## 数据的验证：

客户端的验证，就是前端技术对输入的数据进行验证

服务器端的验证，先通过数据的类型转换或者格式化Converter或者Formatter然后再进行服务器端的验证。

## Jsr 303 Validation验证

规范的实现有两种：一个是apache，一个是Hibernate Validator(与ORM框架Hivernate没有任何关系)

使用：并不需要编写验证器，直接导入相关的依赖包，使用注解就可以

还有一种就是通过继承Validator这个接口，重写里面的两个方法（support方法和validate（Object obj Errors errors））

Support 返回true或者false，并不是所有的object都可以放进来

Errors具体实现类为BindingResult,

hasErrors() true false,验证通过为true，不通过为false

将错误封装到属性类中，FieldError field属性bingdingResult.getFieldError()获得

这个属性中有方法getcode（）方法和getField（）方法。

判断是使用基本的Java代码，进行逻辑判断

不符合条件使用errors的方法rejectValue（）

或者直接通过一个工具类，完成判断加上设置error

类ValidationUtils工具类，里面有许多静态方法。rejectIfEmpty( errors, field, errorCode)

## 通过注解的方式

规范JSR 303规范，实现为Hibernate Validator，使用之前要先引用相关的依赖，

实现逻辑判断，在需要判断的类中，比如前端传给后端的数据，把它抽象成一个Java类。

在这个类上使用注解。进行逻辑判断。

在传入的参数中，配合使用注解@Valid 对后面的参数进行验证，后面跟一个BuildingResult

存放判断的结果，然后在逻辑代码中看看是否通过

BingResult.hasErrors()

FieldError fieldError = bindingResult.getFieldError()

fieldError.getCode() 获得在注解中填入的message，fieldErrors.getField() 获得属性的名称。

常用的注解

@NULL 要为空

@NotNull 对象不为空就可以

@NotEmpty 用于集合，数组，里面元素个数不为空

@NotBlank 用于String，去掉空格之后，长度大于0

Boolean检查

@AssertTrue @AssertFalse

长度检查：

@Size（max= , min =） 用于collection，map，array，

@Length（max=, min=）用于检查字符串的长度范围

日期检查，

@Past（message = “不通过存入errors中的错误信息”）

@Future（）

@Pattern（）用于String对象，看看是否符合正则表达式规则。

数字的检查

@Min 验证int

@Max 验证int

@DecimalMin 验证BigDecimal

@DecimalMax 验证BigDecimal

@Range（max=, min=）检查数字在这个范围之间

@Valid，对对象进行检验，如果是集合，就对集合中的元素进行检验

@Digits 验证Number和String是否合法

@Digits（integer=整数精度， fraction=小数精度）

@Email验证是否为邮箱，为空的时候不验证，直接通过