

19강_Controller 객체 구현 - II

컨트롤러의 URL 맵핑과 파라미터 처리 방법에 대해서 학습합니다.

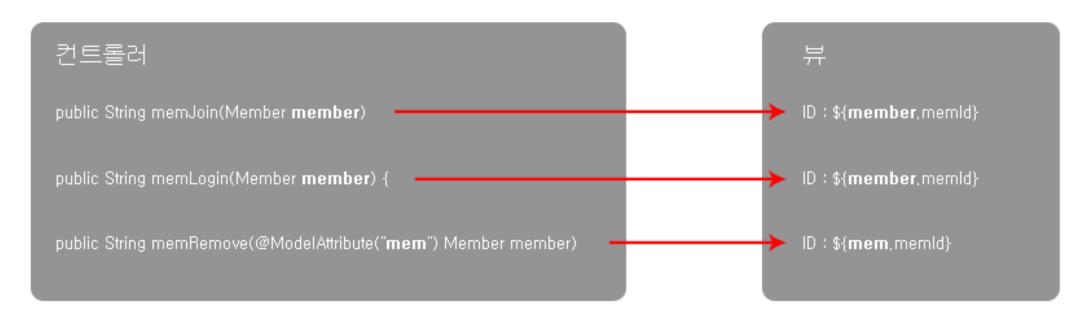
19-1 @ModelAttribute

19-2 커맨드 객체 프로퍼티 데이터 타입

19-3 Model & ModelAndView

19-1: @ModelAttribute

@ModelAttribute를 이용하면 커멘드 객체의 이름을 변경할 수 있고, 이렇게 변경된 이름은 뷰에서 커멘드 객체를 참조할 때 사용된다.



19-2: 커맨드 객체 프로퍼티 데이터 타입

데이터가 기초데이터 타입인 경우

memberJoin.html

ID : (input type="text" name="memld")

PW: (input type="password" name="memPw")

MAIL: (input type="text" name="memMail")

AGE: (input type="text" name="memAge" size="4" value="0")



Member.java

private **String** memld; private **String** memPw; private **String** memMail; private **int** memAge;

19-2: 커맨드 객체 프로퍼티 데이터 타입

데이터가 중첩 커멘드 객체을 이용한 List 구조인 경우

memberJoin.html

```
PHONE1 : <input type="text"
name="memPhones[0].memPhone1" size="5"> -
<input type="text"
name="memPhones[0].memPhone2" size="5"> -
<input type="text"
name="memPhones[0].memPhone3" size="5"> PHONE2 : <input type="text"
name="memPhones[1].memPhone1" size="5"> -
<input type="text"
name="memPhones[1].memPhone2" size="5"> -
<input type="text"
name="memPhones[1].memPhone3" size="5"> -
<input type="text"
name="memPhones[1].memPhone3" size="5"> -
<input type="text"
```



Member.java

private List < MemPhone > memPhones;

19-3: Model & ModelAndView

컨트롤러에서 뷰에 데이터를 전달하기 위해 사용되는 객체로 Model과 ModelAndView가 있다.두 객체의 차이점은 Model은 뷰에 데이터만을 전달하기 위한 객체이고, ModelAndView는 데이터와 뷰의 이름을 함께 전달하는 객체이다.

Model

```
@RequestMapping(value = "/memModify", method = RequestMethod.POST)
public String memModify(Model model, Member member) {
    Member[] members = service.memberModify(member);
    model.addAttribute("memBef", members[0]);
    model.addAttribute("memAft", members[1]);
    return "memModifyOk";
}
ID : ${memBef.memId}
ID : ${memAft.memId}
```

19-3: Model & ModelAndView

컨트롤러에서 뷰에 데이터를 전달하기 위해 사용되는 객체로 Model과 ModelAndView가 있다.두 객체의 차이점은 Model은 뷰에 데이터만을 전달하기 위한 객체이고, ModelAndView는 데이터와 뷰의 이름을 함께 전달하는 객체이다.

ModelAndView

```
@RequestMapping(value = "/memModify", method = RequestMethod.POST)
public String memModify(Model model, Member member) {

Member[] members = service.memberModify(member);

model.addAttribute("memBef", members[0]);
model.addAttribute("memAft", members[1]);

return "memModifyOk";
}

mav.setViewName("memModifyOk");

return mav;
}

@RequestMapping(value = "/memModify", method = RequestMethod.POST)
public ModelAndView memModify(Member member) {

Member[] members = service.memberModify(member);

ModelAndView mav = new ModelAndView();
mav.addObject("memBef", members[0]);
mav.addObject("memAft", members[1]);

return mav;
}

return mav;
}
```

19-3: Model & ModelAndView

