Spring Boot进阶

校验

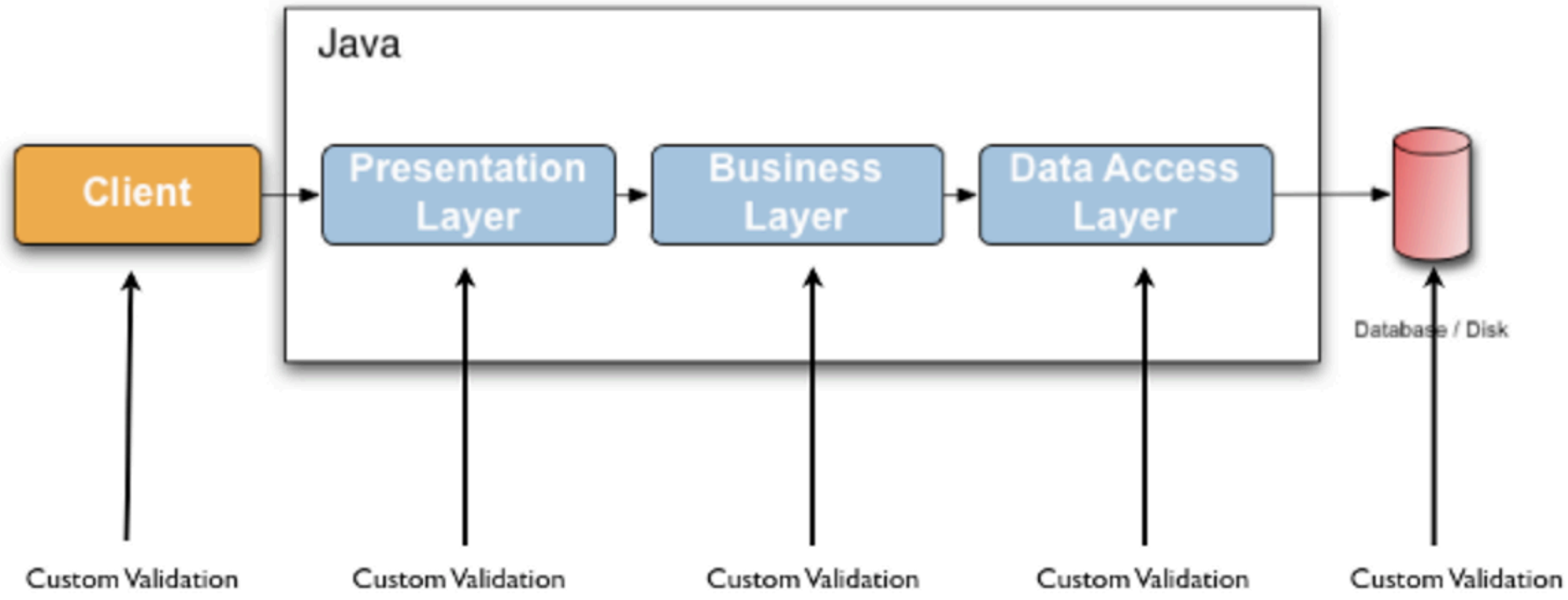
```java
@PostMapping("/user/save")
public User saveUser(@RequestParam("name") String name, @RequestParam("age")
Integer age) {
    if (null == name) {
        logger.info("校验失败：姓名不能为空");
        return null;
    }
    if (name.trim().length() < 3) {
        logger.info("校验失败：姓名长度不能小于3");
        return null;
    }
    if (age < 18) {
        logger.info("校验失败：年龄不能小于18岁");
        return null;
    }

    //...其他逻辑

    return null;
}
```

1. 参数扩展
2. 校验逻辑扩展
3. 容易出错

课程介绍

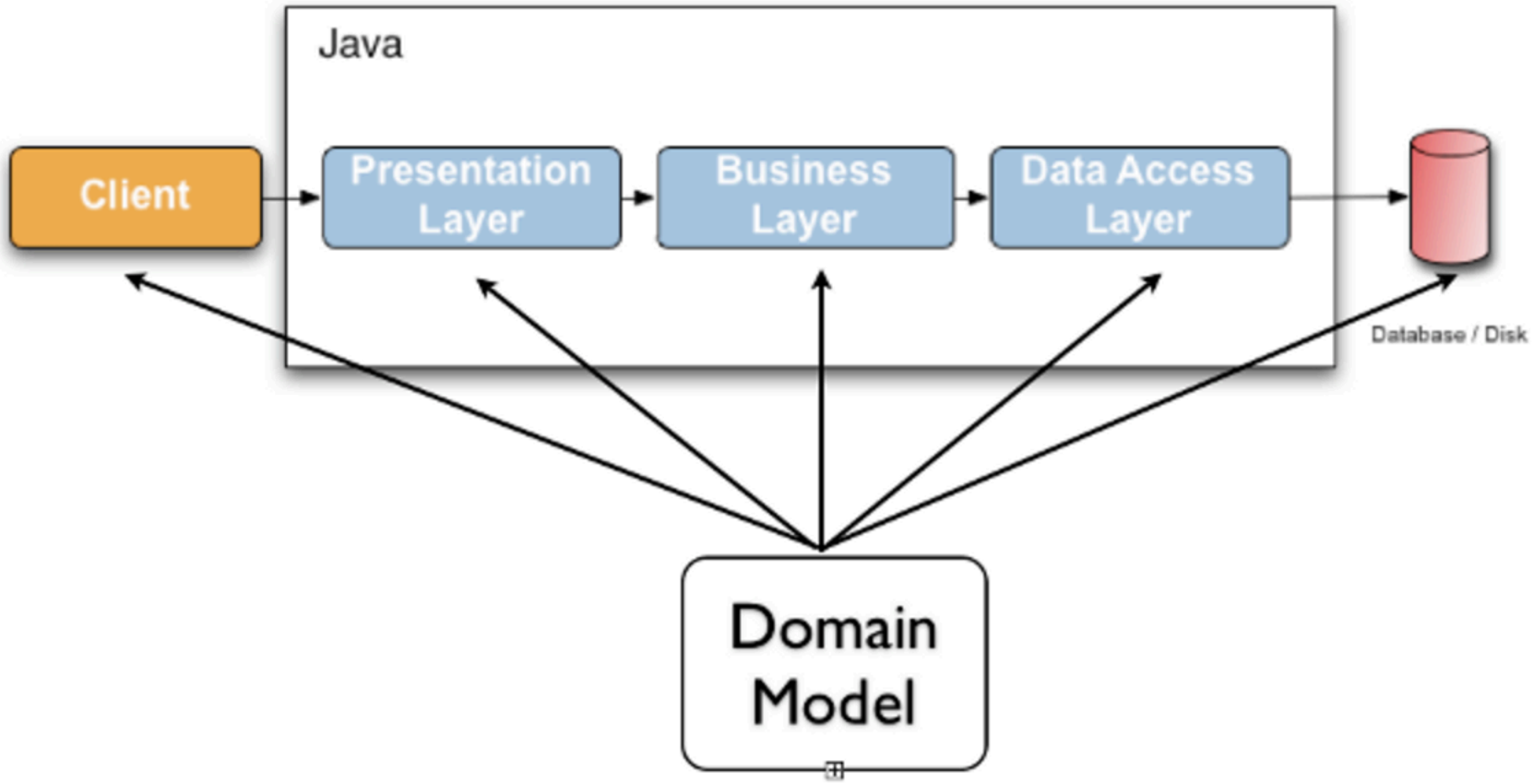
- JSR 303 Bean Validation & Hibernate Validator 简介
- 使用Hibernate Validator注解进行表单校验
- 分组校验
- 自定义hibernate validation注解

Bean Validation & Hibernate Validator 简介



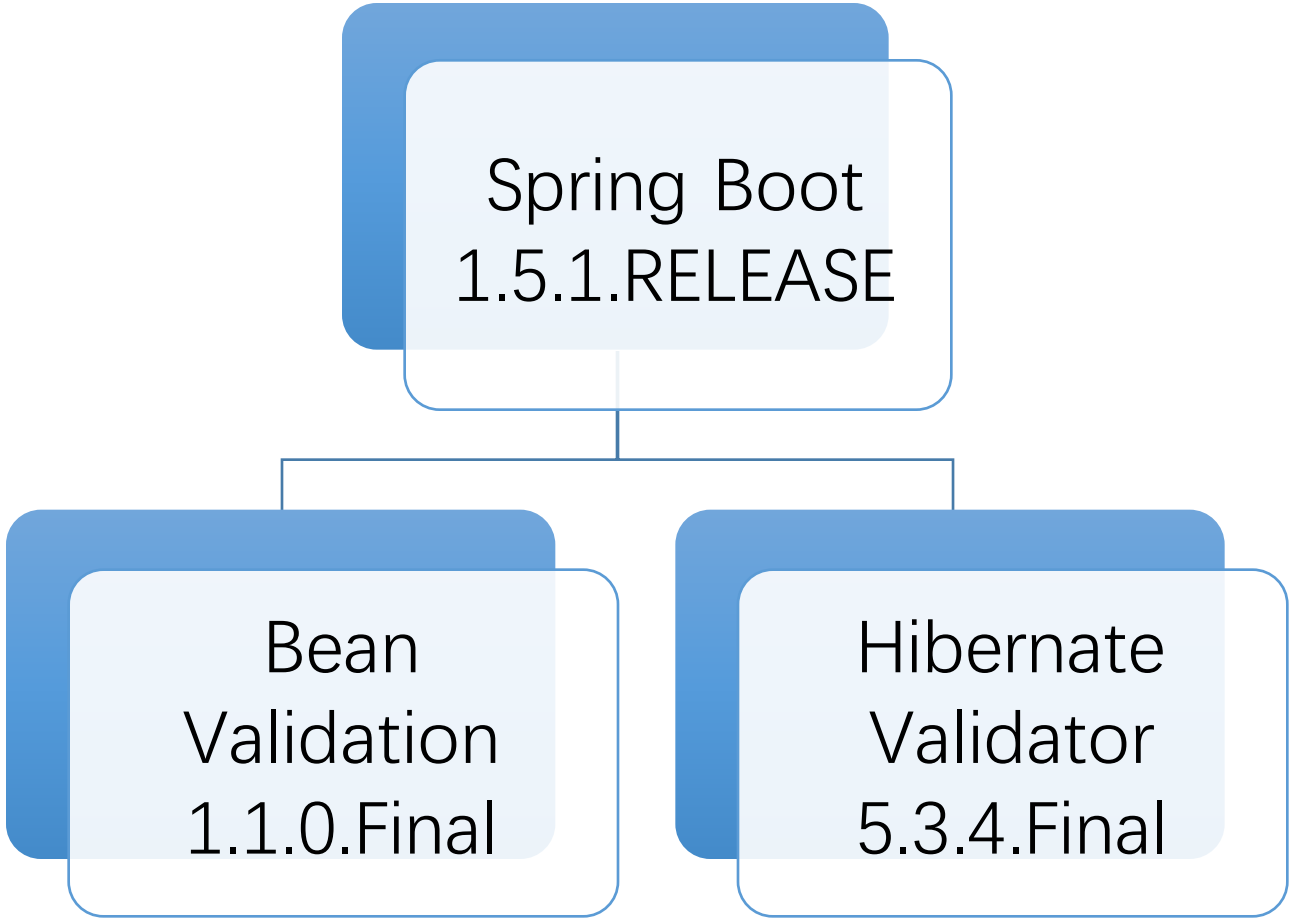
注：该图片来自于Hibernate Validator官网

Bean Validation & Hibernate Validator 简介



注：该图片来自于Hibernate Validator官网

Bean Validation & Hibernate Validator 简介



案例一：
使用Hibernate Validator注解进行表单校验

- 实现功能：添加用户
- 限制条件：
  - Id**必须**为空
  - 姓名不能为空
  - 年龄大于18岁

案例二：分组校验

- 实现功能：更新用户
- 限制条件：
  - Id不能为空
  - 姓名不能为空
  - 年龄大于18岁

案例三：自定义hibernate validation注解

- 实现功能：在用户表增加phone字段
- 限制条件：
  - 符合手机号格式

自定义注解约束的步骤
1. 创建注解 @Phone
2. 创建注解的验证类 class 并实现 ConstraintValidator

JSR-303 Bean Validation & Hibernate Validator

Constraint	详细信息	Constraint	详细信息
@Null	被注释的元素必须为 null	@Email	被注释的元素必须是电子邮箱地址
@NotNull	被注释的元素必须不为 null	@Length	被注释的字符串的大小必须在指定的范围内
@AssertTrue	被注释的元素必须为 true	@NotEmpty	被注释的字符串的必须非空
@AssertFalse	被注释的元素必须为 false	@Range	被注释的元素必须在合适的范围内
@Min(value)	被注释的元素必须是一个数字，其值必须大于等于指定的最小值		
@Max(value)	被注释的元素必须是一个数字，其值必须小于等于指定的最大值		

使用注意

1. 对于多个字段，系统验证的顺序和字段声明的顺序是不一样的
2. 对于每个验证如果没有message属性，系统会使用默认熟悉，如对应@NotBlank相当于@NotBlank(message= “不能为空”)
3. 对于引用类型如String等类型，一定要结合@NotNull、@NotEmpty或者@NotBlank来配合使用，如果不配合使用，经测试，该字段是不参与校验的，如单独使用@Pattern、@Length、@Email

- **容易混淆的3个注解**

  @NotNull： 任何对象的value不能为null

  @NotEmpty：集合对象的元素不为0，即集合不为空，也可以用于字符串不为null

  @NotBlank：只能用于字符串不为null，并且字符串trim()以后length要大于0

使用注意

1. @Valid不支持分组
2. 如果使用@Valid，那么属性中也不能指定，否则校验不生效
3. @Validated是@Valid 的一次封装，是spring提供的校验机制使用（org.springframework.validation.annotation.Validated），@Valid不提供分组功能
4. 分组就是一个空接口interface，用于分组名称

   @Validated({SAVE.class}) @NotNull(message = "id不能为空",groups = UPDATE.class)
5. 如果指定分组@Validated({SAVE.class})，那么属性中必须也同时指定，否则校验不生效
6. 不建议实现较为复杂的校验规则（CPU损耗）、不建议实现基于网络请求的校验（IO等待）

```java
@PostMapping("/user/save")
public User saveUser(@RequestParam("name") String name, @RequestParam("age")
Integer age) {
    if (null == name) {
        logger.info("校验失败：姓名不能为空");
        return null;
    }
    if (name.trim().length() < 3) {
        logger.info("校验失败：姓名长度不能小于3");
        return null;
    }
    if (age < 18) {
        logger.info("校验失败：年龄不能小于18岁");
        return null;
    }

    //...其他逻辑

    return null;
}
```

```java
@PostMapping("/user/save")
public User saveUser(@Validated(User.NAME.class) User user, BindingResult bindingResult) {
    if (bindingResult.hasErrors()) {
        logger.info("验证失败：{}",bindingResult.getFieldError().getDefaultMessage());
        return null;
    }
    user = userService.saveUser(user);
    return user;
}
```

代码地址： git@gitee.com:fairyside/springboot.git