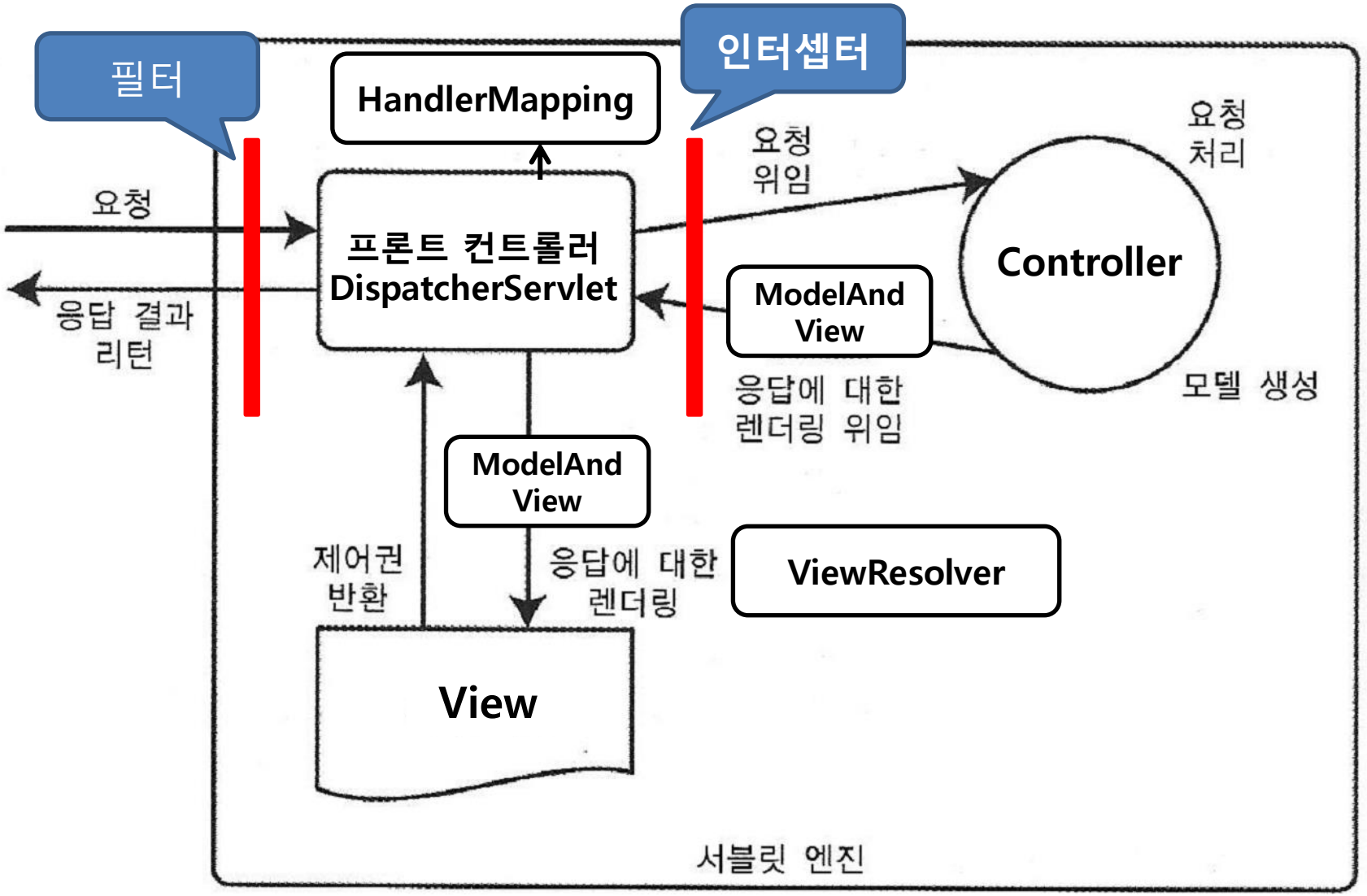
인터셉터를 이용한 로그인 처리

❖ 인터셉터

- 웹 애플리케이션 내에서 특정한 URI 호출을 가로채는 역할
- 요청 처리 전, 후 처리 가능
- JSP/SERVLET의 필터와 유사
- HandlerInterceptorAdapter를 상속하여 정의

❖ Filter와 인터셉터의 차이



❖ HandlerInterceptorAdapter

```
public boolean preHandle(
                HttpServletRequest request,
                HttpServletResponse response,
                Object handler
) throws Exception

public void postHandle(
                HttpServletRequest request,
                HttpServletResponse response,
                Object handler,
                ModelAndView modelAndView
) throws Exception

public void afterCompletion(
                HttpServletRequest request,
                HttpServletResponse response,
                Object handler,
                Exception ex
) throws Exception
```

❖ 인터셉터 메서드

메서드	리턴타입	
preHandle	boolean	1. 클라이언트의 요청을 컨트롤러에 전달 하기 전에 호출 2. false 인 경우 다음 intercepter 또는 controller 를 실행 시키지 않고 요청 종료
postHandle	void	1. 컨트롤러 로직 실행 된 후 호출됨 2. 컨트롤러 실행 도중 error 발생의 경우 postHandle() 는 실행 되지 않음 3. request 로 넘어온 데이터 가공 시 많이 쓰임
afterCompletion	void	1. 컨트롤러 로직 실행 된 후 호출 됨 2. 컨트롤러 실행 도중이나 view 페이지 실행 도중 error 발생 해도 실행됨 3. 공통 Exception 처리 로직 작성시 많이 쓰임

인터셉터를 이용한 로그인 처리

❖ **WEB-INF/spring/extension-context.xml**

```
<interceptors>
   <interceptor>
      <mapping path="/admin/*"/>
      <mapping path="/gallery/*"/>
      <mapping path="/member/*"/>
      <mapping path="/board/create"/>
      <beans:ref bean="loginInterceptor"/>
   </interceptor>

   <interceptor>
      <mapping path="/admin/*"/>
      <beans:ref bean="adminInterceptor"/>
   </interceptor>

</interceptors>
```

인터셉터를 이용한 로그인 처리

❖ **edu.iot.app.interceptor.LoginInterceptor**

```
@Component
public class LoginInterceptor extends HandlerInterceptorAdapter {
   @Autowired
   ServletContext context;

   @Override
   public boolean preHandle(HttpServletRequest request,
                HttpServletResponse response, Object handler)
         throws Exception {

      HttpSession session = request.getSession();
      if(session.getAttribute("USER") == null) {
         saveUrl(request, response);
         response.sendRedirect(context.getContextPath() +
                              "/account/login");
         return false;
      }
      return super.preHandle(request, response, handler);
   }
```

인터셉터를 이용한 로그인 처리

❖ **com.lecture.iot.interceptor.LoginInterceptor**

```
   public void saveUrl(HttpServletRequest request,
                       HttpServletResponse response) {

      String url= request.getRequestURI().substring(
                            context.getContextPath().length());
      String query = request.getQueryString();
      if(query != null) {
         url = url + "?" + query;
      }

      FlashMap flashMap = new FlashMap();
      flashMap.put("url", url);
      FlashMapManager flashMapManager =
                    RequestContextUtils.getFlashMapManager(request);
      flashMapManager.saveOutputFlashMap(flashMap,
                                         request, response);
   }
}
```

인터셉터를 이용한 로그인 처리

❖ Login 모델

```
@Data
public class Login {
   @NotEmpty(message="사용자 ID는 필수 항목입니다.")
   private String userId;

   @NotEmpty(message="비밀 번호는 필수 항목입니다.")
   private String password;

   private String url;
}
```

인터셉터를 이용한 로그인 처리

❖ AccountController

```
@RequestMapping(value="/login", method=RequestMethod.GET)
public String loginForm(Login login,
                        @ModelAttribute("url") String url) {
   login.setUrl(url);
   return "account/login";
}
```

인터셉터를 이용한 로그인 처리

❖ login.jsp

```
<div class="col-md-6 offset-md-3">
   <h1 class="my-5">로그인</h1>

   <c:if test="${not empty login.url}">
      <div class="alert alert-warning">
        <strong>로그인이 필요한 서비스 입니다.</strong>
        </div>
   </c:if>

   <form:form commandName="login">
      <form:hidden path="url"/>
         :
```

인터셉터를 이용한 로그인 처리

❖ AccountController

```
@RequestMapping(value="/login", method=RequestMethod.POST)
public String loginSubmit(Login login,
              HttpSession session, Model model) throws Exception{
   try {
      Member member = service.login(login.getUserId(),
                  login.getPassword());
      // 성공하면
      session.setAttribute("USER", member);

      String url = login.getUrl();
      if(url!=null && !url.isEmpty()) return "redirect:"+url;

      return "redirect:/";
   } catch (LoginFailException e) {
      // 실패하면
      model.addAttribute("error", e.getMessage());
      return "account/login";
   }
}
```

인터셉터를 이용한 관리자 권한 처리

❖ **edu.iot.app.interceptor.AdminInterceptor**

```
@Component
public class AdminInterceptor extends HandlerInterceptorAdapter {
   @Autowired
   ServletContext context;

   @Override
   public boolean preHandle(HttpServletRequest request,
                HttpServletResponse response, Object handler)
         throws Exception {
      HttpSession session = request.getSession();
      Member member = (Member)session.getAttribute("USER");

      if(member.getGrade() != UserLevel.ADMIN) {
         String url = context.getContextPath() + "/account/login";
         response.sendRedirect("url");
         return false;
      }
      return super.preHandle(request, response, handler);
   }
}
```