Курс QA manual
Занятие 17

Основы XML

Использование XPATH

Логическая задачка

Человек, делающий это, в нем не нуждается; человек, покупающий это, сам им не пользуется, а человек пользующийся этим, об этом не знает.

Свекровь Клеопатры очень не любила ее и хотела ее убить. Но Клеопатра была не так глупа и была осторожной, и каждый раз она спасалась. Однажды, свекровь пригласила ее на ужин. Она разрезала ножом кусок мяса пополам, одну часть съела сама, а другую отдала Клео. После этого Клео умерла. Как удалось свекрови отравить Клеопатру?

Что такое XML?

HTML (от англ. HyperText Markup Language — «язык гипертектстовой разметки») — стандартный язык разметки документов.

XML - Extensible Markup Language, Расширяемый Язык Разметки. Возник в результате развития языка HTML.

Особенности XML

XML, в отличие от HTML, НЕ ИМЕЕТ ПРЕДОПРЕДЕЛЕННЫХ ТЭГОВ - точнее, каждый разработчик может создавать СВОИ СОБСТВЕННЫЕ XML-тэги - столько, сколько нужно. Количество таких тэгов практически неограничено. Таким образом, XML является *метаязыком*, позволяющим создавать другие языки разметки, такие как, например, HTML.

Особенности XML

XML служит для ОПИСАНИЯ СТРУКТУРЫ ДАННЫХ, главным образом, ИЕРАРХИЧЕСКИХ СТРУКТУР.

Особенности XML

XML, как средство описания структуры данных, обеспечивает ОБМЕН ДАННЫМИ между различными приложениями, выступая, таким образом, в качестве своеобразного "клея".

Объявление XML

Объявляется версия языка. Поскольку интерпретация содержимого документа, зависит от версии языка, то Спецификация предписывает начинать документ с объявления XML.

Кроме версии XML, объявление может также содержать информацию о кодировке документа
Пример:

<?xml version="1.1" encoding="UTF-8" ?>

или:

<?xml version="1.0" encoding="windows-1251"?>

Правила создания XML-документа

- В заголовке документа помещается объявление XML, в котором указывается язык разметки документа, номер его версии и дополнительная информация
- Каждый открывающий тэг, определяющий некоторую область данных в документе обязательно должен иметь своего закрывающего "напарника", т.е., в отличие от HTML, нельзя опускать закрывающие тэги
- В XML учитывается регистр символов
- Все значения атрибутов, используемых в определении тэгов, должны быть заключены в кавычки (“”)
- Вложенность тэгов в XML строго контролируется, поэтому необходимо следить за порядком следования открывающих и закрывающих тэгов
- Вся информация, располагающаяся между начальным и конечными тэгами, рассматривается в XML как данные и поэтому учитываются все символы форматирования (т.е. пробелы, переводы строк, табуляции не игнорируются, как в HTML)

Наш первый XML файл

Классическим примером использования языка XML является описание электронного письма.

```
<?xml version="1.0"?>
<MESSAGE>
    <TO>STUDENT</TO>
    <FROM>AUTHOR</FROM>
    <SUBJECT>Introduction to XML</SUBJECT>
    <BODY>Welcome to XML!</BODY>
</MESSAGE>
```

Наш первый XML файл

- Тег — конструкция разметки, которая содержит имя элемента.
- Начальный тег: <element1>
- Конечный тег: </element1>
- Тег пустого элемента: <empty_element1 />
- В элементе атрибуты могут использоваться только в начальном теге и теге пустого элемента. Атрибут - это пара "название" = "значение"
- Комментариями является любая область данных, заключенная между последовательностями символов <!-- и -->

Наш второй XML файл

```
<recipe name="хлеб" preptime="5" cooktime="180">
  <title>Простой хлеб</title>
  <composition>
    <ingredient amount="3" unit="стакан">Мука</ingredient>
    <ingredient amount="0.25" unit="грамм">Дрожжи</ingredient>
    <ingredient amount="1.5" unit="стакан">Тёплая вода</ingredient>
    <ingredient amount="1" unit="чайная ложка">Соль</ingredient>
  </composition>
  <instructions>
    <step>Смешать все ингредиенты и тщательно замесить.</step>
    <step>Закрыть тканью и оставить на один час в тёплом помещении.</step>
    <!-- <step>Почитать вчерашнюю газету.</step> - это сомнительный шаг... -->
    <step>Замесить ещё раз, положить на противень и поставить в духовку.</step>
  </instructions>
</recipe>
```

Исключения в XML

Символ	Замена
<	<
>	>
&	&
'	'
"	"

Языки запросов

- XPath — cXPath (XML Path Language) — язык запросов к элементам XML-документа. XPath призван реализовать навигацию по DOM в XML. Выражения XPath используются в языке XQuery.
- Xquery — язык программирования, ориентированный на работу с документами.
- DOM (от англ. Document Object Model — «объектная модель документа») — это не зависящий от платформы и языка программный интерфейс, позволяющий программам и скриптам получить доступ к содержимому HTML, XHTML и XML-документов, а также изменять содержимое, структуру и оформление таких документов.

XPATH – тестовая XML

```
<html>
 <body>
   <div>Первый слой
     <span>блок текста в первом слое</span>
   </div>
   <div>Второй слой</div>
   <div>Третий слой
     <span class="text">первый блок в третьем слое</span>
     <span class="text">второй блок в третьем слое</span>
     <span>третий блок в третьем слое</span>
   </div>
   <img />
 </body>
</html>
```

XPath-путь **/html/body/*/span[@class]**
/child::html/child::body/child::*/child::span[attribute::class]

XPATH

- Путь делится на шаги адресации, которые разделяются символом «косая черта» / . Каждый шаг адресации состоит из трех частей:
  - ось (в данном примере child::), это обязательная часть;
  - условие проверки узлов (в данном примере это имена элементов документа html, body, span, а символ * означает элемент с любым именем), также обязательная часть;
  - предикат (в данном примере attribute::class), необязательная часть, заключаемая в квадратные скобки, в которой могут содержаться оси, условия проверки, функции, операторы (+, -, <, > и пр.).

XPATH – тестовая XML2

```
<?xml version="1.0" encoding="UTF-8"?>
<bookstore>
	<folder>
		<number>25</number>
		<book>
			Hello
			<title lang="eng">Harry Potter</title>
			<price>29.99</price>
			<price lang="eng"></price>
		</book>
	</folder>
	<book>
		<title lan="eng">Learning XML</title>
		<price>39.95</price>
	</book>
	<book>
		<title lang="eng">Learning java</title>
		<price>45.30</price>
	</book>
</bookstore>
```

XPATH - Selecting Nodes

Expression	Description
nodename	Selects all nodes with the name "*nodename*"
/	Selects from the root node
//	Selects nodes in the document from the current node that match the selection no matter where they are
.	Selects the current node
..	Selects the parent of the current node
@	Selects attributes

XPATH - Selecting Nodes

Path Expression	Result
bookstore	Selects all nodes with the name "bookstore"
/bookstore	Selects the root element bookstore**Note:** If the path starts with a slash (/) it always represents an absolute path to an element!
bookstore/book	Selects all book elements that are children of bookstore
//book	Selects all book elements no matter where they are in the document
bookstore//book	Selects all book elements that are descendant of the bookstore element, no matter where they are under the bookstore element
//@lang	Selects all attributes that are named lang

XPATH - Predicates

Predicates are used to find a specific node or a node that contains a specific value.

Predicates are always embedded in square brackets.

Path Expression	Result
/bookstore/book[1]	Selects the first book element that is the child of the bookstore element.**Note:** In IE 5,6,7,8,9 first node is[0], but according to W3C, it is [1]. To solve this problem in IE, set the SelectionLanguage to XPath: *In JavaScript: xml*.setProperty("SelectionLanguage","XPath");
/bookstore/book[last()]	Selects the last book element that is the child of the bookstore element
/bookstore/book[last()-1]	Selects the last but one book element that is the child of the bookstore element
/bookstore/book[position()<2]	Selects the first two book elements that are children of the bookstore element
//title[@lang]	Selects all the title elements that have an attribute named lang
//title[@lang='eng']	Selects all the title elements that have an attribute named lang with a value of 'eng'
/bookstore/book[price>35.00]	Selects all the book elements of the bookstore element that have a price element with a value greater than 35.00
/bookstore/book[price>35.00]/title	Selects all the title elements of the book elements of the bookstore element that have a price element with a value greater than 35.00

XPATH - Selecting Unknown Nodes

Wildcard	Description
*	Matches any element node
@*	Matches any attribute node
node()	Matches any node of any kind

Path Expression	Result
/bookstore/*	Selects all the child nodes of the bookstore element
//*	Selects all elements in the document
//title[@*]	Selects all title elements which have any attribute

XPATH - Selecting Several Paths

By using the | operator in an XPath expression you can select several paths.

Path Expression	Result
//book/title | //book/price	Selects all the title AND price elements of all book elements
//title | //price	Selects all the title AND price elements in the document
/bookstore/book/title | //price	Selects all the title elements of the book element of the bookstore element AND all the price elements in the document

XPATH - Axes

AxisName	Result
ancestor	Selects all ancestors (parent, grandparent, etc.) of the current node
ancestor-or-self	Selects all ancestors (parent, grandparent, etc.) of the current node and the current node itself
attribute	Selects all attributes of the current node
child	Selects all children of the current node
descendant	Selects all descendants (children, grandchildren, etc.) of the current node
descendant-or-self	Selects all descendants (children, grandchildren, etc.) of the current node and the current node itself
following	Selects everything in the document after the closing tag of the current node
following-sibling	Selects all siblings after the current node
namespace	Selects all namespace nodes of the current node
Parent	Selects the parent of the current node
Preceding	Selects all nodes that appear before the current node in the document, except ancestors, attribute nodes and namespace nodes
preceding-sibling	Selects all siblings before the current node
self	Selects the current node

XPATH - Axes

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

XPATH – тестовая XML3

```
<?xml version="1.0" encoding="UTF-8"?>
<first_layer>
        Hello I'm the first layer
        <second_layer>
                Hello I'm the second layer
                <third_layer>
                        <forth_layer>
                                Hello I'm the forth layer
                        </forth_layer>
                </third_layer>
                <third_layer_two>
                        Hello I'm the third layer, but I have a bro above
                </third_layer_two>
                <third_layer_three>
                        Hello I'm the third layer, but I have two bros above
                        <forth_layer_three>
                                Hello I'm the forth layer of the third layer with two bros, I also have one bro above
                        </forth_layer_three>
                </third_layer_three>
        </second_layer>
</first_layer>
```

XPATH - Axes

Example	Result
/first_layer/second_layer/third_layer/ancestor::second_layer	Selects second_layer as an ancestor of the third_layer node
/first_layer/second_layer/third_layer/ancestor-or-self::third_layer	Selects third_layer as an ancestor of itself
/first_layer/second_layer/descendant::forth_layer	Selects forth_layer as a descendant of the second_layer
/first_layer/second_layer/descendant-or-self::second_layer	Selects second_layer as a descendant of itself
/first_layer/second_layer/third_layer/following::third_layer_two	Selects only third_layer_two in the document after the closing tag of the third_layer
/first_layer/second_layer/third_layer/following::*	Selects everything in the document after the closing tag of the third_layer
/first_layer/second_layer/third_layer/following-sibling::third_layer_three	Selects third_layer_three as a sibling after the third_layer

XPATH - Axes

Example	Result
/first_layer/second_layer/third_layer/parent::*	Selects the parent of the third_layer
/first_layer/second_layer/third_layer_three/preceding::*	Selects all nodes that appear before the third_layer_three in the document, except ancestors, attribute nodes and namespace nodes
/first_layer/second_layer/third_layer_three/preceding-sibling::third_layer	Selects third_layer as a sibling before the third_layer_three
/first_layer/second_layer/third_layer_three/self::*	Selects the third_layer_three
/first_layer/second_layer/text()	Selects text of second_layer
/first_layer/second_layer/node()	Selects all children of the second_layer

XPATH

- 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 «.»

XPATH - Operators

Operator	Description	Example
|	Computes two node-sets	//book | //cd
+	Addition	6 + 4
-	Subtraction	6 - 4
*	Multiplication	6 * 4
div	Division	8 div 4
=	Equal	price=9.80
!=	Not equal	price!=9.80
<	Less than	price<9.80
<=	Less than or equal to	price<=9.80
>	Greater than	price>9.80
>=	Greater than or equal to	price>=9.80
or	or	price=9.80 or price=9.70
and	and	price>9.00 and price<9.90
mod	Modulus (division remainder)	5 mod 2

Практика

Скачиваем XML файл и начинаем практиковаться!

- Select all the titles
- Select the title of the first book
- Select all the prices
- Select price nodes with price>35
- Select title nodes with price>35

Домашнее задание

- Дополнить HomeWork.xml данными со своей команды
- Выбрать все комментарии
- Выбрать имя ITShool
- Выбрать всех Боссов
- Выбрать ФИО и Должность всех доп.работников
- Выбрать ФИО только тех студентов кому больше либо равно 23 года.
- Выбрать ноду-предка (в данном случае только отца) для ноды <boss>
  (ancestor::)
- Выбрать дочерние ноды для ноды студенты с id=3 включая ее саму
  (descendant-or-self::)
- Выбрать ноду следующую за <kontaktnie_dannie> студента с id=2
  (following::)
- Выбрать ноду находящуюся на одном уровне с хобби для студента с id=1
  (following-sibling::)
- Выбрать исключительно отцовскую ноду для ноды студенты
  (parent::)
- Выбрать предыдущую ноду на том же уровне,что и должность, для доставщика пиццы
  (preceding-sibling::)

Вопросы?