Spring 스터디 8주차(2021.10.03)

섹션 4: MVC 프레임워크 만들기
섹션 5: 스프링 MVC - 구조 이해
섹션 6: 스프링 MVC - 기본 기능

Model vs ModelAndView

Contollers는 사용자의 request를 받아서 business logic를 처리하고 그 결과를 model에 담아 view에 전달한다.
view에 전달하는 방법은 2가지에 대해 정리한다.

1. Model
    - model attributes를 저장하는 **인터페이스**

```
public interface Model {
    // 오버로딩 제외 (메소드명만 작성)
    Model addAttribute(...);
    Model addAllAttributes(...);
    Model mergeAttributes(...);
    boolean containsAttribute(...);
    Object getAttribute(...);
    Map<String, Object> asMap();
}
```

    - **addAttribute**를 통해 **데이터만** Model에 저장 (View 저장 x)
    - **논리 뷰 이름**을 **String**으로 return

```
// SpringMemberControllerV3 클래스
@PostMapping("/save")
public String save( @RequestParam("username") String username, // 파라미터를 직접 받을 수 있다.
                    @RequestParam("age") int age,
                    Model model) {

   // 비즈니스 로직
    Member member = new Member(username, age);
    memberRepository.save(member);

    // model에 데이터 저장
    model.addAttribute("member", member);
    return "save-result";
}
```

2. ModelAndView
    - **class**로 구현되어 있어 **메소드 내부에서 객체 생성**하여 사용

```
public class ModelAndView {
    public ModelAndView(..);
    public void setViewName(..);
    public String getViewName();
    public void setView(..);
    public View getView()
    public void setStatus(..);
    public HttpStatus getStatus();

    //... 그외 메소드 생략
}
```

- **addObject**를 통해 **데이터** + 이동하고자 하는 **View**를 저장
- **modelAndView** 타입으로 return

```
// SpringMemberControllerV2 클래스
@RequestMapping("/save")
public ModelAndView save(HttpServletRequest request, HttpServletResponse
response) {

    String username = request.getParameter("username");
    int age = Integer.parseInt(request.getParameter("age"));

    // 비즈니스 로직
    Member member = new Member(username, age);
    memberRepository.save(member);

    // 뷰와 데이터 저장
    ModelAndView mv = new ModelAndView("save-result"); // .jsp 뷰 이름
    mv.addObject("member", member);
    return mv; // ModelAndView 리턴
}
```

- ModeAndView는 ModeAndView 객체를 사용하여 View 객체에 대한 **직접 참조**가 가능하다.

```
ModelAndView mv = ...
mv.setView(myView);
```

- 반면에 Model은 뷰 이름을 등록하려면 뷰 리졸버가 필요하다.

- 한줄 요약하면 view 이름 설정하는 방법만 다르고 원리는 동일하다.
  - 다만, ModelAndView가 Model에 비해 기능(메소드)이 더 많다.

ViewResolver

- 뷰의 논리적인 이름이 리턴되면 **DispatcherServlet**의 **뷰 리졸버**가 호출된다.
  - JSP의 경우 기본 뷰 리졸버 **InternalResourceViewResolver**가 자동 등록된다.
  - prefix 접두어, suffix 접미어
  - 스프링 부트에서의 뷰 리졸버 설정

```
spring.mvc.view.prefix=/WEB-INF/views/
spring.mvc.view.suffix=.jsp
```

- 뷰 리졸버 종류
  - InternalResourceViewResolver (기본)
    - jsp, html 등 내부자원을 이용해 뷰 생성
  - BeanNameViewResolver
    - 뷰의 이름과 동일한 이름을 가지는 빈을 view로 사용
    - 사용자 정의 view 객체를 사용하는 경우 사용한다.
  - XmlViewResolver
    - BeanNameViewResolver과 동일하나, 뷰 객체를 xml파일에 설정해 놓는 차이
    - Bean 등록 시 location properties에 xml 파일을 지정

[참고]

- https://bestkingit.tistory.com/155
- https://aridom.tistory.com/62
- **언제 ModelAndView, Model을 사용할까?**
  https://stackoverflow.com/questions/16951609/when-to-use-modelandview-vs-model-in-spring