Bean Validation with Spring

# Maven (pom.xml)

<project xmlns=*"http://maven.apache.org/POM/4.0.0"* xmlns:xsi=*"http://www.w3.org/2001/XMLSchema-instance"*

xsi:schemaLocation=*"http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd"*>

<modelVersion>4.0.0</modelVersion>

<groupId>springvalidator1</groupId>

<artifactId>springvalidator1</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<name>springvalidator1</name>

<url>http://maven.apache.org</url>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

</properties>

<dependencies>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-core</artifactId>

<version>4.0.5.RELEASE</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context-support</artifactId>

<version>4.0.5.RELEASE</version>

</dependency>

**<dependency>**

**<groupId>org.springframework</groupId>**

**<artifactId>spring-context</artifactId>**

**<version>4.0.5.RELEASE</version>**

**</dependency>**

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-beans</artifactId>

<version>4.0.5.RELEASE</version>

</dependency>

<dependency>

<groupId>javax.validation</groupId>

<artifactId>validation-api</artifactId>

<version>1.1.0.Final</version>

</dependency>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>3.8.1</version>

<scope>test</scope>

</dependency>

</dependencies>

</project>

# Type – 1 – Plain Bean Validation

## Person.java

**package** com.ddlab.rnd.spring;

**public** **class** Person {

**private** String name;

**private** **int** age;

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** **int** getAge() {

**return** age;

}

**public** **void** setAge(**int** age) {

**this**.age = age;

}

}

## PersonValidator.java

**package** com.ddlab.rnd.spring;

**import** org.springframework.validation.Errors;

**import** org.springframework.validation.ValidationUtils;

**import** org.springframework.validation.Validator;

**public** **class** PersonValidator **implements** Validator {

/\* (non-Javadoc)

\* @see org.springframework.validation.Validator#supports(java.lang.Class)

\*/

**public** **boolean** supports(Class<?> clazz) {

**return** Person.**class**.equals(clazz);

}

/\* (non-Javadoc)

\* @see org.springframework.validation.Validator#validate(java.lang.Object, org.springframework.validation.Errors)

\*/

**public** **void** validate(Object object, Errors errors) {

ValidationUtils.*rejectIfEmpty*(errors, "name", "name can not be blank");

Person p = (Person) object;

**if**(p.getAge() <= 0 )

errors.rejectValue("age", "Age of Person can not be 0 or negative");

}

}

## App.java (Test Program)

**package** com.ddlab.rnd.spring;

**import** java.util.Iterator;

**import** java.util.List;

**import** org.springframework.util.StringUtils;

**import** org.springframework.validation.BindException;

**import** org.springframework.validation.ObjectError;

**import** org.springframework.validation.ValidationUtils;

**public** **class** App

{

**public** **static** **void** main( String[] args ) **throws** Exception

{

Person person = **new** Person();

// person.setName("Deb");

// person.setAge(-29);

PersonValidator pv = **new** PersonValidator();

**final** BindException errors = **new** BindException(person, person.getClass().getName());

ValidationUtils.*invokeValidator*(pv, person, errors);

**if** (!errors.hasErrors())

{

// there are no validation errors

System.***out***.println("the object is valid");

}

**else** {

**final** StringBuffer errorDescription = **new** StringBuffer("object Person had the following validation errors\n");

**final** List errorsList = errors.getAllErrors();

Iterator errorItem = errorsList.iterator();

**while** (errorItem.hasNext()) {

**final** ObjectError error = (ObjectError) errorItem.next();

errorDescription.append('[');

errorDescription.append(error.getCode());

**final** String defaultMessage = error.getDefaultMessage();

**if** (StringUtils.*hasText*(defaultMessage)) {

errorDescription.append('=');

errorDescription.append(defaultMessage);

}

errorDescription.append(']');

**if** (errorItem.hasNext())

errorDescription.append(',');

}

System.***out***.println(errorDescription.toString());

**throw** errors;

}

}

}

# Output

object Person had the following validation errors

[name can not be blank],[Age of Person can not be 0 or negative]

Exception in thread "main" org.springframework.validation.BindException: org.springframework.validation.BeanPropertyBindingResult:

# Type- 2 with Nested Object

## Address.java

**package** com.ddlab.rnd.nested.objects;

**public** **class** Address {

**private** String cityName;

**private** String pinCode;

**public** String getCityName() {

**return** cityName;

}

**public** **void** setCityName(String cityName) {

**this**.cityName = cityName;

}

**public** String getPinCode() {

**return** pinCode;

}

**public** **void** setPinCode(String pinCode) {

**this**.pinCode = pinCode;

}

}

## Employee.java

**package** com.ddlab.rnd.nested.objects;

**import** com.ddlab.rnd.spring.Person;

**public** **class** Employee **extends** Person {

**private** Address adrs;

**public** Address getAdrs() {

**return** adrs;

}

**public** **void** setAdrs(Address adrs) {

**this**.adrs = adrs;

}

}

## AddressValidator.java

**package** com.ddlab.rnd.nested.objects;

**import** org.springframework.validation.Errors;

**import** org.springframework.validation.ValidationUtils;

**import** org.springframework.validation.Validator;

**public** **class** AddressValidator **implements** Validator {

**public** **boolean** supports(Class<?> clazz) {

**return** Address.**class**.equals(clazz);

}

**public** **void** validate(Object object, Errors errors) {

ValidationUtils.*rejectIfEmpty*(errors, "cityName", "cityName can not be blank");

ValidationUtils.*rejectIfEmpty*(errors, "pinCode", "pinCode can not be blank");

}

}

## EmployeeValidator.java

**package** com.ddlab.rnd.nested.objects;

**import** org.springframework.validation.Errors;

**import** org.springframework.validation.ValidationUtils;

**import** org.springframework.validation.Validator;

**public** **class** EmployeeValidator **implements** Validator {

**private** **final** AddressValidator adrsValidator;

**public** EmployeeValidator(AddressValidator adrsValidator) {

**this**.adrsValidator = adrsValidator;

}

**public** **boolean** supports(Class<?> clazz) {

**return** Employee.**class**.equals(clazz);

}

**public** **void** validate(Object object, Errors errors) {

ValidationUtils.*rejectIfEmpty*(errors, "name", "name can not be blank");

Employee emp = (Employee) object;

**if**(emp.getAge() <= 0 )

errors.rejectValue("age", "Age of Person can not be 0 or negative");

**try** {

errors.pushNestedPath("adrs");

ValidationUtils.*invokeValidator*(**this**.adrsValidator, emp.getAdrs(), errors);

} **finally** {

errors.popNestedPath();

}

}

}

## TestEmployee.java

**package** com.ddlab.rnd.nested.objects;

**import** java.util.Iterator;

**import** java.util.List;

**import** org.springframework.util.StringUtils;

**import** org.springframework.validation.BindException;

**import** org.springframework.validation.ObjectError;

**import** org.springframework.validation.ValidationUtils;

**public** **class** TestEmployee {

**public** **static** **void** main(String[] args) **throws** Exception {

Employee emp = **new** Employee();

EmployeeValidator pv = **new** EmployeeValidator(**new** AddressValidator());

**final** BindException errors = **new** BindException(emp, emp.getClass().getName());

ValidationUtils.*invokeValidator*(pv, emp, errors);

**if** (!errors.hasErrors())

{

// there are no validation errors

System.***out***.println("the object is valid");

}

**else** {

**final** StringBuffer errorDescription = **new** StringBuffer("object Person had the following validation errors\n");

**final** List errorsList = errors.getAllErrors();

Iterator errorItem = errorsList.iterator();

**while** (errorItem.hasNext()) {

**final** ObjectError error = (ObjectError) errorItem.next();

errorDescription.append('[');

errorDescription.append(error.getCode());

**final** String defaultMessage = error.getDefaultMessage();

**if** (StringUtils.*hasText*(defaultMessage)) {

errorDescription.append('=');

errorDescription.append(defaultMessage);

}

errorDescription.append(']');

**if** (errorItem.hasNext())

errorDescription.append(',');

}

System.***out***.println(errorDescription.toString());

**throw** errors;

}

}

}

# Output

Exception in thread "main" object Person had the following validation errors

[name can not be blank],[Age of Person can not be 0 or negative],[cityName can not be blank],[pinCode can not be blank]

org.springframework.validation.BindException: org.springframework.validation.BeanPropertyBindingResult: 4 errors