БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ и ИНФОРМАТИКИ
Кафедра многопроцессорных сетей и систем

Рафеенко Е.Д.

Web- программирование

Thymeleaf

Содержание

Обзор

Thymeleaf — это современный серверный механизм шаблонов Java как для веб-, так и для автономных сред, способный обрабатывать HTML, XML, JavaScript, CSS и даже простой текст.

Thymeleaf опирается на концепцию естественных шаблонов, позволяющую внедрять свою логику в файлы шаблонов таким образом, чтобы это не мешало использованию шаблона в качестве прототипа дизайна.

Обзор

Режим шаблона HTML позволяет использовать любой тип HTML, включая HTML5, HTML 4 и XHTML. Код/структура шаблона будут максимально соблюдаться при выводе.

Режим шаблона XML позволяет использовать XML. В этом случае необходимо, чтобы код был правильно сформирован – нет незакрытых тегов, атрибутов без кавычек и т. д. – и синтаксический анализатор выдаст исключения, если будут обнаружены нарушения правильности формата.

Режим шаблона JAVASCRIPT позволит обрабатывать файлы JavaScript в приложении Thymeleaf.

ThymeLeaf API

В Thymeleaf 3.1 представлены интерфейсы, которые абстрагируют конкретные детали используемого веб-API.

`org.thymeleaf.web.IWebApplication` - Представляет веб-приложение и связанные с ним атрибуты.

`org.thymeleaf.web.IWebExchange` – Дает возможность обработки веб-запроса. Содержит запрос, сессию (если есть) и любые атрибуты, связанные с этим конкретным запросом.

`org.thymeleaf.web.IWebRequest` - Представляет веб-запрос: URL-путь, параметры, заголовки и файлы cookie.

`org.thymeleaf.web.IWebSession` - Представляет сессию, если она существует, содержащий все связанные атрибуты.

ThymeLeaf API

1. Создание объекта WebApplication

```
private final JakartaServletWebApplication application;
```

```
this.application =
JakartaServletWebApplication.buildApplication(
        getServletContext());
```

ThymeLeaf API

2. Создание объекта TemplateResolver

```
final WebApplicationTemplateResolver templateResolver =
new WebApplicationTemplateResolver(application);

// HTML – режим по умолчанию
templateResolver.setTemplateMode(TemplateMode.HTML);

// конвертируем "home" to "/WEB-INF/templates/home.html"
templateResolver.setPrefix("/WEB-INF/templates/");
templateResolver.setSuffix(".html");

// Устанавливаем TTL в 1 час.
templateResolver.setCacheTTLMs(Long.valueOf(3600000L));
```

ThymeLeaf API

Объект TemplateResolver отвечает за то как будет осуществляться доступ к шаблонам.

В данном случае будем получать файлы шаблонов в качестве ресурсов из контекста сервлета (ServletContext).

ThymeLeaf API

Наиболее важный объект в приложении Thymeleaf - TemplateEngine (реализация интерфейса ITemplateEngine). Его можно проинициализировать следующим образом:

```
...
    private final TemplateEngine templateEngine;
...

final TemplateEngine templateEngine = new
  TemplateEngine();
templateEngine.setTemplateResolver(templateResolver);
```

ThymeLeaf API

Объекты webExchange и webRequest:

```
final IServletWebExchange webExchange =
   this.application.buildExchange(request, response);

final IWebRequest webRequest = webExchange.getRequest();
final String path = request.getPathWithinApplication();
```

Contexts

Объекты WebContext:

```
WebContext ctx = new WebContext(webExchange,
  webExchange.getLocale());
```

Контекст должен содержать все данные, необходимые для работы механизма шаблонов (данные находятся в map), а также ссылаться на локаль, которая должна использоваться для интернационализации сообщений.

Стандартные выражения Thymeleaf

Выражения с переменными: ${...}

Выражения с переменными выбора: *{...}

Выражения с сообщениями: #{...}

Выражения с URL-адресами: @{...}

Выражения с фрагментами страниц: ~{...}

Стандартные выражения Thymeleaf

Литералы (текстовые, булевы, числовые, null, токены)

Текстовые выражения (конкатенация)

Арифметические, логические операции

Условные операторы

Стандартные выражения Thymeleaf

${x} вернет переменную x, хранящуюся в контексте Thymeleaf или в качестве атрибута запроса.

Данное выражение эквивалентно `ctx.getVariable("x");`

${param.x} вернет параметр запроса с именем x (параметр может быть многозначным).

${session.x} вернет атрибут сессии с именем x.

${application.x} вернет атрибут контекста сервлета с именем x.

Базовые объекты в выражениях

#ctx: объект контекста.

#vars: переменные контекста.

#locale: локаль контекста.

```
Established locale country:
 <span th:text="${#locale.country}">BLR</span>
```

Специальные объекты в выражениях

#dates: методы для объектов java.util.Date: форматирование, извлечение компонентов и т. д.

#calendars: аналогично #dates, но для объектов java.util.Calendar.

#numbers: методы форматирования числовых объектов.

#strings: методы для объектов String: contains, startsWith, prepending/appending.

#arrays, #lists, #sets, #maps: методы для arrays, lists, sets, maps

#aggregates: методы для агрегирования в массивах или коллекциях.

URL-адреса

URL-адреса являются важными элементами в шаблонах веб-приложений, и стандартный диалект Thymeleaf имеет для них специальный синтаксис : @{...}.

Существуют различные типы URL-адресов:

Абсолютный URL: http://www.thymeleaf.org

Относительный URL может быть:

Странично-зависимый (Page-relative): user/login.html

Контекстно-зависимый (Context-relative) - имя контекста на сервере будет добавлено автоматически: /itemdetails?id=3

Серверно-зависимый (Server-relative)- (позволяет обращаться к URL-адресам в другом контексте (приложении) на том же сервере: ~/billing/processInvoice

URL-адреса

Например:

```
<a href="details.html"
th:href="@{http://localhost:8080/gtvg/order/details
(orderId=${o.id})}">view</a>
```

`th:href` – это атрибут-модификатор: после обработки он вычислит URL-адрес используемой ссылки и установит это значение в атрибут href тега <a>.

Информационные ресурсы
Организационные вопросы