



Spring MVC Web framework

I+₁n+₂t+₃e+₄r+₅n+₆a+₇t+₈i+₉o+₁₀n+₁₁a+₁₂l+₁₃i+₁₄z+₁₅a+₁₆t+₁₇i+₁₈o+₁₉n



1

1



1. Resource Bundles	p 3
2. Validation & i18n	p 8
2.1 Jakarta Bean Validation	p 8
2.1.1 Override the default	
validation annotation messages	p 9
2.1.2 Custom Error Messages	p 11
2.2 Validator class	p 12
2.3 Own custom validation annotation	p 14
2.4 NumberFormatException	p 15
2.5 View & i18n	p 17
3. Assigning values in a Controller with @Value	p 18
4.1 Fragments: Header and Footer	p 24
4.2 Custom Fragment	p 26
5.1 Switch language: LocaleChangeInterceptor	p 27
5.2 Switch Language: LocaleController	p 31

2



1. Resource Bundles

Format and Internationalization
localhost:8080

Welcome to Court Reservation System

Locale : nl

Formatierung und Internationalisierung
localhost:8080

Willkommen zum Spielplatz-Reservierungssystem

Locale : de

Talen

Taal
Duits

Talen rangschikken op basis van je voorkeur

Duits
Deze taal wordt gebruikt om de Google Chrome-UI weer te geven

Engels

Formatierung un...
Formatierung und Internationalisierung
localhost:8080

Willkommen zum Spielplatz-Reservierungssystem

Locale : de


OR

SpringBootApplication

```
@Bean
LocaleResolver localeResolver() {
    SessionLocaleResolver slr = new SessionLocaleResolver();
    slr.setDefaultLocale(Locale.GERMAN);
    return slr;
}
```

3

3



Resource Bundles

application.properties ×

1 spring.messages.basename=i18n/messages

src/main/resources

└ i18n

 └ messages_de.properties

 └ messages.properties

 └ static

 └ templates

 └ application.properties

welcome.message=Willkommen zum Spielplatz-Reservierungssystem
titleWelcome=Formatierung und Internationalisierung

welcome.message=Welcome to Court Reservation System
titleWelcome=Format and Internationalization

4

4

Spring

Resource Bundles

welcome.html:

```

<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<head>
<meta charset="ISO-8859-1">
<title th:text="#{titleWelcome}">Internationalization</title>
</head>
<body>
    <h2 th:text="#{welcome.message}"></h2>

    <p th:text="|Locale : ${#locale}|"></p>
</body>

```

titleWelcome=

welcome.message=

5

5

Spring

Resource Bundles (Format date)

src/main/resources
 i18n

date.format.pattern=dd-MM-yyyy

messages_de.properties
 messages.properties

date.format.pattern=yyyy-MM-dd

← → ↻ ⓘ
localhost:8080/welcome

Welcome to Court Reservation System

2025-03-01

2025-03-01

Locale : en_GB

← → ↻ ⓘ
localhost:8080/welcome

Willkommen zum Spielplatz-Reservierungssystem

01-03-2025

01-03-2025

Locale : de

6

6

Resource Bundles (Format date)

Spring

html:

To bind the local variable **df** to the value of the `date.format.pattern` message key

```
<p th:with="df=#{date.format.pattern}"  
  th:text="${#dates.format(today, df)}"></p>
```



`java.util.Date`

```
<p th:with="df=#{date.format.pattern}"  
  th:text="${#temporals.format(todayLocalDate, df)}"></p>
```



`java.time.LocalDate`

7

7

2. Validation & i18n

Spring

2.1 Jakarta Bean Validation

```
@NotBlank  
@Size(min=3, max=60)  
private String firstName;
```

```
<p><label th:text="|#{label_first_name}*"></label>  
  <input type="text" th:field="*{firstName}" size="20"/>  
  <span th:if="${#fields.hasErrors('firstName')}"  
    th:errorclass="error" th:errors="*{firstName}"></span>  
</p>
```

FirstName* size must be between 3 and 60
must not be blank

8

8

2.1.1 Override the default validation annotation messages



label_first_name=FirstName
NotBlank=Firstname required
Size=Wrong size

converter.properties

converter_fr.properties

label_first_name=Prénom
NotBlank=Vous devez spécifier le prénom
Size=Format incorrect

@NotBlank

@Size(min=3, max=60)

private String firstName;

FirstName*

Wrong size

Firstname required

Prénom*

Vous devez spécifier le prénom

Format incorrect

9

2.1.1 Override the default validation annotation messages



application.properties ×
1 spring.messages.basename=i18n/converter

src/main/resources

i18n

converter.properties

converter_fr.properties

templates

application.properties

10

2.1.2 Custom Error Messages



converter.properties

converter_fr.properties

validation.firstname.NotBlank.message=First name is required
validation.Size.message=Size must be between {min} and {max}

validation.firstname.NotBlank.message=Vous devez spécifier le prénom
validation.Size.message=la taille doit être comprise entre {min} et {max}

domain
Contact.java

```
@NotBlank(message="{validation.firstname.NotBlank.message}")  
@Size(min=3, max=60, message="{validation.Size.message}")  
private String firstname;
```

FirstName*
Size must be between 3 and 60
First name is required

Prénom*
la taille doit être comprise entre 3 et 60
Vous devez spécifier le prénom

11

11

2.2 Validator class



src/main/resources
i18n
converter.properties

matchingPassword.registration.password =
my validatormessage

12

12

2.2 Validator class

```
...
public class RegistrationValidator implements Validator {
...
    @Override
    public void validate(Object target, Errors errors) {
        Registration registration = (Registration) target;

        ...

        if (!(registration.getPassword().equals(
            registration.getConfirmPassword())) {
            errors.rejectValue("password",
                "matchingPassword.registration.password",
                "Password and Confirm Password Not match.");
        }
    }
}

void rejectValue(@Nullable
    String field,
    String errorCode,
    String defaultMessage)
```

Password: my validatormessage

13

2.3 Own custom validation annotation



src/main/resources
i18n
converter.properties

validator.validUserName=my validUserName message

validator
> UserNameConstraintValidator.java
> ValidUserName.java

```
...
public @interface ValidUserName {

    String message() default "{validator.validUserName";

    ...
}
```

```
public class Registration { ...
    @ValidUserName
    private String userName;
}
```

User Name: 123 my validUserName message

14

14

2.4 NumberFormatException



Price increase

Increase (%):

Execute

Failed to convert property value of type java.lang.String to required type java.lang.Integer for property percentage; nested exception is java.lang.NumberFormatException: For input string: "abc"

Overwrite default message →

Price increase

Increase (%):

Execute

my message !!!

15

15

2.4 NumberFormatException



```
package domain;  
public class PriceIncrease {  
    private Integer percentage; // or int
```

@Controller

```
...  
    model.addAttribute("priceIncrease", new PriceIncrease ());
```

```
<form th:action="@{/increase}" th:object="${priceIncrease}"  
    method="post">
```

```
...  
    <input type="text" th:field="*{percentage}"/> ...
```

src/main/resources
i18n
converter.properties

typeMismatch.**priceIncrease**.**percentage**=my message !!!

objectClassName

property

16

16

2.5 View & i18n



src/main/resources
i18n
converter.properties

contact_save_fail=Failed saving contact

registrationForm.html

```
<div th:if="{ #fields.hasErrors()}" class="error"  
th:text="{ #contact_save_fail}"></div>
```

Failed saving contact

17

17

3. Assigning values in a Controller with @Value



Problem:

When creating a controller, you don't want to
hard-code a field value.

Instead, you want to assign a value present in an
bean or properties file (i.e. message.properties)

Solution: **@Value**

18

18

3. Assigning values in a Controller with @Value

...
@Controller

@RequestMapping("/about")

public class AboutController {

@Value("#{ messageSource.getMessage('admin.email',null,'en')}")

private String **email**;

@GetMapping

public String courtReservation(Model model) {

model.addAttribute("email", **email**);

return "about";

}

}

19

19

@Value("#{ messageSource.getMessage('admin.email',null,'en')}")

- The value assigned to the @Value = SpEL statement
(Spring Expression Language)
- messageSource →
org.springframework.context.support.ResourceBundleMessageSource

• getMessage

```
String getMessage(String code,
                  Object[] args,
                  Locale locale)
    throws NoSuchMessageException
```

Try to resolve the message. Treat as an error if the message can't be found.

Parameters:

- code - the code to lookup up, such as 'calculator.noRateSet'args
- Array of arguments that will be filled in for params within the message (params look like "{0}", "{1,date}", "{2,time}" within a message), or null if none.locale
- the Locale in which to do the lookup

Returns:the resolved message

20

20

3. Assigning values in a Controller with @Value

src/main/resources

i18n

converter.properties

admin.email=reservation@court.com

```
converter.properties x
1welcome.message=Welcome to Court Reservation System
2titleWelcome=Format and Internationalization
3version=Version
4email=Email
5admin.email=reservation@court.com
```

```
converter_de.properties x
1welcome.message=Willkommen zum Spielplatz-Reservierungssystem
2titleWelcome=Formatierung und Internationalisierung
3version=Version
4email=E-Mail
```

no **admin.email** in other resourcebundles

21

21

3. Assigning values in a Controller with @Value

Spring_Boot_i18n_Format_Value [boot] [devtools]

- src/main/java
 - com.springboot.i18n.FormatValue
 - AboutController.java
 - SpringBootI18nFormatValueApplication.java
- src/main/resources
 - i18n
 - converter_de.properties
 - converter.properties
 - static
 - templates
 - about.html
 - application.properties

```
...
<td th:text="|#{email}:"></td>
<td th:text="$#{email}"></td>
...
```

localhost:8080/about

Welcome to Court Reservation System

Version: 1.0

Email: reservation@court.com

localhost:8080/about


Willkommen zum Spielplatz-Reservierungssystem

Version: 1.0

E-Mail: reservation@court.com

22

22



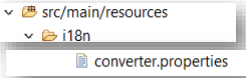
```

...
@Controller
@RequestMapping("/about")
public class AboutController {

    @Value(
        "#{messageSource.getMessage('admin.email', new Object[]{ 'ADMIN', 123 }, 'en')}")
    private String email;

    ...
}

```




admin.email=reservation@court.com {0} and {1}

reservation@court.com ADMIN and 123

23

23



4.1 Fragments: Header and Footer

Fragments in Thymeleaf are reusable parts of HTML templates that can be included or replaced in other templates. They allow you to modularize your templates, making them easier to maintain and reuse across multiple pages.

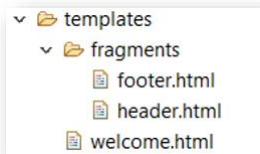
```

...
<body>
    <!-- Include the header fragment -->
    <div th:replace="~{fragments/header :: header}"></div>

    <main>
        <h1 th:text="#{title}"></h1>
        <p th:text="#{text}"></p>
    </main>

    <!-- Include the footer fragment -->
    <div th:replace="~{fragments/footer :: footer}"></div>
</body>
</html>

```



replace the <div> tag with the content of the header fragment defined in the fragments/header.html file.

replace the <div> tag with the content of the footer fragment defined in the fragments/footer.html file.

24

24

header.html

```

<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<header>
<fieldset>
<legend>HEADER</legend>
...
</fieldset>
</header>
</html>

```

footer.html

```

<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<footer>
<fieldset>
<legend>FOOTER</legend>
  <p th:text="#{footerText}"></p>
</fieldset>
</footer>
</html>

```

Spring_Boot_i18n_ChangeLanguage

- templates
 - fragments
 - footer.html
 - header.html
 - welcome.html

welcome.html

```

<!-- Include the header fragment -->
<div th:replace="~{fragments/header :: header}"></div>

```

25

4.2 Custom fragment

Spring

You can create your own **custom fragments**, which are reusable pieces of HTML that you define within your templates.

welcome.html

```

...
<body>
<!-- Include the localeSwitcher fragment (custom fragment) -->
<div th:replace="~{fragments/localeSwitcher :: locale-switch}"></div>
...
</html>

```

localeSwitcher.html

```

<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<div th:fragment="locale-switch">
  <fieldset>
  <legend>A CUSTOM FRAGMENT</legend>
  ...
  </fieldset>
</div>
</html>

```

Spring_Boot_i18n_ChangeLanguage2

- templates
 - fragments
 - footer.html
 - localeSwitcher.html
 - welcome.html

26

5.1 Switch language: LocaleChangeInterceptor

Spring_Boot_i18n_ChangeLanguage [boot] [devtools]

- src/main/java
 - com.springboot_changelanguage
 - SpringBootListFirstExample1Application.java**
 - WelcomeController.java
- src/main/resources
 - i18n**
 - converter_en.properties
 - converter_nl.properties
 - converter.properties
 - static
 - templates
 - fragments
 - footer.html
 - header.html
 - welcome.html
 - application.properties

Demo English

In English

FOOTER

This is the footer

Demo Nederlands

In het Nederlands

FOOTER

Dit is de footer

27

...
<header>
<fieldset>
<legend>HEADER</legend>
<nav>

 <a th:href="@{/?lang=nl}">Nederlands
 <a th:href="@{/?lang=en}">English

</nav>
</fieldset>
</header>
</html>

header.html

Spring

...<body>
<div th:replace="~{fragments/header :: header}"></div> ...


welcome.html

converter_nl.properties	converter_en.properties
1 title=Demo Nederlands	1 title=Demo English
2 text=In het Nederlands	2 text=In English
3 footerText=Dit is de footer	3 footerText=This is the footer

converter.properties

1# Default fallback in case a specific language file is not available

28

... header.html 

```
<ul>
  <li><a th:href="@{/?lang=nl}">Nederlands</a></li>
  <li><a th:href="@{/?lang=en}">English</a></li>
</ul>
```

@SpringBootApplication


```
public class SpringBootListFirstExample1Application
    implements WebMvcConfigurer {

    ...
    @Bean
    LocaleResolver localeResolver() {
        SessionLocaleResolver slr = new SessionLocaleResolver();
        slr.setDefaultLocale(Locale.ENGLISH);
        return slr;
    }

    @Override
    public void addInterceptors(InterceptorRegistry registry) {
        LocaleChangeInterceptor localeChangeInterceptor =
            new LocaleChangeInterceptor();
        localeChangeInterceptor.setParamName("lang");
        registry.addInterceptor(localeChangeInterceptor);
    }
}
```

29

29



The HTML code provides language switch links (Nederlands for Dutch and English for English), passing the selected language as a parameter (lang=nl or lang=en).

```
<li><a th:href="@{/?lang=nl}">Nederlands</a></li>
<li><a th:href="@{/?lang=en}">English</a></li>
```

In the Spring Boot application, the **SessionLocaleResolver** sets the default locale to English.

The **LocaleChangeInterceptor** intercepts requests to change the **locale** based on the "**lang**" parameter in the URL, allowing users to switch between languages dynamically.

```
@Override
public void addInterceptors(InterceptorRegistry registry) {
    LocaleChangeInterceptor localeChangeInterceptor = new LocaleChangeInterceptor();
    localeChangeInterceptor.setParamName("lang");
    registry.addInterceptor(localeChangeInterceptor);
}
```

30

30

5.2 Switch language: LocaleController

```

Spring_Boot_i18n_ChangeLanguage2 [boot] [devtools]
├── src/main/java
│   └── com.springboot_changeLanguage2
│       ├── LocaleController.java
│       ├── SpringBootI18nChangeLanguage2Application.java
│       └── WelcomeController.java
├── src/main/resources
│   ├── i18n
│   │   ├── converter_en.properties
│   │   ├── converter_nl.properties
│   │   └── converter.properties
│   ├── static
│   ├── templates
│   │   ├── fragments
│   │   │   ├── footer.html
│   │   │   ├── localeSwitcher.html
│   │   │   └── welcome.html
│   │   └── application.properties

```

31

31

```

<!DOCTYPE html>
<html xmlns:th="http://www.thymeleaf.org">
<div th:fragment="locale-switch" >
<fieldset>
<legend>A CUSTOM FRAGMENT</legend>

    <form th:action="@{/changeLocale}" method="get">

        <select name="lang" onchange="this.form.submit()">
            <option value="nl"
                th:selected="${#locale.language == 'nl'}">Nederlands</option>

            <option value="en"
                th:selected="${#locale.language == 'en'}">English</option>
        </select>
    </form>
</fieldset>
</div>
</html>

```

32

32

localeSwitcher.html

```
<option value="nl"
    th:selected="${#locale.language == 'nl'}">Nederlands</option>
```

```
<option value="en"
    th:selected="${#locale.language == 'en'}">English</option>
```

The expression `th:selected="${#locale.language == 'nl'}"` in Thymeleaf is used to conditionally mark an `<option>` element within a `<select>` dropdown menu as selected if the current language of the application is Dutch ('nl').

Similarly, `th:selected="${#locale.language == 'en'}"` does the same for English ('en').

This ensures that the correct language option is pre-selected based on the user's language preference when the page is rendered.



33

33

@Controller

```
public class LocaleController {
```

@SpringBootApplication

```
@Bean
LocaleResolver localeResolver() {
    SessionLocaleResolver slr = new SessionLocaleResolver();
    slr.setDefaultLocale(Locale.ENGLISH);
    return slr;
}
```

@Autowired

```
private LocaleResolver localeResolver;
```

localeSwitcher.html

```
<form th:action="@{/changeLocale}" method="get">
```

@GetMapping("/changeLocale")

```
public String changeLocale(HttpServletRequest request,
    HttpServletResponse response,
    @RequestParam("lang") String lang) {
```

HttpServletRequest request: Provides access to the HTTP request information.

HttpServletResponse response: Allows modifying the HTTP response.

@RequestParam("lang") String lang: Retrieves the value of the "lang" query parameter from the request URL.

```
<select name="lang"
```

localeSwitcher.html

34

34

```

@GetMapping("/changeLocale")
public String changeLocale(HttpServletRequest request,
    HttpServletResponse response,
    @RequestParam("lang") String lang) {
    Locale locale = switch (lang) {
        case "nl" -> Locale.forLanguageTag("nl-NL");
        //case "en" -> Locale.ENGLISH;
        default -> Locale.ENGLISH;
    };

    //Change the localeResolver bean to the selected locale
    localeResolver.setLocale(request, response, locale);

    // Redirect back to the previous page (referer = a reserved
    // header name in HTTP)
    return "redirect:" + request.getHeader("Referer");
}
}

```

```

<select name="lang" onch
    <option value="nl"
    <option value="en"

```

35