**数据验证概述：**

用户的输入数据时带有随意性，为了保证数据的合法性，必须对数据进行验证。

1. 数据验证分为客户端验证和服务端验证，客户端验证主要是过滤正常用户的误操作， 通过JavaScript代码完成；但仅有客户端验证是不够的，攻击者可以绕过客户端验证 直接进行非法操作，为了确保数据的合法性，必须加上服务端验证。
2. 在Spring MVC框架中可以利用Spring自带的验证框架验证数据，也可以利用JSR303 验证数据。

**Spring验证器：**

Spring验证器主要实现Validator接口和ValidationUtils类、以及属性文件来实现的。

**Validator接口：**

创建自定义Spring验证器需要实现org.springframework.validation.Validator接口，该接 有两个接口方法：

boolean supports ( Class<?> klass )

void validate ( Object object ,Errors errors )

supports( )方法返回true时，验证器可以处理指定的Class

validate( )方法的功能是验证目标对象object中包含的数据是否合法，不合法则将验证 错误信息存入Errors对象。

往Errors对象存入错误信息的方法是reject( )或rejectValue( )方法，这两个方法的部分重 载如下：

void reject ( String errorCode )

void reject( String error , String defaultMessage )

void rejectValue( String field , String errorCode )

void rejectValue( String field , String errorCode , String defaultMessage )

在一般情况下只需要给reject( )或rejectValue( )方法一个错误代码，Spring MVC会在消息 属性文件中查找错误代码，获取相应的错误信息。

if ( goods.getPrice( ) > 100 || goods.getPrice( ) < 0 ) {

Errors.rejectValue ( “gprice” , “gprice.invalid “ ) ;

}

以上代码表示如果goods的价格大于100或小于0，则从属性文件中取出错误提示信息 并存入errors对象中。

**ValidationUtils类：**

org.springframework.validation.ValidationUtils是一个工具类，在该类中有几个方法可以帮 助用户判定值是否为空。

1）例如：

if (goods.getName==null||goods.getName( ).isEmpty( ) ){

Errors.rejectValue(“gname”,”goods.gname.required” ) ;

}

使用ValidationUtils类的rejectIfEmpty( )方法简化后如下：

ValidationUtils.rejectIfEmpty ( errors , “gname” , “goods.gname.reqrired “ ) ;

2）再例如：

if ( goods.getName==null || goods.getName( ).trim( ) .isEmpty( ) ) {

Errors.rejecValue ( “name” , “goods.name.required “) ;

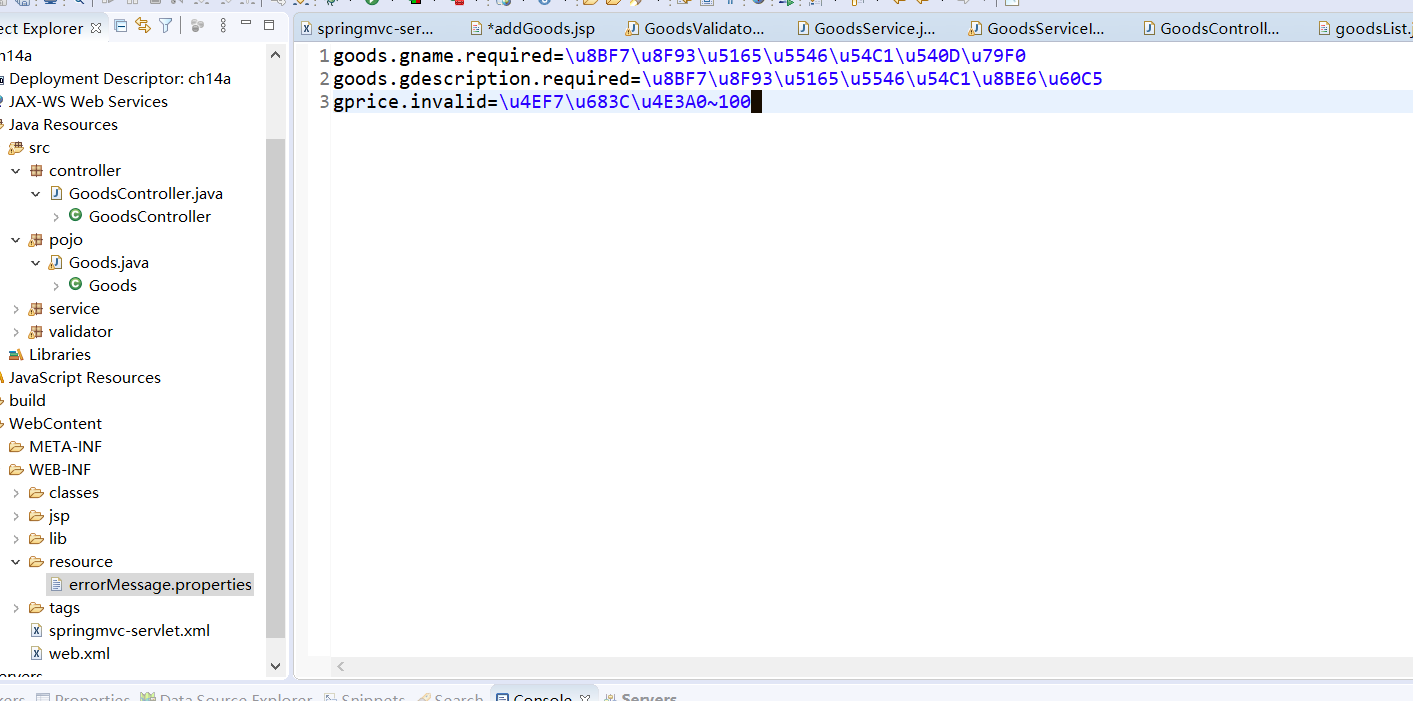
}

使用ValidationUtils类的rejectIfEmptyOrWhitespace( )方法简化后如下：

ValidationUtils.rejectIfEmptyOrWhitespace (error , “name” , “goods.name.required” ) ;

**错误消息属性文件：**

在WEB-INF目录下创建文件夹resource，并在该文件夹中编写属性文件 errorMessages.properties，该属性文件中存放的是错误消息，供验证器使用，当验证器 中验证数据出现了非法，则根据其数据名在此文件中找到相应属性所对应的错误信息， 将其添加到Errors对象中。例如：



在属性文件中，如果包含中文的话，需要进行解码，转换成Unicode编码，但在Eclips e中编写properties属性文件时，会自动进行转码。

**配置消息属性文件：**

消息属性文件编写完成后，如果在程序中验证数据时需要使用到该属性文件中的信息， 则必须还要在Spring MVC中的配置文件中进行配置。如下：



**JSR 303验证简介：**

1. 使用JSR303进行数据验证，有两种实现，一是Hibernate、另一个是Apache BVal。
2. 在此讲解Hibernate Validator实现，需要注意的是它与Hibernate框架无关，我们 只是用它来进行数据验证。

Hibernate Validator的下载安装与配置：

1. 下载安装

用户可以通过以下地址下载Hibernate Validator：

<https://sourceforge.net/projects/hibernate/files/hibernate-validator/>

下载完成后，将\dist\lib\required目录下的hibernate-validator-5.4.0.Final.jar和 \dist\lib\required目录下的classmate-1.3.1.jar、javax.el-3.0.1-b08.jar、 jboss-loggin-3.3.0.Final.jar、validation-api-1.1.0.Final.jar复制到\WEB-INF\lib目录中。

1. 配置：

如果将验证错误信息放在属性文件中，那么需要在配置文件中配置属性文件，并将 属性文件与Hibernate Validator关联，具体配置如下。

**Hibernate Validator常用注解：**

JSR303不需要编写验证器，而是使用一系列的注解来完成数据验证。将注解标注在 领域模型的属性上，等到要验证时使用@Valid注解进行验证。

1. 空检查：

·@Null：验证对象是否为null（无法检查长度为0的字符串）

·@NotNull：验证对象是否不为null（无法检查长度为0的字符串）

·@NotBlack：检查约束字符串是否为null，以及被trim( )后的长度是否大于0，只 针对字符串类型

·@NotEmpty：检查约束元素是否为null或者是empty

1. boolean检查：

·@AssertTrue：验证boolean属性是否为true

·@AssertFalse：验证boolean属性是否为false

1. 长度检查：

·@Size ( min= , max= )：验证对象长度是否在给定范围内，对象可以是Array、 Collection、Map、String类型

·@Length ( min= , max= )：验证字符串长度是否在给定的范围之内

1. 日期检查：  
    ·@Past：验证Date和Calendar对象是否在当前时间之前

·@Future：验证Date和Calendar对象是否在当前时间之后

·@Patter：验证String对象是否符合正则表达式的规则

1. 数值检查

·@Min：验证Number和String对象是否大于指定的值

·@Max：验证Number和String对象是否小于指定的值

·@DecimalMax：被标注的值必须不大于约束中指定的最大值，这个约束的参 数是一个通过BigDecimal定义的最大值的字符串表示，小数 存在精度

·@DecimalMin：被标注的值必须不小于约束中指定的最小值，这个约束的参 数时一个通过BigDecimal定义的最小值的字符串表示，小数 存在精度

·@Digits：验证Number和String的构成是否合法

·@Digits ( integer= , fraction= )：验证字符串是否符合指定格式的数字，integer 指定整数精度，fraction指定小数精度

·@Range( min=, max= )：检查数字是否介于min和max之间

·@Valid：对关联对象进行校验，如果关联对象是个集合或者数组，那么对其 中的元素进行校验，如果是一个map，则对其中的值部分进行校验

·@CreditCardNumber：信用卡验证

·@Email：验证是否为邮件地址，如果为null，不进行验证，通过验证