Языки Интернет-программирования

Лекция 11. Языки запросов и преобразования XML

- XMLТехнологии
- XSL, XSLT
- XPath, XQuery
- XML СУБД

МГТУ им. Н.Э. Баумана, доц. каф. ИУ-6,
к.т.н. Самарев Роман Станиславович

samarev@acm.org

13.11.2018

XML для разработчика

13.11.2018

- Программы для работы с XML
  - Eclipse XSL Developer Tools
  - Atom IDE
  - XMLSpy XML Editor
  - WmHelp XmlPad
  - https://en.wikipedia.org/wiki/Comparison_of_XML_editors
  - …
- XML-СУБД
  - BaseX. The XML Framework - http://basex.org
  - eXist-db XML Database- http://exist-db.org/
  - Oracle Berkeley DB XML - https://www.oracle.com/database/berkeley-db/xml.html
  - Sedna XML Database – http://www.sedna.org
  - …

XML-технологии

13.11.2018

- XML
- XSL
- XSLT
- XPath
- XQuery
- XSL-FO
- EXSLT
- …

XSL (eXtensible Stylesheet Language)

13.11.2018

- XSL Transformations (XSLT) — язык преобразований XML-документов.
- XSL Formatting Objects (XSL-FO) — язык разметки типографских макетов и иных предпечатных материалов.
- XPath — язык путей и выражений, используемый в XSLT для доступа к отдельным частям XML-документа.

XPath

- XML Path Language (XPath) Version 1.0
  W3C Recommendation 16 November 1999
- XML Path Language (XPath) 2.0
  W3C Recommendation 14 December 2010
- XML Path Language (XPath) 3.0
  W3C Recommendation on 8 April 2014

XPath

13.11.2018

- набор синтаксических правил для адресации частей XML-документа.
- главная задача – найти нужный фрагмент (элемент, атрибут) документа XML
- Средства для проверки
  - http://chris.photobooks.com/xml/default.htm
  - http://www.online-toolz.com/tools/xpath-editor.php
  - http://www.freeformatter.com/xpath-tester.html

Пример XML-документа для демонстрации XPath

13.11.2018

```
<?xml version="1.0"?>
<A>
    <B b="b1">B1</B>
    <C a="a1">C1</C>
    <E e="e1">
        <B a="a2">B2</B>
        <G>G</G>
    </E>
    <B b="b3">B3</B>
    <numbers>
        <number id="1">2</number>
        <number id="2">4</number>
        <number id="3">8</number>
    </numbers>
</A>
```

- Примеры из: Г.И. Ревунков, Ю.Е. Гапанюк. «Введение в XML–технологии». Электронное учебное издание. Учебное пособие по дисциплине «XML–технологии». М: МГТУ им. Н.Э. БАУМАНА, 2012

Основные выражения XPath

13.11.2018

/	Корневой элемент.
A	Элемент A.
/A	Элемент A, вложенный в корневой элемент.
A/B	Элемент B, вложенный в элемент A.
A//B	Элемент B, вложенный на любой глубине в элемент A.
/A/*	Любой элемент, вложенный в A, вложенный в корневой элемент.

Основные выражения XPath

13.11.2018

/A/E/B	Элемент B, вложенный в E, вложенный в A, вложенный в корневой элемент.
A/B | A/C	Все элементы B или C, вложенные в A, символ « | » – оператор объединения множеств.
A//*	Все элементы, вложенные в A на любой глубине.
//B/@b	Атрибуты b, вложенные в элементы B.
//*/@*	Все атрибуты всех элементов документа.

XPath
Фильтры и сравнения

13.11.2018

A/B[1]	Первый элемент B, вложенный в A.
A/B[last()]	Последний элемент B, вложенный в A.
//A[(B or C) and E]	Поиск такого A, в который вложены элементы B или C и вложен E.
//A[B and not(Z)]	Поиск такого элемента A, в который вложен B и не вложен Z.
//A[B!="str1"]	Поиск такого элемента A, у которого есть вложенный элемент B не равный "str1".
//B[@b="b1"]	Поиск элемента B с атрибутом b="b1".

XPath
Фильтры и сравнения

13.11.2018

//C[. = "C1"]	Поиск элемента C = C1.
/A//E[@e="e1" and B="B2"]/G	Все эл. G, вложенные в E. Элемент E вложен в A на любой глубине. A непосредственно вложен в корневой. E должен содержать атрибут e=e1 и вложенный элемент B=B2.
//number[@id<2 or @id>2]	Поиск элементов number, у которых атрибут id<2 или атрибут id>2.
//text()	Все текстовые узлы документа.

Функции выборки специальных конструкций

13.11.2018

- text() – любой текстовый узел.
- node() – любой узел, который не является атрибутом и корневым элементом.
- comment() – комментарий.
- processing-instruction() – инструкция обработки.

Оси выборки

13.11.2018

Название	Описание	Сокр. форма
self	Текущий узел	.
child	Непосредственно вложенные узлы	/
parent	Родительский узел	..
descendant	Потомки узла на любой глубине вложенности	//
descendant-or-self	Узел и его потомки	
ancestor	Предки узла	
ancestor-or-self	Сам узел и его предки	
following	Все узлы после данного	
following-sibling	Все узлы этого же уровня после данного	
preceding	Все узлы перед данным	
preceding-sibling	Все узлы этого же уровня перед данным	
attribute	Узлы атрибутов	@
namespace	Узлы пространства имен	

Пример

//A/child::* или //A/*	Элементы, непосредственно вложенные в A.
//A/descendant::* или //A//*	Элементы, вложенные в A на любой глубине.
//A/descendant-or-self::*	Элементы, вложенные в A на любой глубине и сам элемент A.
A/E/preceding-sibling::*	Узлы, следующие перед E на том же уровне иерархии.

Функции в XPath

13.11.2018

- Булевые функции
- Числовые функции
  - number number (object?)
  - number sum (node-set)
  - …
- Строковые функции
  - string concat (string, string, string*)
  - boolean starts-with (string, string)
  - …
- Функции множеств узлов
  - number last ()
  - node-set id (object)
  - …

XPath 2.0

13.11.2018

- язык выражений для обработки последовательностей со встроенной поддержкой XML-документов
- является строгим синтаксическим подмножеством языка XQuery 1.0.
- существует совместно с XPath 1.0
- http://www.xml.com/pub/a/2002/03/20/xpath2.html
- http://www.xmlhack.ru/texts/02/xpath20/xpath20.html

XQuery

13.11.2018

- http://www.w3.org/TR/xquery/
- XQuery - язык запросов для обработки данных в формате XML
- Включает в себя XPath 2.0

Пример XML для преобразования

13.11.2018

```
<bib>
  <book>
    <title>TCP/IP Illustrated</title>
    <author>Stevens</author>
    <publisher>Addison-Wesley</publisher>
  </book>
  <book>
    <title>Advanced Programming
          in the Unix Environment</title>
    <author>Stevens</author>
    <publisher>Addison-Wesley</publisher>
  </book>
  <book>
    <title>Data on the Web</title>
    <author>Abiteboul</author>
    <author>Buneman</author>
    <author>Suciu</author>
  </book>
</bib>
```

Результат трансформации в HTML

13.11.2018

```
<authlist>
  <author>
    <name>Abiteboul</name>
    <books>
      <title>Data on the Web</title>
    </books>
  </author>
  <author>
    <name>Buneman</name>
    <books>
      <title>Data on the Web</title>
    </books>
  </author>
  <author>
    <name>Stevens</name>
    <books>
      <title>Advanced Programming
            in the Unix Environment</title>
      <title>TCP/IP Illustrated</title>
    </books>
  </author>
...
```

XQuery запрос для преобразования

13.11.2018

```
<authlist>
 {
   for $a in fn:distinct-values($bib/book/author)
   order by $a
   return
     <author>
       <name> {$a} </name>
       <books>
         {
           for $b in $bib/book[author = $a]
           order by $b/title
           return $b/title
         }
       </books>
     </author>
 }
</authlist>
```

XSLT
Extensible Stylesheet Language Transformations

13.11.2018

- XSL Transformations (XSLT) Version 1.0
  W3C Recommendation 16 November 1999
- XSL Transformations (XSLT) Version 2.0
  W3C Recommendation 23 January 2007
- Предназначен для преобразования XML

XSL-FO
XSL Formatting Objects

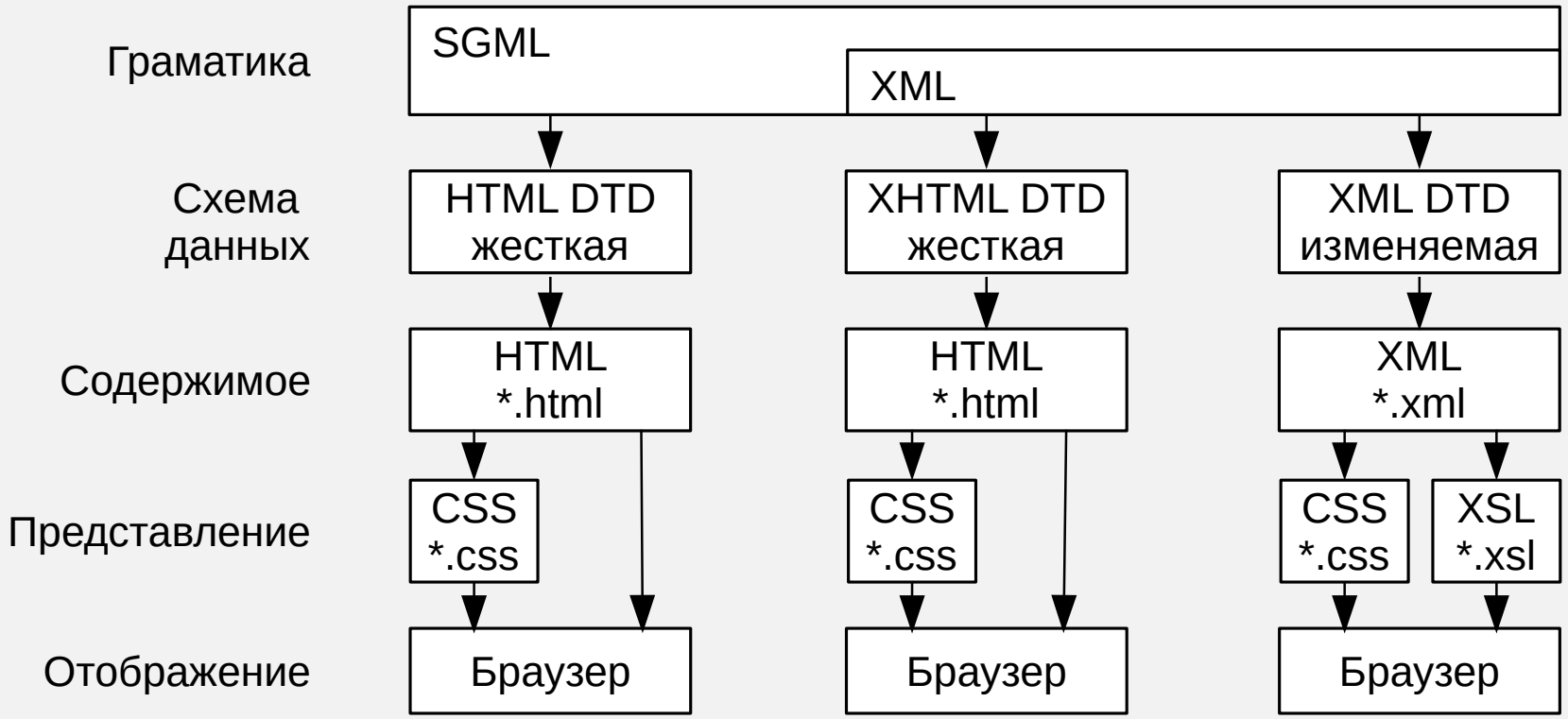
13.11.2018

- Язык разметки для дальнейшей генерации печатных документов (PDF, PS, RTF, PNG)

- https://www.w3.org/TR/xsl11/
- https://xmlgraphics.apache.org/fop/
- https://www.ibm.com/developerworks/ru/library/x-xstrmfo/index.html

HTML, XHTML, XML
Представление данных

13.11.2018



Пример преобразования XML

13.11.2018

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<?xml-stylesheet type="text/xsl" href="some_transformer.xslt"?>
<!-- Edited by XMLSpy® -->
<catalog>
    <cd>
        <title>Empire Burlesque</title>
        <artist>Bob Dylan</artist>
        <country>USA</country>
        <company>Columbia</company>
        <price>10.90</price>
        <year>1985</year>
    </cd>
    <cd>
        <title>Hide your heart</title>
        <artist>Bonnie Tyler</artist>
        <country>UK</country>
        <company>CBS Records</company>
        <price>9.90</price>
        <year>1988</year>
    </cd>
<catalog>
```

Результат преобразования в HTML

13.11.2018

```
<html><body>
<h2>My CD Collection</h2>
<table border="1">
<tr bgcolor="#9acd32">
<th>Title</th>
<th>Artist</th>
</tr>
<tr>
<td>Empire Burlesque</td>
<td>Bob Dylan</td>
</tr>
<tr>
<td>Hide your heart</td>
<td>Bonnie Tyler</td>
</tr>
...
</table>
</body></html>
```

Правила преобразования XSL

13.11.2018

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
  <html>
  <body>
  <h2>My CD Collection</h2>
    <table border="1">
      <tr bgcolor="#9acd32">
        <th>Title</th>
        <th>Artist</th>
      </tr>
      <xsl:for-each select="catalog/cd">
      <tr>
        <td><xsl:value-of select="title"/></td>
        <td><xsl:value-of select="artist"/></td>
      </tr>
      </xsl:for-each>
    </table>
  </body>
  </html>
</xsl:template>
</xsl:stylesheet>
```

Основы XSL

13.11.2018

- <?xml version="1.0" encoding="ISO-8859-1"?>
- <xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
  - Подключение пространства имён
  - xsl:stylesheet синоним xsl:transform.
- <xsl:template match="/">
  - Процедура обработки
  - match – XPath запрос

Основы XSL

13.11.2018

- Тэги без xsl: выводятся без изменений
- <xsl:for-each select="catalog/cd">
  - for-each – единственный вид цикла
  - select – область применения
- <xsl:value-of select="title"/>
  - Получение значения, соответствующего select (XPath-запрос от текущего уровня)

Шаблон с жестко заданной структурой

- Порядок элементов во входном документе не влияет на результат преобразования

- В шаблоне жестко заданы позиции элементов:

```
<td><xsl:value-of select="title"/></td>
<td><xsl:value-of select="artist"/></td>
```

Пример XML для преобразования

13.11.2018

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="Example_1.xsl"?>
<languages>
  <language id="1">
    <name>HTML</name>
    <year>01.01.1990</year>
    <howold>19</howold>
  </language>
  <language id="2">
    <year>01.01.1998</year>
    <howold>11</howold>
    <name>XML</name>
  </language>
  <language id="3">
    <name>SGML</name>
    <year>01.01.1986</year>
    <howold>23</howold>
  </language>
</languages>
```

Ожидаемый результат

Номер: 1
Наименование языка: **HTML**
Год создания: 01.01.1990
Возраст технологии (лет): *19*

Номер: 2
Год создания: 01.01.1998
Возраст технологии (лет): *11*
Наименование языка: **XML**

Номер: 3
Год создания: 01.01.1986
Наименование языка: **SGML**
Возраст технологии (лет): *23*

<<<<<<<<<< >>>>>>>>>>

Шаблоны без заданной структуры

13.11.2018

- Последовательность определяется входным документом

```
<?xml version="1.0" encoding="Windows-1251"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">

<!-- ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
<!-- Шаблон обработки корневого элемента XML - документа -->
<xsl:template match="/">
    <HTML>
    <HEAD>
    <LINK href="met.css" rel="stylesheet" type="text/css"/>
    </HEAD>
    <BODY>
    <xsl:apply-templates/>
    </BODY>
    </HTML>
</xsl:template>
<!-- ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
```

Шаблоны без заданной структуры. Продолжение.

13.11.2018

```
<!-- ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
<!-- Шаблон обработки элемента languages -->

<xsl:template match="languages">
<!-- Вызов шаблона для элемента language (шаблон вызывается необходимое
     количество раз) -->

     <xsl:apply-templates/>

</xsl:template>

<!-- ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
```

Шаблоны без заданной структуры. Продолжение.

13.11.2018

```
<!-- ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
<!-- Шаблон обработки элемента language -->
<xsl:template match="language">
  <!-- Получение значения атрибута id (префикс @ означает атрибут) -->
  <B>Номер: <xsl:value-of select="@id"/></B><BR/>
  <!-- Вызов шаблонов для элементов name, year и howold -->
  <xsl:apply-templates/>
  <HR/>
</xsl:template>
<!-- ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->

<!-- ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
<!-- Шаблон обработки элемента name -->
<xsl:template match="name">
  Наименование языка: <B><xsl:value-of select="."/></B><BR/>
  <!-- select="." - получение значения текущего элемента -->
</xsl:template>
<!-- ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
```

Шаблоны без заданной структуры. Продолжение.

13.11.2018

```
<!-- ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
<!-- Шаблон обработки элемента year -->
<xsl:template match="year">
  Год создания: <U><xsl:value-of select="."/></U><BR/>
</xsl:template>
<!-- ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->

<!-- ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
<!-- Шаблон обработки элемента howold -->
<xsl:template match="howold">
  Возраст технологии (лет): <I><xsl:value-of select="."/></I><BR/>
</xsl:template>
<!-- ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++ -->
</xsl:stylesheet>
```

Условия применения шаблона

- <xsl:template match="languages">
- <xsl:template match="language">
- <xsl:template match="name">
  - <xsl:value-of select="."/> или <xsl:value-of select="self::*"/>
  - Контекст применения внутри name

Поглощение CDATA

13.11.2018

- Исходный фрагмент

```
<CDATA_Example>
    <![CDATA[<<<<<<<<<<
>>>>>>>>>>]]>
</CDATA_Example>
```

- Правило для поглощения

```
<xsl:template match="CDATA_Example"/>
```

Создание узлов HTML

13.11.2018

- <xsl:comment>Комментарий в выходном документе</xsl:comment>

- <xsl:element name="P" use-attribute-sets="**p_attrs**">
  <xsl:attribute name="align">center</xsl:attribute>Параграф текста
  </xsl:element>

- <xsl:attribute-set name="**p_attrs**">
  <xsl:attribute name="title">Подсказка</xsl:attribute>
  <xsl:attribute name="onclick">alert('Подсказка')
  </xsl:attribute>
  </xsl:attribute-set>

Ветвление

13.11.2018

```
<xsl:template match="language">
<!-- Получение значения атрибута id (префикс @ означает атрибут) -->
<xsl:choose>

  <xsl:when test="name[.='XML']">
     <B><I>Номер: <xsl:value-of select="@id"/></I></B><BR/>
  </xsl:when>

  <xsl:otherwise>
     Номер: <xsl:value-of select="@id"/><BR/>
  </xsl:otherwise>

</xsl:choose>
```

Вызов шаблона как функции

13.11.2018

```
<!-- Вызов шаблона с помощью xsl:call-template для элемента howold -->
<xsl:call-template name="Howold_Function">
    <!--<xsl:with-param name="ParamHowold" select="./howold"/>-->
    <xsl:with-param name="ParamHowold">
        <xsl:value-of select="howold"/>
    </xsl:with-param>
</xsl:call-template>

<xsl:template name="Howold_Function">
    <xsl:param name="ParamHowold" select="0"/>
    <I>Возраст технологии (лет):
     <xsl:value-of select="$ParamHowold"/></I><BR/>
</xsl:template>
```

Проверка простых условий

13.11.2018

```
<xsl:if test="number(substring(.,7,4)) &lt; number('1995')">
  <xsl:attribute name="align">right</xsl:attribute>
</xsl:if>
```

Включение стилей (правил)

13.11.2018

- <xsl:include href="math_include.xsl"/>

```
<?xml version="1.0"?>
 <xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
   version="1.0">
  <xsl:template match="op[@symbol='+']">
    <xsl:value-of select="operand[1]"/>
    <xsl:value-of select="@symbol"/>
    <xsl:value-of select="operand[2]"/>
    =
    <xsl:value-of select="sum(operand)"/>
  </xsl:template>
</xsl:stylesheet>
```

Импорт стилей (правил)

13.11.2018

- Операция переопределяет правила

Сортировка элементов

13.11.2018

```
<xsl:variable name="LangVar" select="languages"/>
<!-- В переменную LangVar помещается значение
    элемента languages -->

<xsl:for-each select="$LangVar/language">

<xsl:sort select="howold" data-type="number"
    order="ascending"/>
<!-- Перебор в цикле всех элементов language,
    вложенных в элемент languages. Сортировка по
    возрастанию значения элемента howold. -->
```

Формирование текстового документа

13.11.2018

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
<xsl:output method="text" encoding="UTF-8"/> 93

<!-- output определяет формат выходного документа. method="text" - выходной документ:
     (возможны "text", "xml", "html"). encoding - кодировка выходного документа -->
<xsl:template match="/">
<!-- Правило обработки корневого элемента XML - документа -->
<xsl:for-each select="languages/language">
<!-- Перебор в цикле всех элементов language, вложенных в элемент languages. -->
<xsl:text>&lt;</xsl:text>
<!-- text формирует текстовый узел в выходном документе.
В этом примере формируется тэговая скобка -->
    <xsl:value-of select="@id"/>,
      <xsl:value-of select="name"/>,
      <xsl:value-of select="year"/>,
      <xsl:value-of select="howold"/>
      <xsl:text>&gt;</xsl:text>
    </xsl:for-each>
  </xsl:template>
</xsl:stylesheet>
```

Копирование узлов

13.11.2018

<xsl:copy> узлы для копирования </xsl:copy>

<xsl:copy-of select="XPath-выражение, задающее
узлы для копирования"/>

XInclude
вставка других документов

13.11.2018

```
<?xml version="1.0"?>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:xi="http://www.w3.org/2001/XInclude">
  <head>...</head>
  <body>
    ...
    <p><xi:include href="license.txt" parse="text"/></p>
  </body>
</html>
```

- Получаем:

```
<?xml version="1.0"?>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:xi="http://www.w3.org/2001/XInclude">
  <head>...</head>
  <body>
    ...
    <p>This document is published under GNU Free Documentation License</p>
  </body>
</html>
```

XLink
XML Linking Language

13.11.2018

```
<?xml version="1.0"?>
<document xmlns="http://example.org/xmlns/2002/document" xmlns:xlink="http://
    www.w3.org/1999/xlink">


  <heading id="someHeading">Some Document</heading>


    <para>Here is
      <anchor xlink:type="simple" xlink:href="#someHeading">
        a link</anchor> to the header.
     </para>


  <para>It is an anchor that points to the element with the id "someHeading" on
     the current page.</para>
</document>
```

XPointer - язык идентификации фрагментов ресурсов

13.11.2018

```
<foobar id="foo">
  <bar/>
  <baz>
    <bom a="1"/>
  </baz>
  <bom a="2"/>
</foobar>
```

- Получаем:

xpointer(id("foo")) => foobar
xpointer(/foobar/1) => bar
xpointer(//bom) => bom (a=1), bom (a=2)
element(/1/2/1) => bom (a=1) (/1 descend into first element (foobar),
/2 descend into second child element (baz),
/1 select first child element (bom))

Место преобразования XML->HTML

13.11.2018

- Преобразование браузером
  - + минимизация трафика
  - + разгрузка сервера
  - - вычислительная нагрузка на клиента
- Преобразование сервером
  - + не зависим от браузера
  - - вычислительная нагрузка на сервер

Ruby + Nokogiri
Преобразование XSLT

13.11.2018

- gem install nokogiri

```
require 'nokogiri'

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

XML-СУБД

13.11.2018

- Объект хранения: XML-документ
- Язык запросов: XPath или XQuery
- Отсутствует строгая схема данных
- Результат:
  - в формате XML
  - содержит документы
  - содержит отдельные узлы

- Целесообразно использовать, если нужен доступ к узлам XML-документов

Литература

- Г.И. Ревунков, Ю.Е. Гапанюк. Введение в XML – технологии. Учебное пособие. МГТУ им. Баумана.- 2010

- http://www.w3schools.com/xsl/
- http://www.w3schools.com/xpath/default.asp
- http://www.sedna.org/quick-start.html

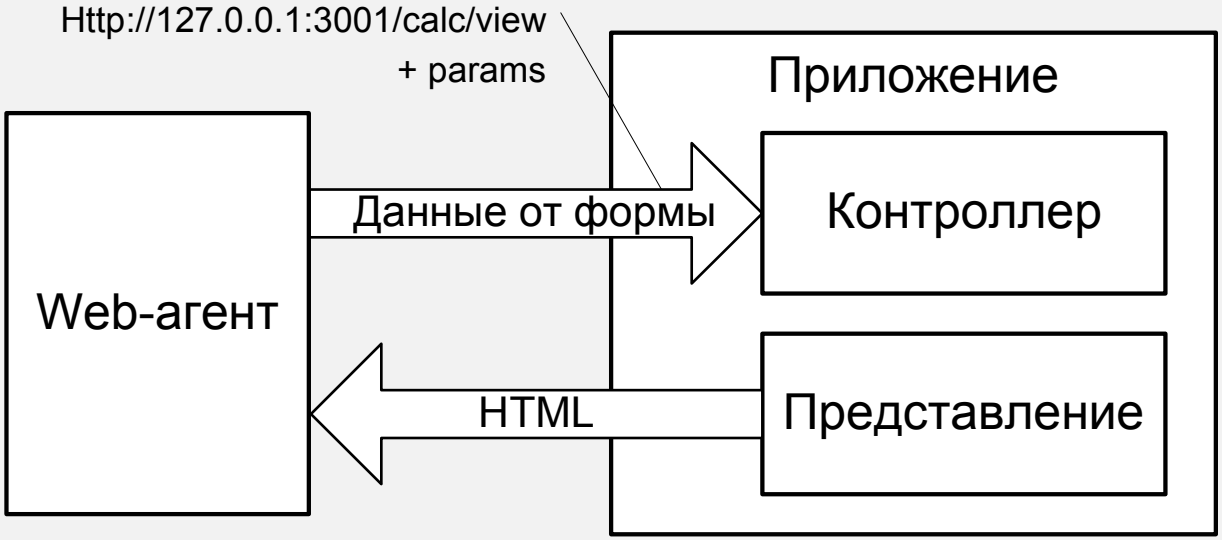
Выполнение лабораторных работ 9 и 10

13.11.2018

Базовая реализация в ЛР 8

13.11.2018

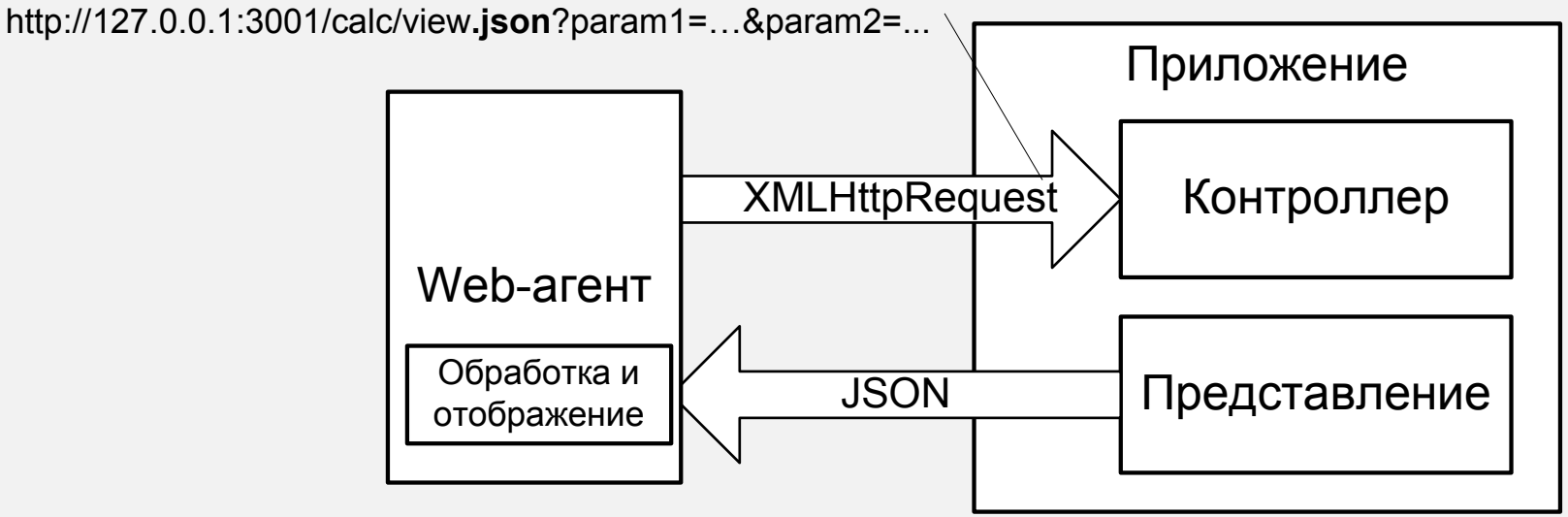
- Формирование ответа в формате HTML



ЛР 9

13.11.2018

- Формирование ответа в формате JSON



ЛР 9

13.11.2018

- Формирование отклика в формате html, json, xml...

```
respond_to do |format|
    format.xml { render json: data }
    format.xml { render xml: data}
    format.any # index.html.erb
end
```

ЛР 10. Преобразование XML в HTML с помощью XSLT

13.11.2018