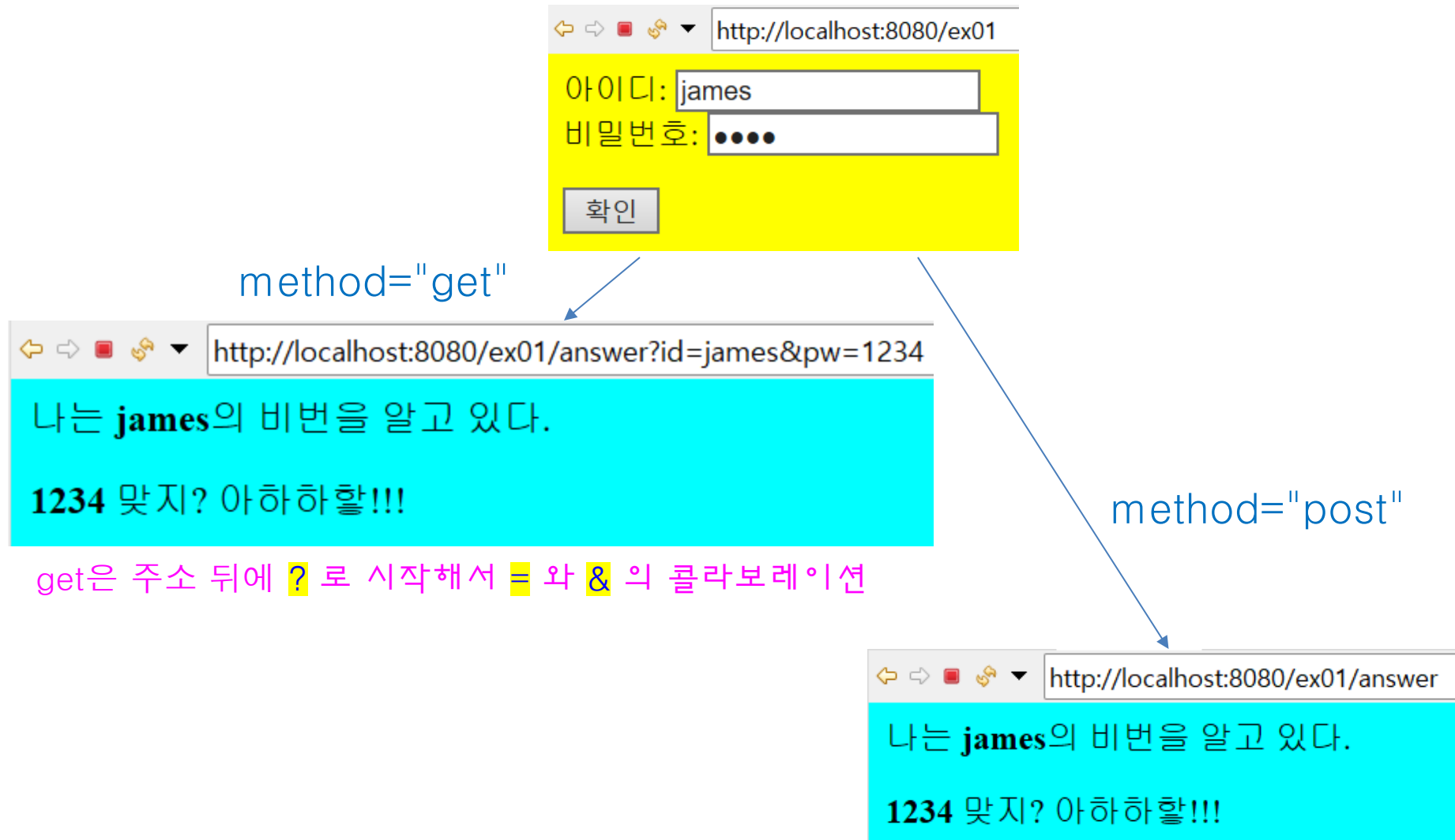
확인



나는 **james**의 비번을 알고 있다.

1234 맞지? 아하하할!!!

get VS post (어떤 차이?)

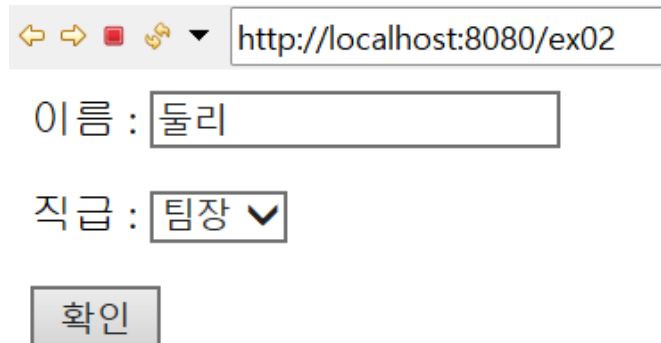


두 번째 예제: 연봉

ex02.html

```
<!DOCTYPE html> <html>
<head> <meta charset="UTF-8">
<title>직급선택</title> </head>
<body>
<form method="post" action="/ex02/answer">
이름 : <input type="text" name="mname"> <p>
직급 : <select name="po">
    <option>사원
    <option>대리
    <option>팀장
    <option>임원
</select> <p>
<input type="submit" value="확인">
</form>
</body> </html>
```

종료없이 <p>만 여러번
사용하는 경우도 많음

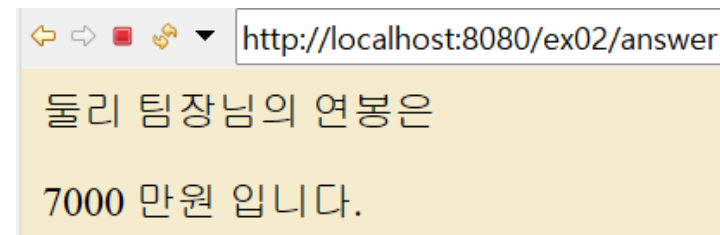


ex02Answer.html

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head> <meta charset="UTF-8">
<title>연봉</title> </head>
<body style="background-color:#F5ECCE">

<span th:text="${mname}">mname</span>
<span th:text="${po}">po</span> 님의 연봉은
<p>
<span th:text="${salary}">salary</span> 만원
입니다.

</body> </html>
```



MyController.java에 ex02, ex02Answer 메소드 추가

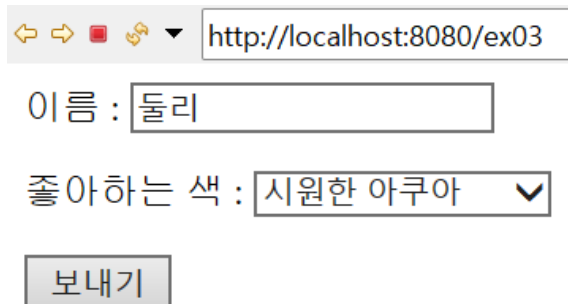
```
@GetMapping("/ex02")
public String ex02() {
    return "ex02";
}

@PostMapping("/ex02/answer")
public String ex02Answer(@RequestParam("mname") String mname,
    @RequestParam("po") String po, Model mo) {
    mo.addAttribute("mname", mname);
    mo.addAttribute("po", po);
    int salary = 0;
    switch(po){
        case "사원" -> salary = 3500; /* Java switch문 */
        case "대리" -> salary = 5000;
        case "팀장" -> salary = 7000;
        case "임원" -> salary = 9900;
    }
    mo.addAttribute("salary", salary);
    return "ex02Answer";
}
```

세 번째 예제: 좋아하는 색상

ex03.html

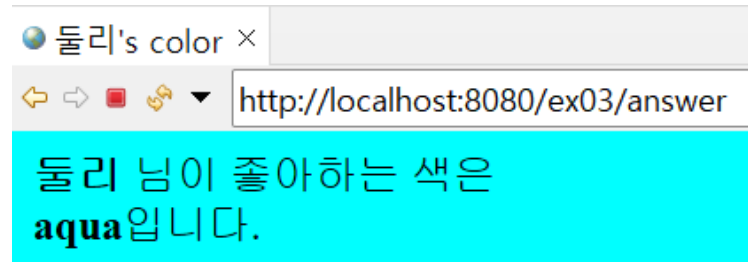
```
<!DOCTYPE html> <html>
<head> <meta charset="UTF-8">
<title>색 선택</title> </head>
<body>
<form method="post" action="/ex03/answer">
이름 : <input type="text" name="mname"> <p>
좋아하는 색 : <select name="color">
  <option value="aqua">시원한 아쿠아
  <option value="lime">라임색이 좋아요!
  <option value="orange">상큼한 오렌지색
  <option value="white">역시 흰색 최고
</select> <p>
  <input type="submit" value="보내기">
</form>
</body> </html>
```



ex03Answer.html

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head> <meta charset="UTF-8">
<title th:text="|${mname}'s color|">
</title> </head>
<body th:style="|background-color: ${color}|">
<strong th:text="${mname}">mname</strong>
님이 좋아하는 색은 <br>
<strong
th:text="${color}">color</strong> 입니다.

</body>
</html>
```



MyController.java에 ex03, ex03Answer 메소드 추가

```
@GetMapping("/ex03")
public String ex03() {
    return "ex03";
}
```

```
@PostMapping("/ex03/answer")
public String ex03Answer(@RequestParam("mname") String mname,
    @RequestParam("color") String color, Model mo) {
    mo.addAttribute("mname", mname);
    mo.addAttribute("color", color);
    return "ex03Answer";
}
```

네 번째 예제: 회원 리스트

ex04.html

```
<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
<meta charset="UTF-8"><title>list</title>
<link rel="stylesheet" href="/style.css">
</head>
<body>
<h2>회원 리스트</h2>
<table>
<tr id="firstRow">
  <th>회원ID
<tr th:each="a:${arr}">
  <td th:text="${a}">
</table>
</body>
</html>
```

style.css (static 폴더)

```
@charset "UTF-8";
body {background-color:#ECE0F8;}
table {width:200px; border:5px dashed lime; }
#firstRow {background-color:lightcyan; }
td {color:blue; text-align:center;}
```



MyController.java에 ex04 메소드 추가

```
/* 자동import 하면 import java.util.ArrayList; 가 생깁니다. */
```

```
@GetMapping("/ex04")  
public String ex04Answer(Model mo) {  
    var arr = new ArrayList<String>();
```

```
    arr.add("고흐");  
    arr.add("james");  
    arr.add("dooli");  
    arr.add("bread");
```

```
    /* 지금은 회원정보 하드코딩. 나중에는 database에서 가져옴 */
```

```
    mo.addAttribute("arr", arr);
```

```
    return "ex04";
```

```
}
```

[참고] 타임리프에는

- ◆ th:text
- ◆ th:style
- ◆ th:each
- ◆ th:if
- ◆ 기타 등등 여러가지가 있습니다.