

Form Validation

@Valid

- ▶ 스프링에서는 서블릿 2.3 표준 스펙 중 JSR-303 Validator를 확장하고 Annotation 기반으로 제공하고 있다
- ▶ mvc:annotation-drive을 이용, 간단하게 Bean Validation을 사용할 수 있다
- ▶ 검증을 위한 Bean에는 @Valid를 사용하여 자동 검증을 하게 된다
- ▶ 실제 검증은 모델 오브젝트에 달린 제약 조건 Annotation을 이용해 검증 작업이 이루어진다

Constraint Annotation

▶ JSR 303 Spec 기본 제공

- `@AssertFalse` : 거짓인가?
- `@Max` : 지정 값 이하인가?
- `@AssertTrue` : 참인가?
- `@Min` : 지정 값 이상인가?
- `@DecimalMax` : 지정 값 이하 실수인가?
- `@NotNull` : Null이 아닌가?
- `@DecimalMin` : 지정 값 이상 실수인가?
- `@Null` : Null인가?
- `@Digits(integer=, fraction=)` : 대상 수가 지정된 정수, 소수 자리 수 이내인가?
- `@Pattern(regex=, flag=)` : 정규식을 만족하는가?
- `@Future` : 날짜가 미래인가?
- `@Past` : 날짜가 과거인가?
- `@Size(min=, max=)` : 문자열, 배열 등의 크기가 지정 크기를 만족하는가?

Constraint Annotation

▶ Hibernate 제공

- @NotEmpty : Empty 값이 아닌가?
- @Email : 이메일 형식인가?
- @URL : URL 형식인가?
- @Length(min=, max=) : 문자열 길이 min과 max 사이인가?
- @Range(min=, max=) : 숫자 범위 체크

Validation 예제

: mysite에 Validation 추가

▶ Dependency 추가

```
<dependencies>
  <!-- Validation -->
  <dependency>
    <groupId>javax.validation</groupId>
    <artifactId>validation-api</artifactId>
    <version>1.0.0.GA</version>
  </dependency>

  <!-- Validation for Hibernate -->
  <dependency>
    <groupId>org.hibernate</groupId>
    <artifactId>hibernate-validator</artifactId>
    <version>4.2.0.Final</version>
  </dependency>
</dependencies>
```

Validator 예제

: Validation 체크

▶ Controller에서 Validation 체크

```
@RequestMapping(value="/join", method=RequestMethod.GET)
public String join(@ModelAttribute @Valid UserVo vo, BindingResult result) {
    if (result.hasErrors()) {
        //에러 출력
        List<ObjectError> list = result.getAllErrors();
        for (ObjectError e: list) {
            System.err.println("Object Error: " + e);
        }
        return "user/join";
    }
    userService.join(vo);
    return "redirect:/user/joinsuccess";
}
```

Validator 예제

: Validation 체크

▶ DTO에 Validation 적용

- ▶ 실제 Validation은 모델 내에서 수행됨

```
public class UserVo {  
    private Long no;  
  
    @NotEmpty  
    @Length(min = 2, max = 8)  
    private String name;  
  
    @NotEmpty  
    @Email  
    private String email;  
}
```

Validator 예제

: Validation 체크

▶ DTO에 Validation 적용

- ▶ 모델 필드의 제약조건을 만족하지 않은 경우, 콘솔에 출력되는 에러 정보를 확인
- ▶ 에러 발생시 필요한 조치를 취함 (예: 입력 폼으로 포워딩)

```
@RequestMapping(value="/join", method=RequestMethod.GET)  
public String join( @ModelAttribute @Valid UserVo vo, BindingResult result, Model model ) {  
    if ( result.hasErrors() ) {  
        model.addAllAttributes( result.getModel() );  
        return "/user/joinform";  
    }  
    userService.join(vo);  
    return "redirect:/user/joinsuccess";  
}
```


Validator 예제

: Validation Error 메시지 출력

- ▶ 입력폼(JSP)에서 에러 메시지 출력하기

- ▶ 태그 라이브러리 추가

```
<%@ taglib uri="http://www.springframework.org/tags" prefix="spring" %>
```

- ▶ 메시지 출력

```
<spring:hasBindErrors name="userVo">
  <c:if test="${errors.hasFieldErrors('name')}">
    <strong>${errors.getFieldError('name').defaultMessage}</strong>
  </c:if>
</spring:hasBindErrors>
```

Validator 예제

: Validation Error 메시지 출력

▶ 커스텀 메시지

▶ MessageSource 추가

```
<!-- MessageSource -->
<bean id="messageSource"
      class="org.springframework.context.support.ResourceBundleMessageSource">
  <property name="basenames">
    <list>
      <value>messages/messages_ko</value>
    </list>
  </property>
</bean>
```

▶ classpath:messages/messages_ko-properties 파일에 에러 메시지 작성

```
NotEmpty.userVo.name = ...
NotEmpty.userVo.email = ...
Email.userVo.email = ...
```

Validator 예제

: Validation Error 메시지 출력

▶ 커스텀 메시지 출력 (JSP)

```
<spring:hasBindErrors name="userVo">
  <c:if test="${errors.hasFieldErrors('email') }">
    <strong style="color:red">
      <spring:message
        code="${errors.getFieldError( 'email' ).codes[0] }"
        text="${errors.getFieldError( 'email' ).defaultMessage }" />
    </strong>
  </c:if>
</spring:hasBindErrors>
```

Validator 예제

▶ Form Tag Library 활용(JSP)

```
<%@ taglib uri="http://www.springframework.org/tags/form" prefix="form" %>
```

```
<form:form modelAttribute="userVo" .... >

    ....
    <input id="name" name="name" type="text" value="">
    <form:errors path="name" />

    ...
    ...

</form:form>
```