



**«Московский государственный технический университет  
имени Н.Э. Баумана»  
(МГТУ им. Н.Э. Баумана)**

---

ФАКУЛЬТЕТ \_\_\_\_\_ ИНФОРМАТИКИ И СИСТЕМ УПРАВЛЕНИЯ  
КАФЕДРА \_\_\_\_\_ КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ 6)


**О т ч е т**

**по лабораторной работе № 10**

**Дисциплина: Языки программирования**

**Название лабораторной работы: Формирование и отображение XML в  
HTML средствами сервера и клиента**

Студент гр. ИУ6-34

  
\_\_\_\_\_  
(Подпись, дата)

**Габолаев Г.К.**

(И.О. Фамилия)

Преподаватель

\_\_\_\_\_  
(Подпись, дата)

**Изория В.Ф.**

(И.О. Фамилия)

Москва, 2017

## 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

## Коды программ:

```
> cat xslt-api/app/controllers/xml_controller.rb
class XmlController < ApplicationController
  include XmlHelper

  before_action :check_params, only: :index

  def index
    data = if @error.nil?
             find_factorials(@to_value)
           else
             { message: "Неверный формат входных данных. Значение =
#{@to_value}" }
           end

    respond_to { |format|
      format.rss { render xml: data.to_xml }
      format.xml { render xml: data.to_xml }
    }
  end

  protected

  def check_params

    @to_value = params[:to_value]
    if @to_value =~ /^[0-9]+$/
      @to_value = @to_value.to_i
      @error = true if @to_value <= 2 || @to_value > 15
    else
      @error = true
    end
  end
end
```

```

> cat xslt-api/transform.xslt
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="/">
    <html>
      <head>
        <title>ОТВЕТ</title>
        <style>
          h2 {
            font-family: "Courier New", serif;
          }
          table {
            border: solid 2px black;
            border-collapse: collapse;
          }
          td {
            text-align: center;
            border: 2px solid black;
          }
        </style>
      </head>
      <body>
        <xsl:choose>
          <xsl:when test="/*/message">
            <h2>
              <xsl:value-of select="."/>
            </h2>
          </xsl:when>
          <xsl:otherwise>
            <h2>Количество найденных факториалов:
              <xsl:value-of select="count(/*/results/*)"/>
            </h2>
            <xsl:for-each select="/*/results/result">
              <p>
                <xsl:value-of select="."/>
              </p>
            </xsl:for-each>
            <table>
              <xsl:for-each select="/*/for-facts/for-fact">
                <xsl:variable name="ffPos" select="position()"/>
                <tr>
                  <td><xsl:value-of select="$ffPos + 2"/>! =
                    <xsl:value-of select="."/>
                  </td>
                  <td>
                    <xsl:for-each select="/*/iters/iter[$ffPos]/iter[not(@nil='true')]">
                      <p>
                        <xsl:value-of select="."/>
                      </p>
                    </xsl:for-each>
                  </td>
                </tr>
              </xsl:for-each>
            </table>
          </xsl:otherwise>
        </xsl:choose>
      </body>
    </html>
  </xsl:template>
</xsl:stylesheet>

```

```

> cat xslt-proxy/app/controllers/proxy_controller.rb
require 'nokogiri'
require 'open-uri'

class ProxyController < ApplicationController
  before_action :parse_params, only: :output
  before_action :prepare_url, only: :output

  def input; end

  def output
    api_response = open(@url)

    case @side
    when 'client'
      render xml: api_response
    when 'client_with_xslt'
      render xml: insert_browser_xslt(api_response).to_xml
    when 'server'
      @result = apply_xslt_transform(api_response).to_html
    end
  end

  private

  BASE_API_URL = 'http://localhost:3000/?format=xml'.freeze
  XSLT_SERVER_TRANSFORM = "#{Rails.root}/public/server_transform.xslt".freeze
  XSLT_BROWSER_TRANSFORM = '/browser_transform.xslt'.freeze

  def parse_params
    @to_value = params[:to_value]
    @side = params[:side]
  end

  def prepare_url
    @url = BASE_API_URL + "&to_value=#{@to_value}"
  end

  def apply_xslt_transform(data, transform: XSLT_SERVER_TRANSFORM)
    doc = Nokogiri::XML(data)
    xslt = Nokogiri::XSLT(File.read(transform))
    xslt.transform(doc)
  end

  def insert_browser_xslt(data, transform: XSLT_BROWSER_TRANSFORM)
    doc = Nokogiri::XML(data)
    xslt = Nokogiri::XML::ProcessingInstruction.new(doc,
                                                    'xml-stylesheet',
                                                    "type=\"text/xsl\"")
    href = "#{transform}\"")
    doc.root.add_previous_sibling(xslt)
    doc
  end
end

> cat xslt-proxy/public/server_transform.xslt
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  <xsl:template match="/">
    <xsl:choose>
      <xsl:when test="/*/message">
        <h2>
          <xsl:value-of select="."/>
        </h2>
      </xsl:when>
    </xsl:choose>
  </xsl:template>
</xsl:stylesheet>

```

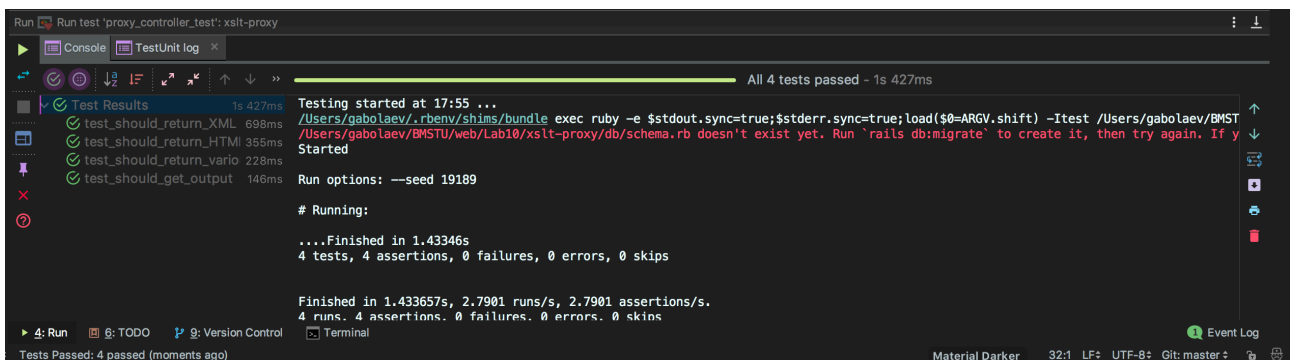
```

<xsl:otherwise>
  <h2>Количество найденных факториалов:
  <xsl:value-of select="count(/*/results/*)"/>
</h2>
<xsl:for-each select="*/results/result">
  <p>
    <xsl:value-of select="."/>
  </p>
</xsl:for-each>
<table>
  <xsl:for-each select="*/for-facts/for-fact">
    <xsl:variable name="ffPos" select="position()"/>
    <tr>
      <td><xsl:value-of select="$ffPos + 2"/>! =
      <xsl:value-of select="."/>
    </td>
    <td>
      <xsl:for-each select="*/iters/iter[$ffPos]/iter[not(@nil='true')]">
        <p>
          <xsl:value-of select="."/>
        </p>
      </xsl:for-each>
    </td>
    </tr>
  </xsl:for-each>
</table>
</xsl:otherwise>
</xsl:choose>
</xsl:template>
</xsl:stylesheet>

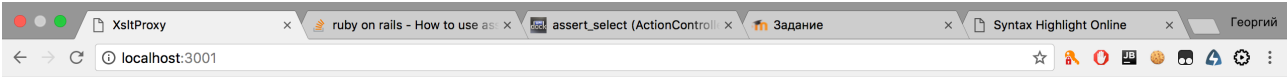
```

Файл browser\_transform.xslt является полной копией файла transform.xslt

## Тесты:



## Результат работы:



### Легенда

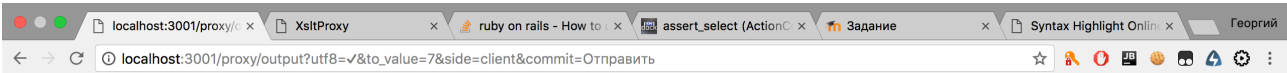
Существует гипотеза Симона о факториале. Она гласит, что существует 4 факториала, которые представимы в виде произведения трех последовательных чисел. Например:  $4! = 2 \cdot 3 \cdot 4$ .

Введите число (не больше 15), все факториалы до которого будем анализировать на поиск вышеуказанных 4 чисел.

### Выбор стороны рендеринга

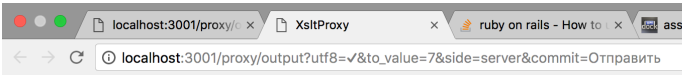
- ☒ Рендеринг на стороне сервера этого приложения
- ☐ Отображение сырого XML на стороне браузера
- ☐ Рендеринг на стороне браузера с помощью XSLT

Отправить



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" ?>
<hash>
  <results type="array">
    <result type="float">6.0</result>
    <result type="float">24.0</result>
    <result type="float">120.0</result>
    <result type="float">720.0</result>
  </results>
  <for-facts type="array">
    <for-fact type="float">6.0</for-fact>
    <for-fact type="float">24.0</for-fact>
    <for-fact type="float">120.0</for-fact>
    <for-fact type="float">720.0</for-fact>
    <for-fact type="float">5040.0</for-fact>
  </for-facts>
  <iters type="array">
    <iter type="array">
      <iter>1 * 2 * 3 = 6</iter>
    </iter>
    <iter type="array">
      <iter nil="true"/>
      <iter>2 * 3 * 4 = 24</iter>
    </iter>
    <iter type="array">
      <iter nil="true"/>
      <iter>3 * 4 * 5 = 60</iter>
      <iter>4 * 5 * 6 = 120</iter>
    </iter>
    <iter type="array">
      <iter nil="true"/>
      <iter>4 * 5 * 6 = 120</iter>
      <iter>5 * 6 * 7 = 210</iter>
      <iter>6 * 7 * 8 = 336</iter>
      <iter>7 * 8 * 9 = 504</iter>
      <iter>8 * 9 * 10 = 720</iter>
    </iter>
    <iter type="array">
      <iter nil="true"/>
      <iter>5 * 6 * 7 = 210</iter>
      <iter>6 * 7 * 8 = 336</iter>
      <iter>7 * 8 * 9 = 504</iter>
    </iter>
  </iters>
</hash>
```

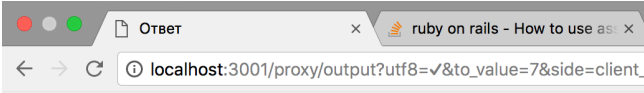


### Количество найденных факториалов: 4

6.0  
24.0  
120.0  
720.0

3! = 6.0	1 * 2 * 3 = 6
4! = 24.0	2 * 3 * 4 = 24
5! = 120.0	3 * 4 * 5 = 60
6! = 720.0	4 * 5 * 6 = 120
	4 * 5 * 6 = 120
	5 * 6 * 7 = 210
	6 * 7 * 8 = 336
	7 * 8 * 9 = 504
	8 * 9 * 10 = 720
	5 * 6 * 7 = 210
	6 * 7 * 8 = 336
	7 * 8 * 9 = 504

output.xml



### Количество найденных факториалов: 4

6.0  
24.0  
120.0  
720.0

3! = 6.0	1 * 2 * 3 = 6
4! = 24.0	2 * 3 * 4 = 24
5! = 120.0	3 * 4 * 5 = 60
6! = 720.0	4 * 5 * 6 = 120
	4 * 5 * 6 = 120
	5 * 6 * 7 = 210
	6 * 7 * 8 = 336
	7 * 8 * 9 = 504
	8 * 9 * 10 = 720
	5 * 6 * 7 = 210
	6 * 7 * 8 = 336
	7 * 8 * 9 = 504
	8 * 9 * 10 = 720
	9 * 10 * 11 = 990
	10 * 11 * 12 = 1320
7! = 5040.0	11 * 12 * 13 = 1716

output.xml