FrontBackEnd

Using th:each in Thymeleaf

January 10, 2020 5 Comments Thymeleaf Attribute Iteration th:each

1. Overview

Thymeleaf is a Java-based template engine used for processing **HTML**, **XML**, **JS**, and many other documents. In this tutorial, we will show how to use Thymeleaf attribute `th:each` dedicated for iteration over collections.

If you need some more information about how to start working with Thymeleaf in Spring Boot, just take a look at our introduction article here.

2. Iteration attribute `th:each`

Thymeleaf comes with special attribute `th:each` , used to iterate over collections of different object types.

There are several objects that Thymeleaf considered as iterable:

- objects implementing `java.util.Iterable` interface,
- objects implementing `java.util.Enumeration` interface,
- objects implementing `java.util.Iterator` iterface,
- objects implementing `java.util.Map` interface,
- and also `arrays` .

Let's assume that we want to display a list of customers in a simple HTML table using Thymeleaf engine.

In our example application **Customer** object will have the following structure:

Copy

```
package com.frontbackend.thymeleaf.customers.model;

import java.util.List;

import lombok.Data;

@Data
public class Customer {

    private String firstName;
    private String lastName;
    private String email;
    private int age;
    private List<Address> addressList;

}
```

Address list in the **Customer** object is added to show how to iterate over nested lists in Thymeleaf templates.

In this case **Address** will contain the following fields:

Copy

```
package com.frontbackend.thymeleaf.customers.model;

import lombok.Data;

@Data
public class Address {

    private String street;
    private String city;
    private String zip;
    private String state;

}
```

We will use `th:each` to iterate through the list of customers and list of addresses for each customer.

`CustomerController` class was defined to handle all **GET** requests to `/customers` URI and return a rendered page `customers.html` as an output (which is our Thymeleaf template located in `/resources/templates`).

Copy

```
package com.frontbackend.thymeleaf.customers.controller;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;

import com.frontbackend.thymeleaf.customers.service.CustomerService;

@Controller
public class CustomerController {

    private final CustomerService customerService;

    @Autowired
    public CustomerController(CustomerService customerService) {
        this.customerService = customerService;
    }

    @GetMapping("customers")
    public String main(Model model) {
        model.addAttribute("customers", customerService.mockCustomers());

        return "customers";
    }
}
```

`CustomerService` returns sample customer list prepared in a JSON (mockedCustomers.json).

Copy

```java
package com.frontbackend.thymeleaf.customers.service;

import com.fasterxml.jackson.core.type.TypeReference;
import com.fasterxml.jackson.databind.ObjectMapper;
import com.frontbackend.thymeleaf.customers.model.Customer;
import lombok.extern.slf4j.Slf4j;
import org.springframework.stereotype.Service;

import java.io.IOException;
import java.util.List;

@Slf4j
@Service
public class CustomerService {

    public List<Customer> mockCustomers() {
        ObjectMapper objectMapper = new ObjectMapper();

        try {
            return objectMapper.readValue(getClass().getClassLoader()

.getResourceAsStream("mockedCustomers.json"),
                    new TypeReference<List<Customer>>() {
                    });
        } catch (IOException e) {
            log.error(e.getMessage(), e);
        }

        return null;
    }
}
```

Thymeleaf template, that we used in this example, will list all customer information like: **firstName**, **lastName**, **age**, **email** with the number of row in table. Additionally we present all address data (**street**, **city**, **zip**, **state**) for each customer. This template have the following structure:

Copy

```
<!DOCTYPE HTML>
<html lang="en" xmlns:th="http://www.thymeleaf.org">
<head>
    <meta charset="UTF-8"/>
    <title>Spring Boot Thymeleaf Application</title>
</head>
<body>
<h1>List of Customers</h1>
<table>
    <tr>
        <th>No</th>
        <th>First Name</th>
        <th>Last Name</th>
        <th>Age</th>
        <th>Email</th>
        <th>Address list</th>
    </tr>
    <tr th:each="customer, custStat : ${customers}">
        <td th:text="${custStat.count}">1</td>
        <td th:text="${customer.firstName}">John</td>
        <td th:text="${customer.lastName}">Doe</td>
        <td th:text="${customer.age}">18</td>
        <td th:text="${customer.email}">john.doe@frontbackend.com</td>
        <td>
            <p th:each="address : ${customer.addressList}">
                <span th:text="${address.street}">Street</span>
                <span th:text="${address.city}">City</span>
                <span th:text="${address.zip}">ZIP</span>
                <span th:text="${address.state}">State</span>
            </p>
        </td>
    </tr>
</table>
</body>
</html>
```

Thymeleaf engine will repeat fragment of the template that contains `th:each` attribute (in this case the tr.../tr element) for each element in the iterable collection.

3. Iteration status

Thymeleaf provides a special `status` object that gives some useful information about the iteration process.

The status object contains the following information:

- `iteration index` , starting from 0 - the `${custStat.index}` property,

- `iteration index` , starting from 1 - the `${custStat.count}` property.
- `total amount` of elements in the iterated variable - the `${custStat.size}` property.
- `iter variable` for each iteration - the `${custStat.current}` property.
- Whether the current iteration is even or odd. These are the `${custStat.even}` and `${custStat.odd}` boolean properties.
- Whether the current iteration is the first one. This is the `${custStat.first}` boolean property.
- Whether the current iteration is the last one. This is the `${custStat.last}` boolean property.

4. Conclusion

In this article, we presented how iterations in Thymeleaf work. We explained `th:each` attribute using a simple real-life example.

Working application with code shown in this article is available in our GitHub Repository.

Comments (5)

R

Roshan

3 weeks ago | #

sir ,can i get its controller and service layer

Reply

franek

3 weeks ago | #

yes, just added CustomerController and CustomerService, you can always check the full example used in this article, It is available under GitHub Repository (https://github.com/martinwojtus/tutorials/tree/master/thymeleaf/thymeleaf-iteration).

Reply

R

Roshan

3 weeks ago | #

Thank you sir

Reply