**2022년도 졸업프로젝트[Back-end]**

**팀이름: 황제펭귄**

주제: TLS를 구현을 통한 홈페이지 제작 및 보안 채널 로그인 기능 구현

|  |  |  |  |
| --- | --- | --- | --- |
| **작성자** | 이유민 | **작성일자** | 2022.03.01 |

* **금일 진행 상황**

|  |  |
| --- | --- |
| **진행 상황** | **비고** |
| **질문 및 진행방식에 대한 논의**  **진행 중**   1. 스프링 스터디 진행.   **완료**  스프링 빈과 의존관계   1. **컴포넌트 스캔과 자동 의존관계 설정.**   -회원 컨트롤러가 회원서비스와 회원 리포지토리를 사용할 수 있게 의존관계를 준비하는 과정을 학습함.  **1.회원 컨트롤러에 의존관계 추가**  package hello.hellospring.controller;  import hello.hellospring.service.MemberService;  import org.springframework.beans.factory.annotation.Autowired;  import org.springframework.stereotype.Controller;  @Controller  public class MemberController {  private final MemberService memberService;  @Autowired //생성자에 @Autowired 가 있으면 스프링이 연관된 객체를 스프링 컨테이너에서 찾아서 넣어준다.  public MemberController(MemberService memberService) {  this.memberService = memberService;  }  }  #의존성 주입: 생성자에 @Autowired 가 있으면 스프링이 연관된 객체를 스프링 컨테이너에서 찾아서 넣어주는데, 이렇게 객체 의존관계를 외부에서 넣어주는 것을 DI (Dependency Injection:의존성 주입)이라고 한다.  #위의 코드를 실행하면 아래의 메세지와 함께 오류가 발생하는데,이는 memberservice가 스프링 빈으로 등록되어 있지 않기 때문이다.    “Consider defining a bean of type 'hello.hellospring.service.MemberService' in your configuration”  # helloController는 스프링이 제공하는 컨트롤러여서 @Controller 가 있으면 스프링 빈으로 자동 등록된다.  **컴포넌트 스캔 원리**  #@Component :애노테이션이 있으면 스프링 빈으로 자동 등록된다.  #@Controller :컨트롤러가 스프링 빈으로 자동 등록된 이유도 컴포넌트 스캔 때문이다.  #@Component를 포함하는 다음 애노테이션도 스프링 빈으로 자동 등록된다.   * + - @Controller     - @Service     - @Repositor   **2.회원 서비스 스프링 빈 등록**  @Service  public class MemberService {  private final MemberRepository memberRepository;  @Autowired  public MemberService(MemberRepository memberRepository) {  this.memberRepository = memberRepository;  }  }  #생성자에 @Autowired 를 사용하면 객체 생성 시점에 스프링 컨테이너에서 해당 스프링 빈을 찾아서 주입한다. 생성자가 1개만 있으면 @Autowired 는 생략할 수 있다.  **3.회원 리포지토리 스프링 빈 등록**  @Repository  public class MemoryMemberRepository implements MemberRepository {}  **4.스프링 빈 등록 이미지**  #memberService 와 memberRepository 가 스프링 컨테이너에 스프링 빈으로 등록되었다.  #스프링은 스프링 컨테이너에 스프링 빈을 등록할 때, 기본으로 싱글톤으로 등록한다. 그러므로 같은 스프링 빈이면 모두 같은 인스턴스다. 특별한 경우를 제외하면 대부분 싱글톤을 사용한다.   1. **자바 코드로 직접 스프링 빈 등록하기**   package hello.hellospring;  import hello.hellospring.repository.MemberRepository;  import hello.hellospring.repository.MemoryMemberRepository;  import hello.hellospring.service.MemberService;  import org.springframework.context.annotation.Bean;  import org.springframework.context.annotation.Configuration;  @Configuration  public class SpringConfig {  @Bean  public MemberService memberService() {  return new MemberService(memberRepository());  }  @Bean  public MemberRepository memberRepository() {  return new MemoryMemberRepository();  }  }  #정형화된 컨트롤러, 서비스, 리포지토리 같은 코드는 컴포넌트 스캔을 사용한다. 그리고 정형화 되지 않거나, 상황에 따라 구현 클래스를 변경해야 하면 설정을 통해 스프링 빈으로 등록한다.  #@Autowired 를 통한 DI는 helloConroller , memberService 등과 같이 스프링이 관리하는 객체에서만 동작한다. 스프링 빈으로 등록하지 않고 내가 직접 생성한 객체에서는 동작하지 않는다.  회원관리 예제-웹 MVC 개발   1. **회원 웹 기능-홈 화면 추가**   **1.홈 컨트롤러 추가**  package hello.hellospring.controller;  import org.springframework.stereotype.Controller;  import org.springframework.web.bind.annotation.GetMapping;  @Controller  public class HomeController {  @GetMapping("/")  public String home() {  return "home";  }  }  **2.회원 관리용 홈**  <!DOCTYPE HTML>  <html xmlns:th="http://www.thymeleaf.org">  <body>  <div class="container">  <div>  <h1>Hello Spring</h1>  <p>회원 기능</p>  <p>  <a href="/members/new">회원 가입</a>  <a href="/members">회원 목록</a>  </p>  </div>  </div> <!-- /container -->  </body>  </html>  #컨트롤러가 정적파일보다 우선순위가 높다.   1. **회원 웹 기능-등록**   **-회원 등록 폼 개발**  **1.회원 등록 폼 컨트롤러**  @Controller  public class MemberController {  private final MemberService memberService;  @Autowired  public MemberController(MemberService memberService) {  this.memberService = memberService;  }  @GetMapping(value = "/members/new")  public String createForm() {  return "members/createMemberForm";  }  }  **2. 회원 등록 폼 HTML**  <!DOCTYPE HTML>  <html xmlns:th="http://www.thymeleaf.org">  <body>  <div class="container">  <form action="/members/new" method="post">  <div class="form-group">  <label for="name">이름</label>  <input type="text" id="name" name="name" placeholder="이름을  입력하세요">  </div>  <button type="submit">등록</button>  </form>  </div> <!-- /container -->  </body>  </html>  **-회원 등록 컨트롤러**  **1.웹 등록 화면에서 데이터를 전달 받을 폼 객체**  package hello.hellospring.controller;  public class MemberForm {  private String name;  public String getName() {  return name;  }  public void setName(String name) {  this.name = name;  }  }  **2.회원 컨트롤러에서 회원을 실제 등록하는 기능**  @PostMapping(value = "/members/new")  public String create(MemberForm form) {  Member member = new Member();  member.setName(form.getName());  memberService.join(member);  return "redirect:/";  }   1. **회원 웹 기능-조회**   **1. 회원 컨트롤러에서 조회 기능**  @GetMapping(value = "/members")  public String list(Model model) {  List<Member> members = memberService.findMembers();  model.addAttribute("members", members);  return "members/memberList";  }  **2. 회원 리스트 HTML**  <!DOCTYPE HTML>  <html xmlns:th="http://www.thymeleaf.org">  <body>  <div class="container">  <div>  <table>  <thead>  <tr>  <th>#</th>  <th>이름</th>  </tr>  </thead>  <tbody>  <tr th:each="member : ${members}">  <td th:text="${member.id}"></td>  <td th:text="${member.name}"></td>  </tr>  </tbody>  </table>  </div>  </div> <!-- /container -->  </body>  </html>  **진행 예정**   1. **3/1일 8시 Back-End 스터디** |  |
| * **특이사항 / 협업 사항**   매주 화, 수, 목 7시30부터 10시 30분까지 유동적으로 하기. | |