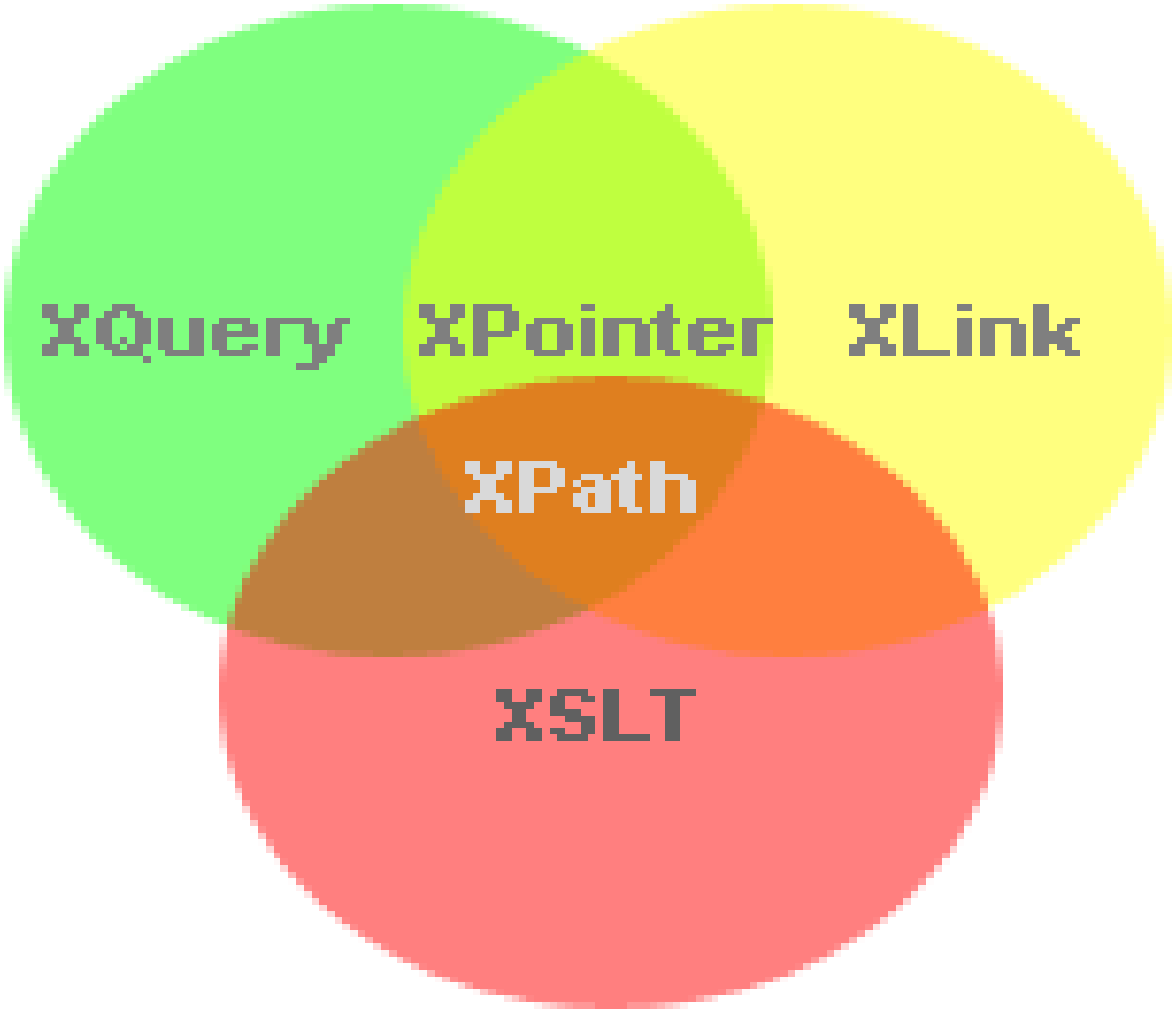
Лабораторная работа №2

XPath

XPath



XPath

XPath (XML Path Language) — *язык запросов к элементам XML-документа*. Разработан для организации доступа к частям документа XML в файлах трансформации XSLT и является стандартом консорциума W3C. XPath призван реализовать навигацию по DOM в XML. В XPath используется компактный синтаксис, отличный от принятого в XML. В 2007 году завершилась разработка версии 2.0, которая теперь является составной частью языка XQuery 1.0. В декабре 2009 года началась разработка версии 2.1, которая использует XQuery 1.1.

На данный момент, самой популярной версией является XPath 1.0. Это связано с отсутствием поддержки XPath 2.0 со стороны открытых библиотек. В частности, речь идёт о LibXML, от которой зависит поддержка языка в браузерах с одной стороны и поддержка со стороны серверного интерпретатора с другой.

Виды узлов

- Узлы документа
- Узлы-элементы
- Узлы-атрибуты
- Узлы пространств имен
- Узлы инструкций по обработке
- Узлы-комментарии
- Текстовые узлы

XPath

- XPath выражения похожи на пути к файлам

```
<xsl:template match="job/title">
```

- / - абсолютный путь

```
<xsl:template match="/employee/job/title">
```

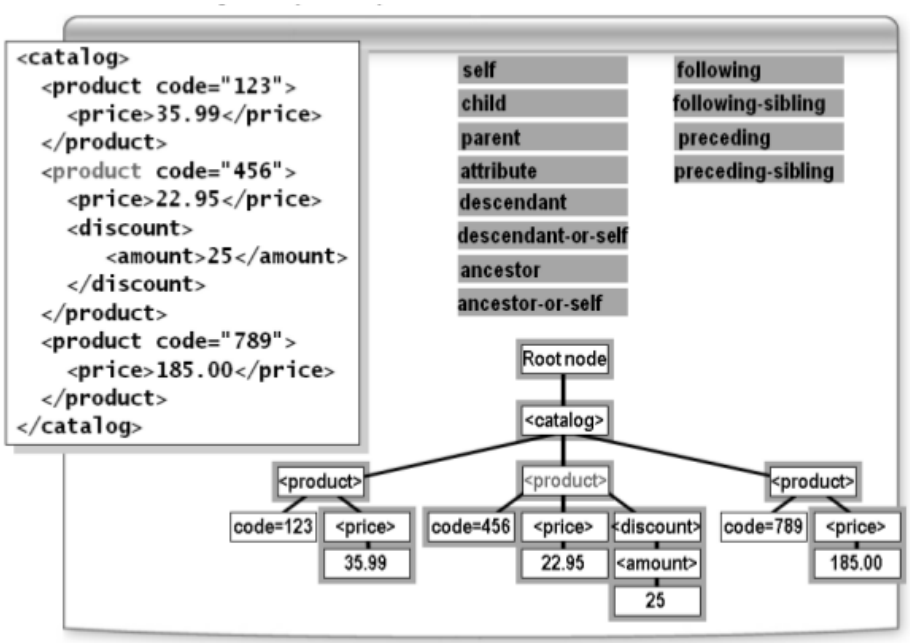
- // - поиск элементов на любой глубине

```
<xsl:template match="employee//name">
```

- * - любой элемент

```
<xsl:template match="*">
```

Оси



В Xpath используются выражения, указывающие на часть документа. Для этого используются: *оси поиска* и *фильтры*.

Поиск состоит из:

Ось поиска :: тест узла [предикат]

Оси

- ancestor:: — Возвращает множество предков.
- ancestor-or-self:: — Возвращает множество предков и текущий элемент.
- attribute:: — Возвращает множество атрибутов текущего элемента.
- child:: — Возвращает множество потомков на один уровень ниже.
- descendant:: — Возвращает полное множество потомков.
- descendant-or-self:: — Возвращает полное множество потомков и текущий элемент.
- following:: — Возвращает необработанное множество, ниже текущего элемента.
- following-sibling:: — Возвращает множество элементов на том же уровне, следующих за текущим.
- namespace:: — Возвращает множество имеющее пространство имён (то есть присутствует атрибут xmlns).

Оси

- parent:: — Возвращает предка на один уровень назад.
- preceding:: — Возвращает множество обработанных элементов исключая множество предков.
- preceding-sibling:: — Возвращает множество элементов на том же уровне, предшествующих текущему.
- self:: — Возвращает текущий элемент.

Существуют сокращения для некоторых осей, например:

- attribute:: — можно заменить на «@»
- child:: — часто просто опускают
- descendant:: — можно заменить на «//»
- parent:: — можно заменить на «..»
- self:: — можно заменить на «.»

Тест узла (по имени и по виду)

Примеры

Example	Result
child::book	Selects all book nodes that are children of the current node
attribute::lang	Selects the lang attribute of the current node
child::*	Selects all element children of the current node
attribute::*	Selects all attributes of the current node
child::text()	Selects all text node children of the current node
child::node()	Selects all children of the current node
descendant::book	Selects all book descendants of the current node
ancestor::book	Selects all book ancestors of the current node
ancestor-or-self::book	Selects all book ancestors of the current node - and the current as well if it is a book node
child::*/child::price	Selects all price grandchildren of the current node

Задача 3

1. Посмотреть содержимое таблицы lab2
2. Подсчитайте количество документов (documents) в первой записи
3. Подсчитайте количество книг (item) во второй записи
4. Получите список заголовков книг

Предикаты

- **Выражение может иметь предикаты**

```
axis::node-test[predicate]…
```

- **Фильтрация**

Положение узла	`product[last()]`
Содержание узла	`product[@code="123"]`
Присутствие узла	`product[@code]`

- **Несколько предикатов**

Вычисление слева - направо	`product[3][@discount]`

Предикаты

1. Первый из элементов CCC, вложенных в элементы BBB:

   ```
   /AAA/BBB/CCC[1]
   ```

2. Последний из элементов CCC, вложенных в элементы BBB:

   ```
   /AAA/BBB/CCC[last()]
   ```

3. Все элементы документа с двумя вложенными элементами CCC:

   ```
   //*[count(CCC)=2]
   ```

4. Все элементы документа с двумя любыми вложенными элементами:

   ```
   //*[count(*)=2]
   ```

Предикаты

- **Логические операции:**

```
A and B, A or B
```

- **Арифметические операции:**

```
A + B, A - B, A * B, A div B,
A mod B, - A
```

- **Операции сравнения:**

```
A = B, A != B, A > B, A < B,
A >= B, A <= B
```

Предикаты (Логические функции)

- or — логическое «или»
- and — логическое «и»
- = — логическое «равно»
- < (<) — логическое «меньше»
- > (>) — логическое «больше»
- <= (<=) — логическое «меньше либо равно»
- >= (>=) — логическое «больше либо равно»

boolean **boolean**(object) - Приводит объект к логическому типу;
boolean **true**() - Возвращает истину.
boolean **false**() - Возвращает ложь.
boolean **not**(boolean) - Отрицание, возвращает истину если аргумент ложь и наоборот.

Предикаты (Числовые функции)

- \+ — сложение
- − — вычитание
- * — умножение
- div — обычное деление (не деление нацело!)
- mod — остаток от деления

number **number**(object?) - Переводит объект в число.
number **sum**(node-set) - Вернёт сумму множества, каждый тег множества будет преобразован в строку и из него получено число.
number **floor**(number) - Возвращает наибольшее целое число, не большее, чем аргумент.
number **ceiling**(number) - Возвращает наименьшее целое число, не меньшее, чем аргумент.
number **round**(number) - Округляет число по математическим правилам.

Предикаты (Строковые функции)

- string **string**(object?) - Возвращает текстовое содержимое элемента. По сути возвращает объединенное множество текстовых элементов на один уровень ниже.
- string **concat**(string, string, string*) - Объединяет две или более строк
- number **string-length**(string?) - Возвращает длину строки.
- boolean **contains**(string, string) - Возвращает истину, если первая строка содержит вторую, иначе возвращает ложь.
- string **substring**(string, number, number?) - Возвращает строку вырезанную из строки начиная с указанного номера, и если указан второй номер — количество символов.
- string **substring-before**(string, string) - Если найдена вторая строка в первой, возвращает строку до первого вхождения второй строки.

Предикаты (Строковые функции)

- string **substring-after**(string, string) - Если найдена вторая строка в первой, возвращает строку после первого вхождения второй строки.
- boolean **starts-with**(string, string) - Возвращает истину если вторая строка входит в начало первой, иначе возвращает ложь.
- boolean **ends-with**(string, string) - Возвращает истину если вторая строка входит в конец первой, иначе возвращает ложь.
- string **normalize-space**(string?) - Убирает лишние и повторные пробелы, а также управляющие символы, заменяя их пробелами.
- string **translate**(string, string, string) - Заменяет символы первой строки, которые встречаются во второй строке, на соответствующие позиции символам из второй строки символы из третьей строки. translate(«bar», «abc», «ABC») вернет BAr.

Предикаты (Функции множеств)

* — обозначает любое имя или набор символов, @* — любой атрибут

$name — обращение к переменной, где name — имя переменной или параметра.

[] — дополнительные условия выборки

{} — если применяется внутри тега другого языка (например HTML), то XSLT процессор рассматривает содержимое фигурных скобок как XPath.

/ — определяет уровень дерева

node-set **node**() - Возвращает все узлы. Для этой функции часто используют заменитель '*', но в отличие от звездочки — node() возвращает и текстовые узлы.
string **text**() - Возвращает набор текстовых узлов

Предикаты (Функции множеств)

node-set **current**() - Возвращает множество из одного элемента, который является текущим. Если мы делаем обработку множества с условиями, то единственным способом дотянуться из этого условия до текущего элемента будет данная функция.

number **position**() - Возвращает позицию элемента в множестве. Корректно работает только в цикле <xsl:for-each/>

number **last**() - Возвращает номер последнего элемента в множестве. Корректно работает только в цикле <xsl:for-each/>

number **count**(node-set) - Возвращает количество элементов.

string **name**(node-set?) - Возвращает полное имя первого тега в множестве.

string **namespace-uri**(node-set?) - Возвращает ссылку на url определяющий пространство имён.

string **local-name**(node-set?) - Возвращает имя первого тега в множестве, без пространства имён.

node-set **id**(object) - Находит элемент с уникальным идентификатором

Предикаты (Функции множеств)

node-set **current**() - Возвращает множество из одного элемента, который является текущим. Если мы делаем обработку множества с условиями, то единственным способом дотянуться из этого условия до текущего элемента будет данная функция.

number **position**() - Возвращает позицию элемента в множестве. Корректно работает только в цикле <xsl:for-each/>

number **last**() - Возвращает номер последнего элемента в множестве. Корректно работает только в цикле <xsl:for-each/>

number **count**(node-set) - Возвращает количество элементов.

string **name**(node-set?) - Возвращает полное имя первого тега в множестве.

string **namespace-uri**(node-set?) - Возвращает ссылку на url определяющий пространство имён.

string **local-name**(node-set?) - Возвращает имя первого тега в множестве, без пространства имён.

node-set **id**(object) - Находит элемент с уникальным идентификатором

Предикаты (Системные функции)

node-set **document**(object, node-set?) -Возвращает документ, указанный в параметре object.
string **format-number**(number, string, string?) - Форматирует число согласно образцу, указанному во втором параметре, третий параметр указывает именованный формат числа, который должен быть учтён.
string **generate-id**(node-set?) - Возвращает строку, являющуюся уникальным идентификатором.
node-set **key**(string, object) - Возвращает множество с указанным ключом (аналогично функции id для идентификаторов).
string **unparsed-entity-uri**(string) - Возвращает непроанализированный URI, если такового нет, возвращает пустую строку.
boolean **element-available**(string) - Проверяет, доступен ли элемент или множество, указанное в параметре. Параметр рассматривается как XPath.

Предикаты (Системные функции)

boolean **function-available**(string) - Проверяет, доступна ли функция, указанная в параметре. Параметр рассматривается как XPath.
object **system-property**(string) - Параметры, возвращающие системные переменные, могут быть:

* xsl:version — возвращает версию XSLT процессора.
* xsl:vendor — возвращает производителя XSLT процессора.
* xsl:vendor-url — возвращает URL, идентифицирующий производителя.

Если используется неизвестный параметр, функция возвращает пустую строку.
boolean **lang**(string) - Возвращает истину, если у текущего тега имеется атрибут xml:lang, либо родитель тега имеет атрибут xml:lang и в нем указан совпадающий строке символ.

Задача 2

1. Получить название (title) первой книги.
2. Получить название (title) второй книги.
3. Получить название первой книги только на русском.
4. Подсчитать количество публикаций от ИрГУПС.

Подсказки

SELECT SUBSTRING(content,1,1000) FROM