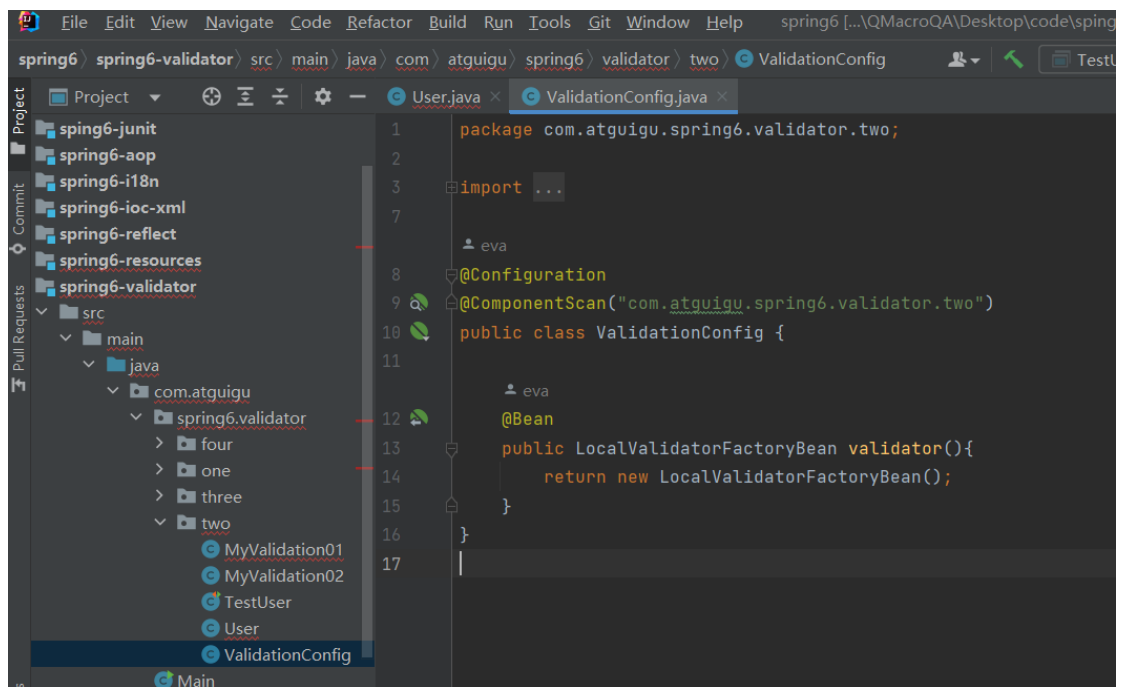


第一步 创建配置类，配置LocalValidatorFactoryBean

第二步 创建实体类，定义属性，生成get和set方法，在属性上面使用注解设置校验规则

第三步 创建校验器

第四步 完成最终测试

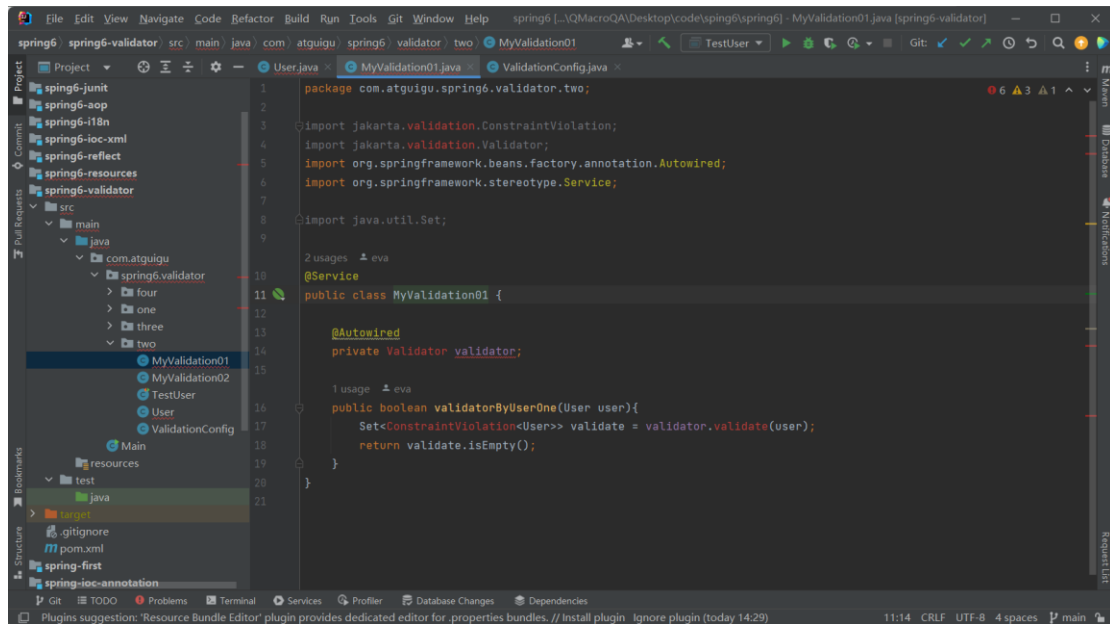


```
1 package com.atguigu.spring6.validator.two;
2
3 import jakarta.validation.constraints.Max;
4 import jakarta.validation.constraints.Min;
5 import jakarta.validation.constraints.NotNull;
6
7 public class User {
8     @NotNull
9     private String name;
10
11     @Min(0)
12     @Max(150)
13     private int age;
14
15     public String getName() { return name; }
16
17     public void setName(String name) { this.name = name; }
18 }
```

常用注解说明

- @NotNull 限制必须不为null
- @NotEmpty 只作用于字符串类型，字符串不为空，并且长度不为0
- @NotBlank 只作用于字符串类型，字符串不为空，并且trim()后不为空串
- @DecimalMax(value) 限制必须为一个不大于指定值的数字
- @DecimalMin(value) 限制必须为一个不小于指定值的数字
- @Max(value) 限制必须为一个不大于指定值的数字
- @Min(value) 限制必须为一个不小于指定值的数字
- @Pattern(value) 限制必须符合指定的正则表达式
- @Size(max,min) 限制字符串长度必须在min到max之间
- @Email 验证注解的元素值是Email，也可以通过正则表达式和flag指定自定义的email格式

校验器：原生



```
package com.atguigu.spring6.validator.two;

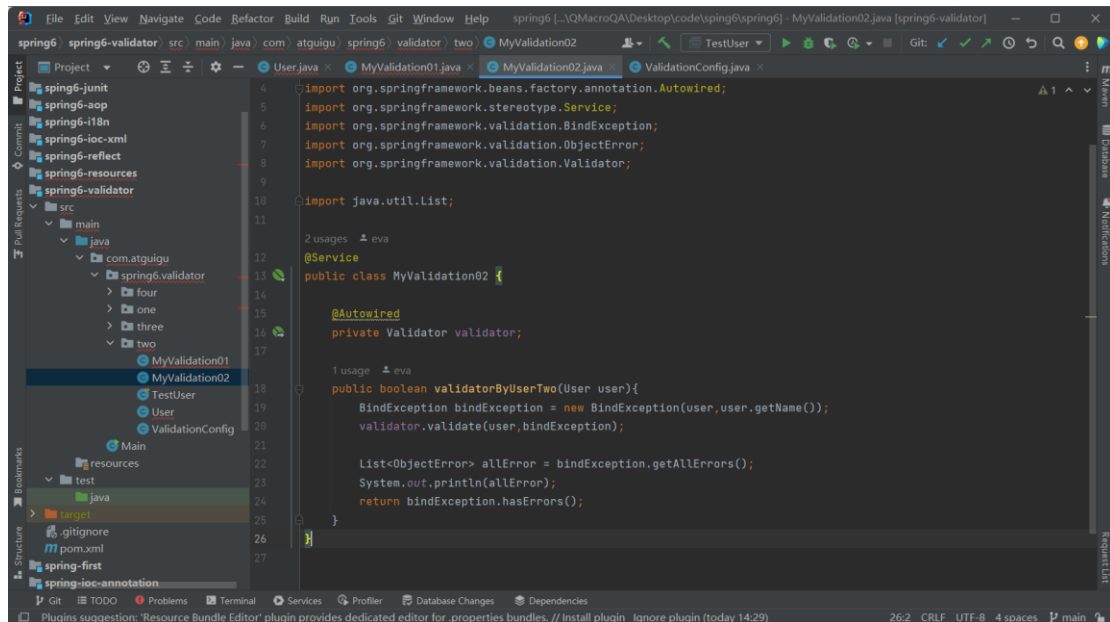
import jakarta.validation.ConstraintViolation;
import jakarta.validation.Validator;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import java.util.Set;

@Service
public class MyValidation01 {

    @Autowired
    private Validator validator;

    public boolean validatorByUserOne(User user){
        Set<ConstraintViolation<User>> validate = validator.validate(user);
        return validate.isEmpty();
    }
}
```

第二种:



```
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import org.springframework.validation.BindException;
import org.springframework.validation.ObjectError;
import org.springframework.validation.Validator;
import java.util.List;

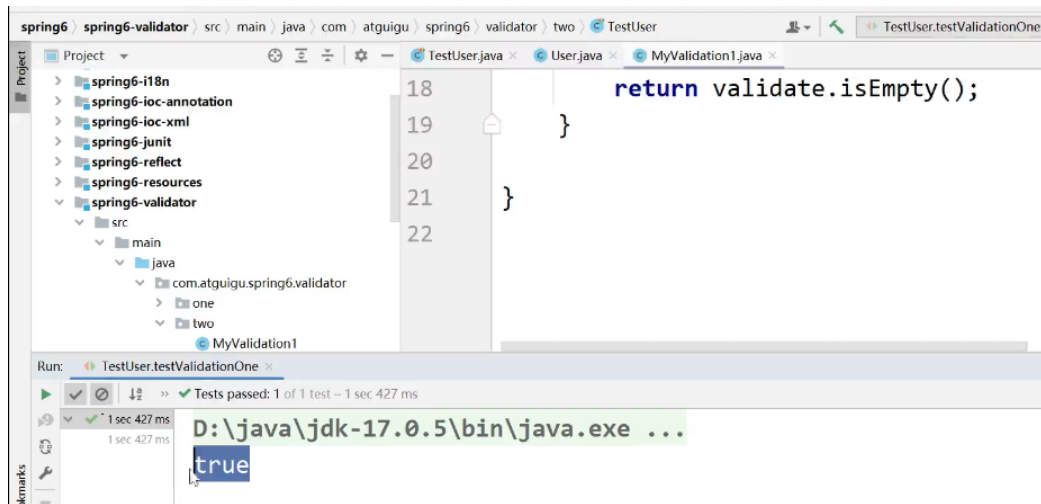
@Service
public class MyValidation02 {

    @Autowired
    private Validator validator;

    public boolean validatorByUserTwo(User user){
        BindException bindException = new BindException(user, user.getName());
        validator.validate(user, bindException);

        List<ObjectError> allError = bindException.getAllErrors();
        System.out.println(allError);
        return bindException.hasErrors();
    }
}
```

测试:



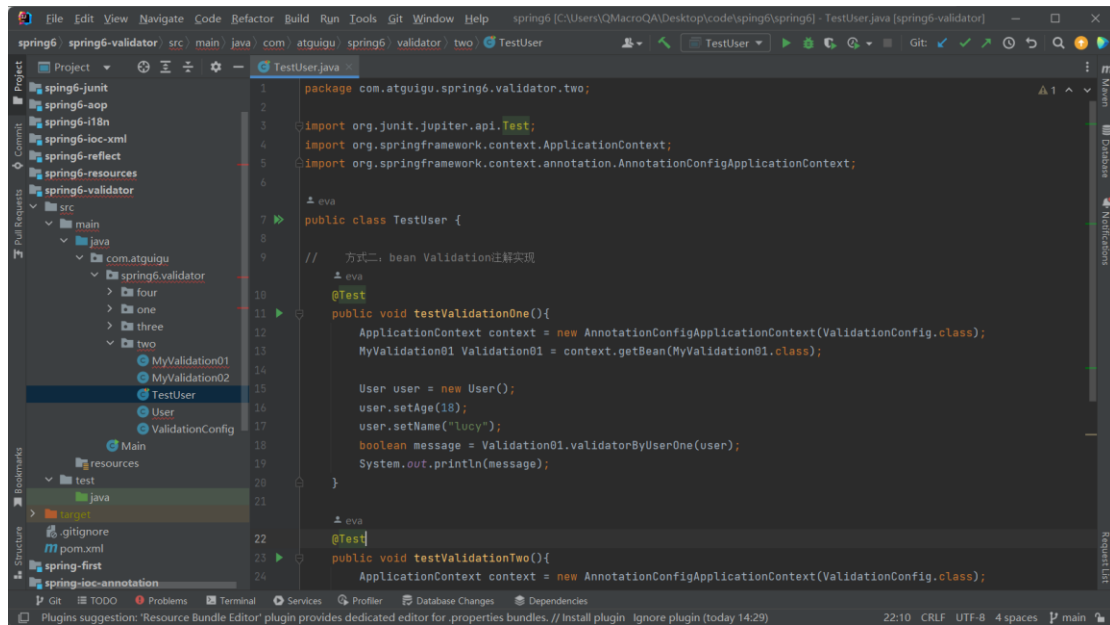
```
18 return validate.isEmpty();
19 }
20
21 }
22
```

Run: TestUser.testValidationOne

Tests passed: 1 of 1 test - 1 sec 427 ms

D:\java\jdk-17.0.5\bin\java.exe ...

true



```
1 package com.atguigu.spring6.validator.two;
2
3 import org.junit.jupiter.api.Test;
4 import org.springframework.context.ApplicationContext;
5 import org.springframework.context.annotation.AnnotationConfigApplicationContext;
6
7 public class TestUser {
8
9     // 方式二: bean Validation注解实现
10    @Test
11    public void testValidationOne(){
12        ApplicationContext context = new AnnotationConfigApplicationContext(ValidationConfig.class);
13        MyValidation01 Validation01 = context.getBean(MyValidation01.class);
14
15        User user = new User();
16        user.setAge(18);
17        user.setName("lucy");
18        boolean message = Validation01.validatorByUserOne(user);
19        System.out.println(message);
20    }
21
22    @Test
23    public void testValidationTwo(){
24        ApplicationContext context = new AnnotationConfigApplicationContext(ValidationConfig.class);
```

```
25        MyValidation02 Validation02 = context.getBean(MyValidation02.class);
26
27        User user = new User();
28        user.setAge(18);
29        user.setName("lucy");
30        boolean message = Validation02.validatorByUserTwo(user);
31        // 有错误信息报错, 没有错误信息返回false
32        System.out.println(message);
33    }
34 }
35 }
```