Validation(데이터 검증)

global빈등록 (별도등록해야 동작함)
모든 컨트롤에서 사용하는 검증기(객체검증기)

```
<annotation-driven   validator="globalvalidator" />
<beans:bean  id="globalvalidator"  class="com.acorn.validation.GlobalValidator" />
```

servlet_context.xml

```
<!--   error_message.properties   정보 등록   -->
<beans:bean        id="messageSource"  class="org.springframework.context.support.ResourceBundleMessageSource">
        <beans:property  name="basenames">
                <beans:list>
                        <beans:value>error_message</beans:value>
<!--  /src/main/resources/error_message.properties  -->
                </beans:list>
        </beans:property>
        <beans:property  name="defaultEncoding"  value="UTF-8"/>
</beans:bean>
```

error_message.properties

위치확인:
src/main/resources/error_message.properties

```
required=필수입력입니다.
required.user.pwd=사용자  비밀번호는  필수  항목입니다.
invalidLength.id=아이디의  길이는  {0}~{1}사이어야  합니다.
```

properties파일 한글설정
window->preferences->general->content types -> Text선택
UTF-8설정

```
<dependency>
    <groupId>javax.validation</groupId>
    <artifactId>validation-api</artifactId>
    <version>2.0.1.Final</version>
</dependency>

<!--  https://mvnrepository.com/artifact/org.hibernate/hibernate-validator  -->
<dependency>
    <groupId>org.hibernate</groupId>
    <artifactId>hibernate-validator</artifactId>
    <version>5.4.2.Final</version>
</dependency>
```

```
import javax.validation.Valid;

import org.springframework.stereotype.Controller;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.WebDataBinder;
import org.springframework.web.bind.annotation.InitBinder;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;


@Controller
public class MyController {
	@InitBinder
	public void init(WebDataBinder binder) {
		binder.setValidator(new UserValidator());    // Controller 단위의 Validator 등록
		// binder.addValidators(new UserValidator());     global Validator를 사용할때는  꼭 add 해야함
	}
	@RequestMapping(value="/valid2"  , method=RequestMethod.GET)
	public String  valid() {
		return "reg";
	}

	@RequestMapping(value="/valid2"  , method=RequestMethod.POST)
	public String  valid2( @Valid  User user ,  BindingResult result ) {
		System.out.println( result);
		System.out.println( user.getId());
		if( result.hasErrors())
			return "reg";

		return "ok";
	}
}
```

validator의 결과가 담긴다

```
참고 : UserValidator 객체 직접 생성하여 사용
UserValidator userValidator  = new UserValidator();
userValidator.validate(user, result);
if( result.hasErrors()) {
  return "registerForm";
}
```

```
public class User {

	String id;
	String pw;
	public String getId() {
		return id;
	}
	public void setId(String id) {
		this.id = id;
	}
	public String getPw() {
		return pw;
	}
	public void setPw(String pw) {
		this.pw = pw;
	}

}
```

```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
<%@ taglib uri="http://www.springframework.org/tags/form" prefix="form" %>

<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>


<!--<form action="/validation/valid2" method="post">-->
 <form:form modelAttribute="user">
    <div id="msg" class="msg"><form:errors path="id"/></div>
    <input type="text" name="id">
    <input type="text" name="pw">
    <button>로그인</button>
 </form:form>
</body>
</html>
```

```
path:속성명 ,입력정보가 그대로 출력됨
<form:input path="id"/>
<form:input path="pw"/>
```

```java
package com.acorn.validation;

import org.springframework.validation.Errors;
import org.springframework.validation.ValidationUtils;
import org.springframework.validation.Validator;

public class UserValidator implements Validator {
	@Override
	public boolean supports(Class<?> clazz) {
//		return User.class.equals(clazz); // 검증하려는 객체가 User타입인지 확인
		return User.class.isAssignableFrom(clazz); // clazz가 User 또는 그 자손인지 확인
	}

	@Override
	public void validate(Object target, Errors errors) {
		System.out.println("UserValidator.validate() is called");
		User user = (User)target;
		String id = user.getId();
//		if(id==null || "".equals(id.trim())) {
//			errors.rejectValue("id", "required");
//		}
		ValidationUtils.rejectIfEmptyOrWhitespace(errors, "id", "required");
		ValidationUtils.rejectIfEmptyOrWhitespace(errors, "pw", "required");

		if(id==null || id.length() < 5 || id.length() > 12) {
			errors.rejectValue("id", "invalidLength", new String[]{"5","12"}, null);
		}
	}
}
```

```
Validator 작성
:객체를 검증하기 위한 인터페이스

public interface Validator{
    boolean supports( Class<?> clazz) ;  //검증기로 검증가능한 객체인지
    void validate( Object target , Errors erros );
   //객체를 검증하는 매서드 , target:검증할 객체 , errors: 검증시 발생한 에러 저장소
}
```