



Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ

КАФЕДРА КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)

НАПРАВЛЕНИЕ ПОДГОТОВКИ 09.03.01 Информатика и вычислительная техника

О Т Ч Е Т

по лабораторной работе № 10

Название: Формирование и отображение XML в HTML средствами сервера и клиента

Дисциплина: Языки Интернет-программирования

Студент

ИУ6-33Б
(Группа)

(Подпись, дата)

И.А. Нуруллаев
(И.О. Фамилия)

Преподаватель

(Подпись, дата)

(И.О. Фамилия)

Москва, 2022

Задание

Модифицировать код ЛР 8 таким образом, чтобы по запросу с указанными параметрами выдавался результат в формате XML (средствами стандартной сериализации ActiveSupport).

- Проверить формирование XML и сохранить в файл для отладки XSLT и второго приложения.
- Написать функциональный тест, проверяющий формат выдаваемых данных при запросе RSS.

Разработать XSLT-программу преобразования полученной XML в HTML.

Добавить в проверяемый XML-файл строку привязки к преобразованию `<?xml-stylesheet type="text/xsl" href="some_transformer.xslt"?>`. Проверить корректность отображения браузером результата преобразования.

Проверить на автономной Ruby-программе корректность преобразования, используя следующий фрагмент кода:

```
require 'nokogiri'
doc = Nokogiri::XML(File.read('some_file.xml'))
xslt = Nokogiri::XSLT(File.read('some_transformer.xslt'))
puts xslt.transform(doc)
```

Разработать второе приложение, являющееся посредником между клиентом и первым приложением, задачей которого является преобразование XML в HTML или передача в неизменном виде браузеру для отображения браузером. Приложение должно запускаться с указанием номера порта TCP, отличным от номера порта первого приложения (например rails server -p 3001)!

- Подготовить каркас приложения, а также форму формирования запроса, форму отображения результата и соответствующие действия контролера.
- Добавить в контроллер преобразование XML в HTML с помощью ранее разработанного XSLT-файла.
- Подключить запрос XML с первого приложения и проверить работу приложений в связке.
- Написать функциональный тест, проверяющий что при различных входных данных результат генерируемой страницы различен.
- Доработать код контроллера и представлений данного приложения для выдачи браузеру XML-потока в неизменном виде (организовать возможность выбора формата выдачи для пользователя).
- Проверить, что браузер получает XML первого приложения в неизменном виде.
- Доработать код контроллера приложения таким образом, чтобы XML-поток первого приложения получал дополнительную строку, указывающую xsl. Модифицировать форму запроса параметров таким образом, чтобы браузер получал в ответ XML. При этом разместить XSLT-файл в директории public.
- Проверить, что браузер производит преобразование XML->HTML в соответствии с xlt.
- Реализовать функциональные тесты второго приложения. Проверить результаты, формируемые приложением, на соответствие выбранному формату выдачи.

Итоговая форма ввода параметра должна содержать кнопки или селектор, позволяющие проверить два варианта преобразования:

- Серверное xml+xslt->html
- Клиентское xml+xslt->html

Решение:

Часть первая - API

```
WPL_bmstu - api_controller.rb

1  # Simon's hypothesis
2  class ApiController < ActionController::Base
3    before_action :parse_params, only: :result
4
5    def initialize
6      super
7      @simons = Enumerator.new do |steps|
8        a1 = 1
9        a2 = 2
10       a3 = 3
11       iteration = 1
12
13       loop do
14         fact = a1 * a2 * a3
15         steps << { iteration: iteration, f: fact, a1: a1, a2: a2, a3: a3, fits: factorial?(fact) }
16         a1 = a2
17         a2 = a3
18         a3 += 1
19         iteration += 1
20       end
21     end
22   end
23
24   def result
25     numbers = @simons.take_while { |step| step[:iteration] < @max_n }.select { |step| step[:fits] }
26
27     respond_to do |format|
28       format.xml { render xml: numbers.to_xml }
29       format.rss { render xml: numbers.to_xml }
30     end
31   end
32
33   protected
34
35   def parse_params
36     @max_n = params[:max_n].to_i
37   end
38
39   private
40
41   def factorial?(number)
42     i = 1
43     n = number.to_f
44     while (n % i).zero?
45       n /= i
46       i += 1
47     end
48
49     n == 1
50   end
51 end
52
```

```

WPL_bmstu - api_controller_test.rb

1 require 'test_helper'
2
3 class ApiControllerTest < ActionDispatch::IntegrationTest
4   test 'can get rss' do
5     get '/',
6       params: { max_n: 1000, format: :rss }
7     assert_response :success
8     assert_includes @response.headers['Content-Type'], 'application/rss'
9   end
10
11   test 'can get xml' do
12     get '/',
13       params: { max_n: 1000, format: :xml }
14     assert_response :success
15     assert_includes @response.headers['Content-Type'], 'application/xml'
16   end
17 end
18

```

```

WPL_bmstu - routes.rb

1 Rails.application.routes.draw do
2   root to: 'api#result'
3 end
4

```

Часть вторая - PROXY

```

WPL_bmstu - input.html.erb

1 <div class="container">
2   <h1>Гипотеза Симона</h1>
3   <p>Существует гипотеза Симона о факториале.<br/>
4   Она гласит, что существует 4 факториала, которые представимы в виде произведения трех последовательных чисел.<br/>
5   Например: 4! = 2 · 3 · 4.<br/>
6   Определить эту четверку, подсчитать действительное количество таких факториалов, если это возможно и тем самым доказать
   неправильность гипотезы.
7   <br/>
8   Вывести на печать результаты каждой итерации и финальных расчетов.</p>
9   <%= form_tag url_for(controller: :proxy, action: :output), method: 'get' do %>
10     <%= label_tag "Максимальное количество итераций n:" %>
11     <%= number_field_tag 'max_n', '100', min: 1 %>
12     <p>Выберите, где будет обрабатываться результат:</p>
13     <div>
14       <input type="radio" id="server-choice"
15         name="side" value="server">
16       <label for="server-choice">На сервере</label>
17       <input type="radio" id="client-choice"
18         name="side" value="client" checked="">
19       <label for="client-choice">XML в браузере</label>
20       <input type="radio" id="client-xslt-choice"
21         name="side" value="client-with-xslt">
22       <label for="client-xslt-choice">XML с помощью XSLT в браузере</label>
23     </div>
24     <br/>
25     <%= submit_tag 'Найти числа' %>
26   <% end %>
27 </div>

```

```

WPL_bmstu - proxy_controller_test.rb

1 require 'test_helper'
2
3 class ProxyControllerTest < ActionDispatch::IntegrationTest
4   test 'should get html' do
5     get '/proxy/output', params: { max_n: 1000, side: 'server' }
6     assert_response :success
7     assert_includes @response.headers['Content-Type'], 'text/html; charset=utf-8'
8   end
9   test 'should get xml for client' do
10    get '/proxy/output', params: { max_n: 1000, side: 'client' }
11    assert_response :success
12    assert_includes @response.headers['Content-Type'], 'application/xhtml+xml; charset=utf-8'
13  end
14  test 'should get xml+xslt for client' do
15    get '/proxy/output', params: { max_n: 1000, side: 'client-with-xslt' }
16    assert_response :success
17    assert_includes @response.headers['Content-Type'], 'application/xhtml+xml; charset=utf-8'
18  end
19 end
20

```

WPL_bmstu - proxy_controller.rb

```
1 require 'nokogiri'
2 require 'open-uri'
3 # Контроллер для запросов к нашему "API"
4 class ProxyController < ApplicationController
5   before_action :parse_params, only: :output
6   before_action :prepare_url, only: :output
7   def input; end
8
9   def output
10     api_response = URI.parse(@url).open
11     case @side
12     when 'server'
13       @result = xslt_transform(api_response).to_html
14     when 'client-with-xslt'
15       render xml: insert_browser_xslt(api_response).to_xml, content_type:
16         'application/xhtml+xml; charset=utf-8'
17     else
18       render xml: api_response, content_type: 'application/xhtml+xml; charset=utf-8'
19     end
20   end
21
22   private
23
24   BASE_API_URL = 'http://localhost:3000/?format=xml'.freeze
25   XSLT_SERVER_TRANSFORM = "#{Rails.root}/public/server_transform.xslt".freeze
26   XSLT_BROWSER_TRANSFORM = '/browser_transform.xslt'.freeze
27   def parse_params
28     @max_n = params[:max_n].to_i
29     @side = params[:side]
30   end
31
32   def prepare_url
33     @url = BASE_API_URL + "&max_n=#{@max_n}"
34   end
35
36   def xslt_transform(data, transform: XSLT_SERVER_TRANSFORM)
37     doc = Nokogiri::XML(data)
38     xslt = Nokogiri::XSLT(File.read(transform))
39     xslt.transform(doc)
40   end
41
42   def insert_browser_xslt(data, transform: XSLT_BROWSER_TRANSFORM)
43     doc = Nokogiri::XML(data)
44     xslt = Nokogiri::XML::ProcessingInstruction.new(doc,
45       'xml-stylesheet',
46       "type=\"text/xsl\" href=\"#{transform}\"")
47     doc.root.add_previous_sibling(xslt)
48     # Возвращаем doc, так как предыдущая операция возвращает не XML-документ.
49     doc
50   end
51 end
```

WPL_bmstu - output.html.erb

```
1 <h1>Результат (сервер)</h1>
2 <%= render inline: @result %>
```

WPL_bmstu - server_transform.xslt

```
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
4   <!--xsl:template говорит о том, что тут будет замена. match показывает, к какой части документа это применимо-->
5   <xsl:template match="/">
6     <!--Внутри шаблона пишем наше преобразование-->
7     <table class="simons-table">
8       <thead>
9         <tr>
10          <th>Факториал</th>
11          <th>Последовательность</th>
12        </tr>
13      </thead>
14
15      <!--Цикл-->
16      <xsl:for-each select="objects/object">
17        <!--Создание переменной-->
18        <xsl:variable name="counter" select="position()" />
19        <tbody>
20          <tr>
21            <th>
22              <!--Извлекаем значение из XML-тега-->
23              <xsl:value-of select="f"></xsl:value-of>
24            </th>
25            <th>
26              <xsl:value-of select="a1"></xsl:value-of>
27              *
28              <xsl:value-of select="a2"></xsl:value-of>
29              *
30              <xsl:value-of select="a3"></xsl:value-of>
31            </th>
32          </tr>
33        </tbody>
34      </xsl:for-each>
35    </table>
36  </xsl:template>
37 </xsl:stylesheet>
```

WPL_bmstu - browser_transform.xslt

```
1 <?xml version="1.0" encoding="UTF-8"?>
2
3 <xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
4   <!--xsl:template говорит о том, что тут будет замена. match показывает, к какой части документа это применимо-->
5   <xsl:template match="/">
6     <!--Внутри шаблона пишем наше преобразование-->
7     <html>
8       <head>
9         <title>Результат</title>
10      </head>
11      <body>
12        <h1>Результат (браузер)</h1>
13        <table class="simons-table">
14          <thead>
15            <tr>
16              <th>Факториал</th>
17              <th>Последовательность</th>
18            </tr>
19          </thead>
20
21          <!--Цикл-->
22          <xsl:for-each select="objects/object">
23            <tbody>
24              <tr>
25                <th>
26                  <!--Извлекаем значение из XML-тега-->
27                  <xsl:value-of select="f"></xsl:value-of>
28                </th>
29                <th>
30                  <xsl:value-of select="a1"></xsl:value-of>
31                  *
32                  <xsl:value-of select="a2"></xsl:value-of>
33                  *
34                  <xsl:value-of select="a3"></xsl:value-of>
35                </th>
36              </tr>
37            </tbody>
38          </xsl:for-each>
39        </table>
40      </body>
41    </html>
42  </xsl:template>
43 </xsl:stylesheet>
```

Результат:

localhost:3001

Гипотеза Симона

Существует гипотеза Симона о факториале.

Она гласит, что существует 4 факториала, которые представимы в виде произведения трех последовательных чисел.

Например: $4! = 2 \cdot 3 \cdot 4$.

Определить эту четверку, подсчитать действительное количество таких факториалов, если это возможно и тем самым доказать неправильность гипотезы.

Вывести на печать результаты каждой итерации и финальных расчетов.

Максимальное количество итераций n:

Выберите, где будет обрабатываться результат:

☐ На сервере

☒ XML в браузере

☐ XML с помощью XSLT в браузере

Найти числа

localhost:3001/proxy/output?max_n=100&side=server&commit=Найти+числа

Результат (сервер)

Факториал	Последовательность
6	1 * 2 * 3
24	2 * 3 * 4
120	4 * 5 * 6
720	8 * 9 * 10

localhost:3001/proxy/output?max_n=100&side=client-with-xslt&commit=Найти+числа

Результат (браузер)

Факториал	Последовательность
6	1 * 2 * 3
24	2 * 3 * 4
120	4 * 5 * 6
720	8 * 9 * 10

localhost:3001/proxy/output?max_n=100&side=client&commit=Найти+числа

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<?xml version="1.0" encoding="UTF-8" ?>
<objects type="array">
  <object>
    <iteration type="integer">1</iteration>
    <f type="integer">6</f>
    <a1 type="integer">1</a1>
    <a2 type="integer">2</a2>
    <a3 type="integer">3</a3>
    <fits type="boolean">true</fits>
  </object>
  <object>
    <iteration type="integer">2</iteration>
    <f type="integer">24</f>
    <a1 type="integer">2</a1>
    <a2 type="integer">3</a2>
    <a3 type="integer">4</a3>
    <fits type="boolean">true</fits>
  </object>
  <object>
    <iteration type="integer">4</iteration>
    <f type="integer">120</f>
    <a1 type="integer">4</a1>
    <a2 type="integer">5</a2>
    <a3 type="integer">6</a3>
    <fits type="boolean">true</fits>
  </object>
  <object>
    <iteration type="integer">8</iteration>
    <f type="integer">720</f>
    <a1 type="integer">8</a1>
    <a2 type="integer">9</a2>
    <a3 type="integer">10</a3>
    <fits type="boolean">true</fits>
  </object>
</objects>
```


Тестирование:

```
PS C:\Users\might\Desktop\BMSTU\sem_3\WPL_bmstu\Lab10\Lab10_api> rails test
Running 2 tests in a single process (parallelization threshold is 50)
Run options: --seed 25247
```

```
# Running:
```

```
..
```

```
Finished in 0.555399s, 3.6010 runs/s, 10.8030 assertions/s.
```

```
2 runs, 6 assertions, 0 failures, 0 errors, 0 skips
```

```
PS C:\Users\might\Desktop\BMSTU\sem_3\WPL_bmstu\Lab10\Lab10_api> |
```

```
PS C:\Users\might\Desktop\BMSTU\sem_3\WPL_bmstu\Lab10\Lab10_proxy> rails test
Running 3 tests in a single process (parallelization threshold is 50)
Run options: --seed 7315
```

```
# Running:
```

```
...
```

```
Finished in 0.927214s, 3.2355 runs/s, 9.7065 assertions/s.
```

```
3 runs, 9 assertions, 0 failures, 0 errors, 0 skips
```

```
PS C:\Users\might\Desktop\BMSTU\sem_3\WPL_bmstu\Lab10\Lab10_proxy> |
```

Итоговый код данной лабораторной работы доступен по ссылке:

https://github.com/tenessinum/WPL_bmstu/tree/main/Lab10