[Thực hành] Custom Validation

Mục tiêu

Luyện tập tạo các validator tuỳ biến.

 Mô tả

Trong phần này, chúng ta phát triển một chức năng cho phép validate số điện thoại của người dùng.

Ứng dụng cho phép nhập vào số điện thoại và hiển thị thông báo tương ứng với giá trị của số điện thoại.

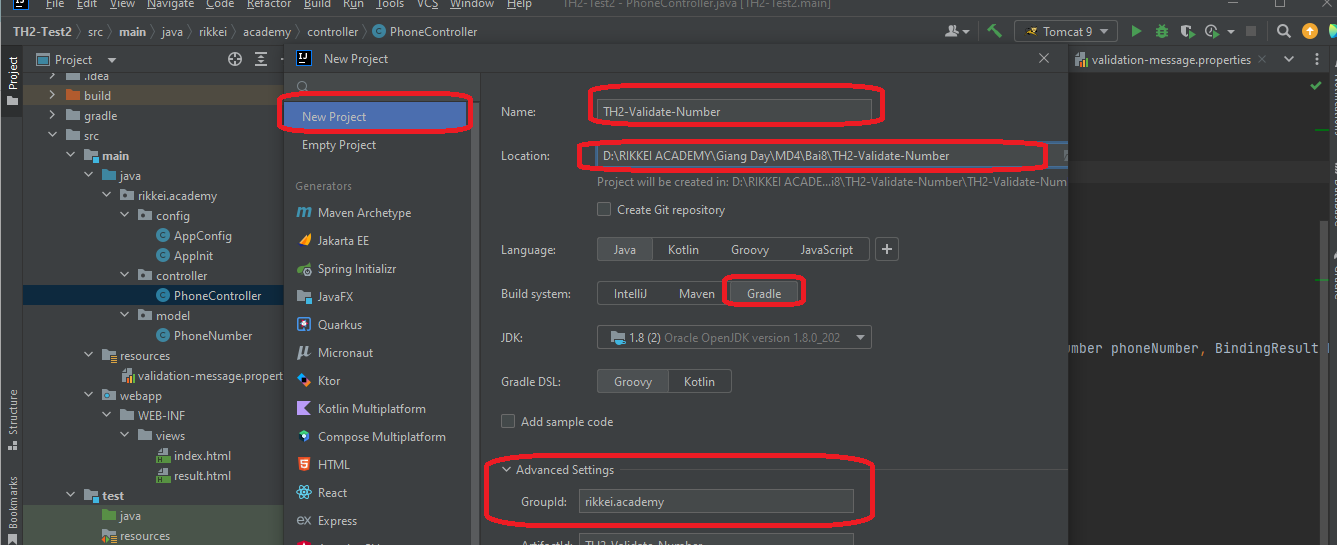
* Số điện thoại không được để trống
* Bắt đầu bằng số 0
* Có độ dài từ 10 -11 ký tự
* Chỉ bao gồm số

**Để hoàn thành bài thực hành, học viên cần:**

* Đưa mã nguồn lên GitHub
* Dán link của repository lên phần nộp bài trên CodeGymX

Hướng dẫn

* Tạo project Gradle:



* Thêm các depencies trong file buid.gradle:

compileOnly('javax.servlet:javax.servlet-api:4.0.1')  
 implementation group: 'org.springframework', name: 'spring-core', version: '5.3.2'  
 implementation group: 'org.springframework', name: 'spring-context', version: '5.3.2'  
 implementation group: 'org.springframework', name: 'spring-beans', version: '5.3.2'  
 implementation group: 'org.springframework', name: 'spring-web', version: '5.3.2'  
 implementation group: 'org.springframework', name: 'spring-webmvc', version: '5.3.2'  
 implementation group: 'org.thymeleaf', name: 'thymeleaf-spring5', version: '3.0.11.RELEASE'  
 implementation group: 'nz.net.ultraq.thymeleaf', name: 'thymeleaf-layout-dialect', version: '2.5.2'  
 implementation group: 'org.hibernate', name: 'hibernate-core', version: '5.3.0.Final'  
 implementation group: 'org.hibernate', name: 'hibernate-entitymanager', version: '5.3.0.Final'  
 implementation group: 'org.springframework', name: 'spring-orm', version: '5.3.2'  
 implementation group: 'mysql', name: 'mysql-connector-java', version: '8.0.22'  
 implementation group: 'org.springframework.data', name: 'spring-data-jpa', version: '2.4.2'  
 //Thư vện validate  
// https://mvnrepository.com/artifact/javax.validation/validation-api  
 implementation group: 'javax.validation', name: 'validation-api', version: '2.0.1.Final'  
 implementation group: 'org.hibernate', name: 'hibernate-validator', version: '6.0.10.Final'

plugins **{** id 'java'  
 id 'war'  
**}**

* Tạo package: **rikkei.academy.config :**
* Tạo: class: **AppConfig** trong package: **config**

package rikkei.academy.config;  
  
import org.springframework.beans.BeansException;  
import org.springframework.beans.factory.annotation.Qualifier;  
import org.springframework.context.ApplicationContext;  
import org.springframework.context.ApplicationContextAware;  
import org.springframework.context.MessageSource;  
import org.springframework.context.annotation.Bean;  
import org.springframework.context.annotation.ComponentScan;  
import org.springframework.context.annotation.Configuration;  
import org.springframework.context.support.ResourceBundleMessageSource;  
import org.springframework.data.jpa.repository.config.EnableJpaRepositories;  
import org.springframework.data.web.config.EnableSpringDataWebSupport;  
import org.springframework.format.FormatterRegistry;  
import org.springframework.jdbc.datasource.DriverManagerDataSource;  
import org.springframework.orm.jpa.JpaTransactionManager;  
import org.springframework.orm.jpa.JpaVendorAdapter;  
import org.springframework.orm.jpa.LocalContainerEntityManagerFactoryBean;  
import org.springframework.orm.jpa.vendor.HibernateJpaVendorAdapter;  
import org.springframework.transaction.PlatformTransactionManager;  
import org.springframework.transaction.annotation.EnableTransactionManagement;  
import org.springframework.web.servlet.config.annotation.EnableWebMvc;  
import org.springframework.web.servlet.config.annotation.WebMvcConfigurer;  
import org.thymeleaf.spring5.SpringTemplateEngine;  
import org.thymeleaf.spring5.templateresolver.SpringResourceTemplateResolver;  
import org.thymeleaf.spring5.view.ThymeleafViewResolver;  
import org.thymeleaf.templatemode.TemplateMode;  
  
import javax.persistence.EntityManager;  
import javax.persistence.EntityManagerFactory;  
import javax.sql.DataSource;  
import java.util.Properties;  
  
@Configuration  
@EnableWebMvc  
@EnableTransactionManagement  
@EnableSpringDataWebSupport  
@ComponentScan("rikkei.academy.controller")  
public class AppConfig implements WebMvcConfigurer, ApplicationContextAware {  
 private ApplicationContext applicationContext;  
  
 @Override  
 public void setApplicationContext(ApplicationContext applicationContext) throws BeansException {  
 this.applicationContext = applicationContext;  
 }  
 //Cấu hình Thymleaf  
 @Bean  
 public SpringResourceTemplateResolver templateResolver() {  
 SpringResourceTemplateResolver templateResolver = new SpringResourceTemplateResolver();  
 templateResolver.setApplicationContext(applicationContext);  
 templateResolver.setPrefix("/WEB-INF/views");  
 templateResolver.setSuffix(".html");  
 templateResolver.setTemplateMode(TemplateMode.*HTML*);  
 templateResolver.setCharacterEncoding("UTF-8");  
 return templateResolver;  
 }  
 @Bean  
 public SpringTemplateEngine templateEngine() {  
 SpringTemplateEngine templateEngine = new SpringTemplateEngine();  
 templateEngine.setTemplateResolver(templateResolver());  
 return templateEngine;  
 }  
 @Bean  
 public ThymeleafViewResolver viewResolver() {  
 ThymeleafViewResolver viewResolver = new ThymeleafViewResolver();  
 viewResolver.setTemplateEngine(templateEngine());  
 viewResolver.setCharacterEncoding("UTF-8");  
 viewResolver.setContentType("UTF-8");  
 return viewResolver;  
 }  
  
 //Cấu hình JPA  
 @Bean  
 @Qualifier(value = "entityManager")  
 public EntityManager entityManager(EntityManagerFactory entityManagerFactory) {  
 return entityManagerFactory.createEntityManager();  
 }  
  
 @Bean  
 public LocalContainerEntityManagerFactoryBean entityManagerFactory() {  
 LocalContainerEntityManagerFactoryBean em = new LocalContainerEntityManagerFactoryBean();  
 em.setDataSource(dataSource());  
 em.setPackagesToScan("rikkei.academy.model");  
  
 JpaVendorAdapter vendorAdapter = new HibernateJpaVendorAdapter();  
 em.setJpaVendorAdapter(vendorAdapter);  
 em.setJpaProperties(additionalProperties());  
 return em;  
 }  
  
 @Bean  
 public DataSource dataSource() {  
 DriverManagerDataSource dataSource = new DriverManagerDataSource();  
 dataSource.setDriverClassName("com.mysql.cj.jdbc.Driver");  
 dataSource.setUrl("jdbc:mysql://localhost:3306/bai8\_validate");  
 dataSource.setUsername("root");  
 dataSource.setPassword("Minhtri29092014");  
 return dataSource;  
 }  
  
 @Bean  
 public PlatformTransactionManager transactionManager(EntityManagerFactory emf) {  
 JpaTransactionManager transactionManager = new JpaTransactionManager();  
 transactionManager.setEntityManagerFactory(emf);  
 return transactionManager;  
 }  
  
 public Properties additionalProperties() {  
 Properties properties = new Properties();  
 properties.setProperty("hibernate.hbm2ddl.auto", "update");  
 properties.setProperty("hibernate.dialect", "org.hibernate.dialect.MySQL5Dialect");  
 return properties;  
 }  
}

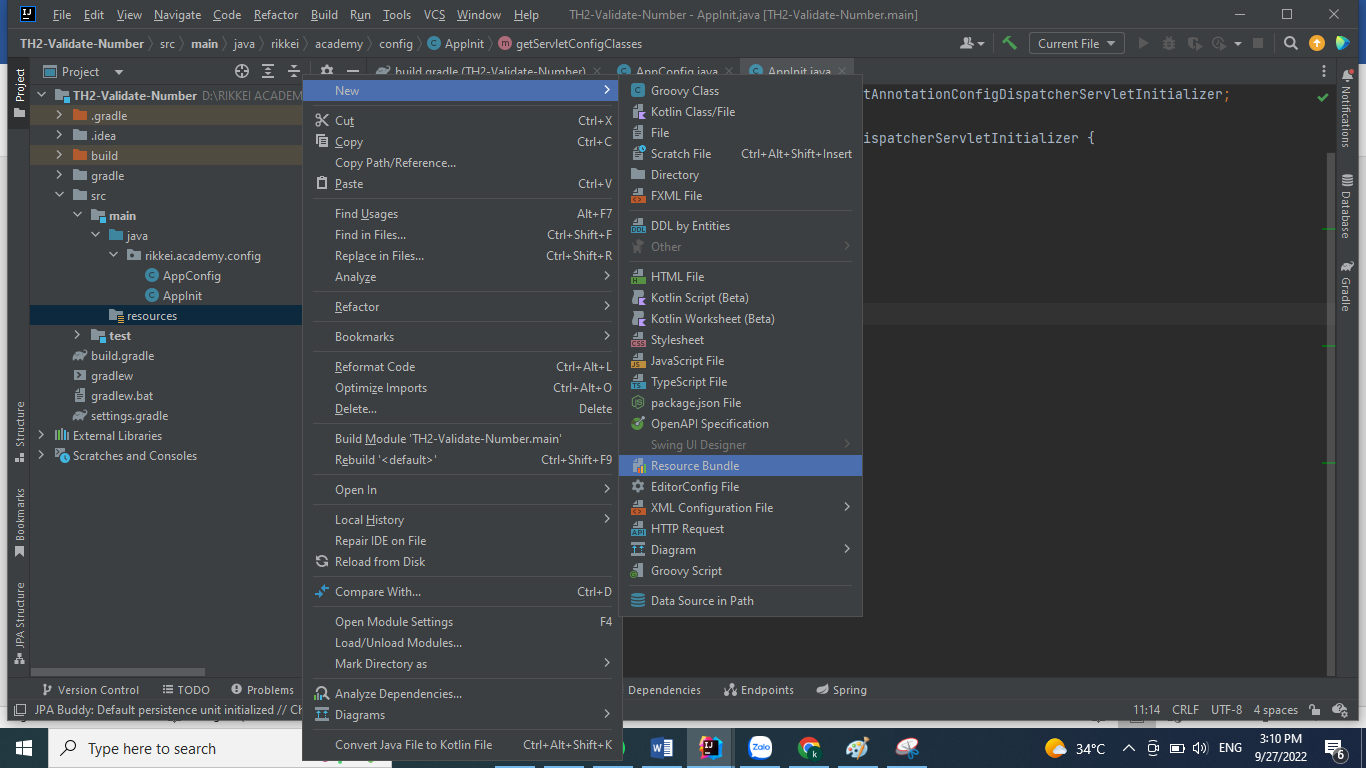
* Tạo class: **AppInit** trong package: **config:**

package rikkei.academy.config;  
  
import org.springframework.web.servlet.support.AbstractAnnotationConfigDispatcherServletInitializer;  
  
public class AppInit extends AbstractAnnotationConfigDispatcherServletInitializer {  
 @Override  
 protected Class<?>[] getRootConfigClasses() {  
 return new Class[]{AppConfig.class};  
 }  
  
 @Override  
 protected Class<?>[] getServletConfigClasses() {  
 return new Class[0];  
 }  
  
 @Override  
 protected String[] getServletMappings() {  
 return new String[]{"/"};  
 }  
}

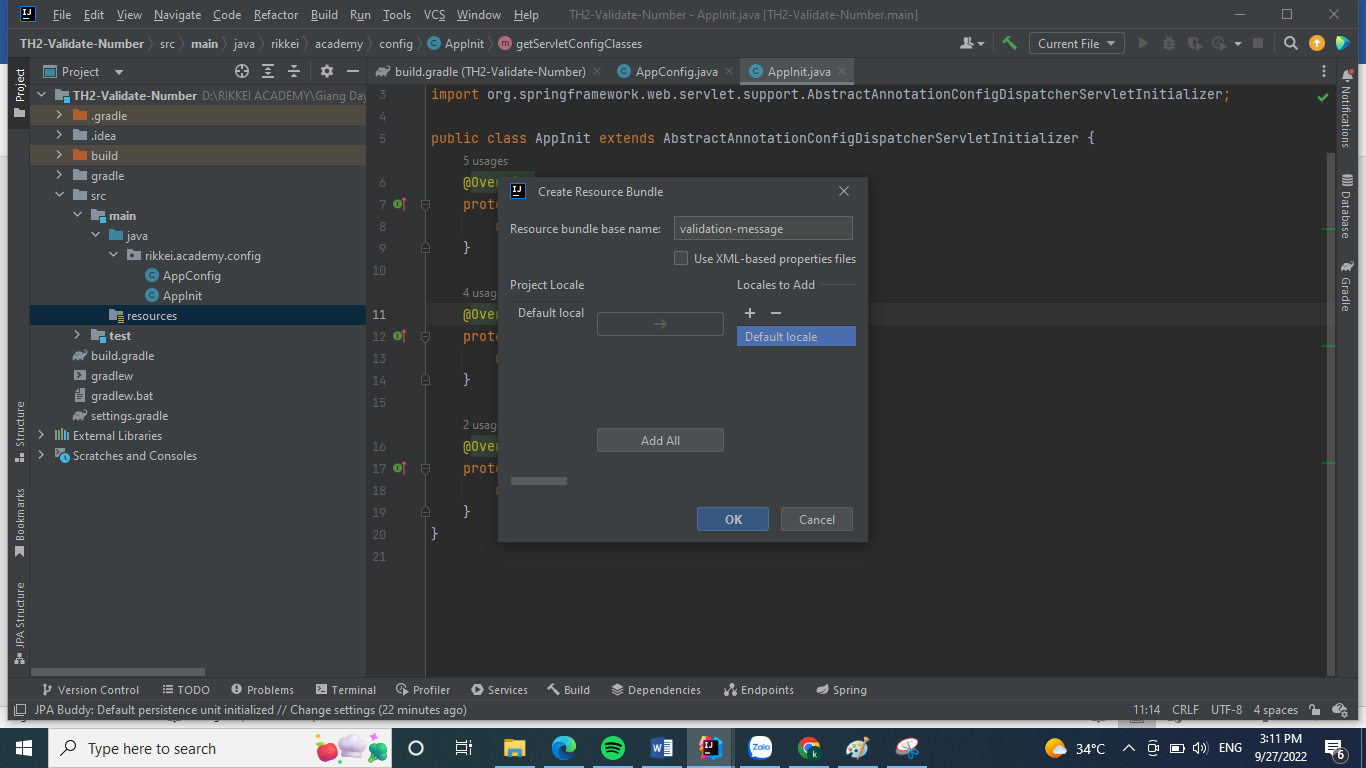
* Tạo package: **model** trong package: **rikkei.academy =>** Tạo class: **PhoneNumber** trong package này:

package rikkei.academy.model;  
  
import org.springframework.stereotype.Component;  
import org.springframework.validation.Errors;  
import org.springframework.validation.ValidationUtils;  
import org.springframework.validation.Validator;  
@Component  
public class PhoneNumber implements Validator {  
 private String number;  
  
 public String getNumber() {  
 return number;  
 }  
  
 public void setNumber(String number) {  
 this.number = number;  
 }  
  
 @Override  
 public boolean supports(Class<?> clazz) {  
 return PhoneNumber.class.isAssignableFrom(clazz);  
 }  
  
 @Override  
 public void validate(Object target, Errors errors) {  
  
 PhoneNumber phoneNumber = (PhoneNumber) target;  
 String number = phoneNumber.getNumber();  
 ValidationUtils.*rejectIfEmpty*(errors, "number", "number.empty");  
 if (number.length()>11 || number.length()<10){  
 errors.rejectValue("number", "number.length");  
 }  
 if (!number.startsWith("0")){  
 errors.rejectValue("number", "number.startsWith");  
 }  
 if (!number.matches("(^$|[0-9]\*$)")){  
 errors.rejectValue("number", "number.matches");  
 }  
 }  
}

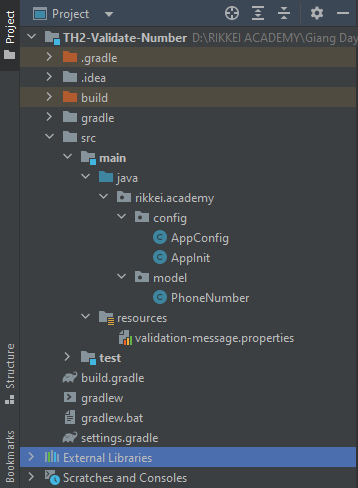
* Tạo file: **validation-message.properties** trong thư mục **resources**



* Đặt tên: **validation-message**



* Cấu trúc dự án lúc này như sau:



* **Thêm đoạn code sau vào file: validation-message.properties:**

number.empty = phoneNumber not empty.  
number.startsWith = phoneNumber start with 0.  
number.length = length form 10 to 11.  
number.matches = phoneNumber only include number.

* **Thêm config đường dẫn đến file này trong AppConfig**

@Bean  
public MessageSource messageSource() {  
 ResourceBundleMessageSource messageSource = new ResourceBundleMessageSource();  
 messageSource.setBasenames("validation-message");  
 return messageSource;  
}

* **Tạo package: controller => Tạo class: PhoneController:**

package rikkei.academy.controller;  
  
import org.springframework.stereotype.Controller;  
import org.springframework.ui.Model;  
import org.springframework.validation.BindingResult;  
import org.springframework.web.bind.annotation.GetMapping;  
import org.springframework.web.bind.annotation.ModelAttribute;  
import org.springframework.web.bind.annotation.PostMapping;  
import rikkei.academy.model.PhoneNumber;  
  
import javax.validation.Valid;  
  
@Controller  
public class PhoneController {  
 @GetMapping("/")  
 public String showForm(Model model){  
 model.addAttribute("phoneNumber", new PhoneNumber());  
 return "/index";  
 }  
 @PostMapping("/")  
 public String checkValidation (@Valid @ModelAttribute("phoneNumber")PhoneNumber phoneNumber, BindingResult bindingResult, Model model){  
 new PhoneNumber().validate(phoneNumber, bindingResult);  
 if (bindingResult.hasFieldErrors()){  
 return "/index";  
 }  
 else {  
 model.addAttribute("phoneNumber", phoneNumber);  
 return "/result";  
 }  
 }  
}

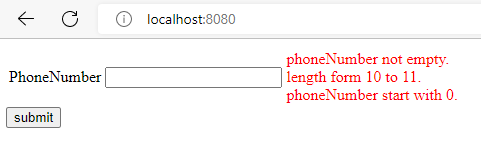
* **Tạo ra tầng views: webapp/WEB-INF/views:**
* **Tạo file: index.html trong views:**

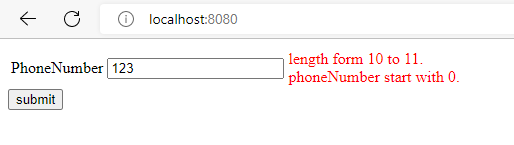
<!DOCTYPE html>  
<html xmlns:th="http://www.thymeleaf.org">  
<head>  
  
 <title>Input PhoneNumber</title>  
</head>  
<body>  
<form action="#" th:action="@{/}" th:object="${phoneNumber}"  
 method="POST">  
 <table>  
 <tr>  
 <td>PhoneNumber</td>  
 <td><input type="text" th:field="\*{number}"/></td>  
 <td ><span style="color: red" th:if="${#fields.hasErrors('number')}" th:errors="\*{number}"></span></td>  
 </tr>  
 </table>  
 <button type="submit">submit</button>  
</form>  
</body>  
</html>

* **Tạo file: result.html trong views**

<!DOCTYPE html>  
<html xmlns:th="http://www.thymeleaf.org">  
<head>  
 <title>PhoneNumber</title>  
</head>  
<body>  
<h1>Your phone <span style="color: red" th:text="${phoneNumber.number}"></span></h1>  
</body>  
</html>

* **Add Tomcat chạy chương trình và quan sát các trường hợp validate:**







**https://github.com/nguyendongminhtri/MD4-Bai8-TH2-Validate-Phone.git**

### **Hướng dẫn nộp bài:**

Up code lên github.

Paste link github vào phần nộp bài.