Обьединяем мощь XPath и библиотек тегов JSP

Простой доступ к нужным данным посредством XPath из JSP.

XPath-JSP Tutorial

Стэнли Сантьяго

В этой статье мы исследуем пользовательскую библиотеку тегов (ПБТ) XPath для JSP, рассмотрим набор тегов обеспечивающий простое управление и универсальную подмену значений аттрибутов что в совместном применении уменьшает сложность исходного кода и увеличивает функциональность. Использование XPath библиотеки дополнительно обеспечивает ясное разделение данных от вариантов их представления. *(2500 слов)*

Java Server Pages и XML являются естественными партнерами в создании веб приложений основой для которых выступают гетерогенные источники данных. DOM API для XML представляет собой универсальную возможность представления этих разрозненых данных. В свою очередь XPath обеспечивает стандартный способ и простой синтаксис для доступа к DOM. JSP страницы выполняются как Java серлеты обычно в контексте веб сервера, в котором генерят сессию и собственно страницы с данными для ответа на запрос. Если обьеденить все эти особенности, плюс использовать XPath тогда мы получим что-то на подобие стандартной и универсальной системы для представления данных и доступа к ним.

Библиотека тегов XPath представляет инфраструктуру (framework) в рамках которой динамическое содержание представляется в виде DOM документа с возможностью манипулирования и включением в JSP страницы для:

- Упрощения кода в JSP и сделать его вообще без Java

- Упрощения шаблонов или мест подмена внутри JSP
- Убрать сложность к доступу к динамическим данным, при этом делая его более мощным (доступ)

Библиотека тегов XPath обьеденненная с JSP способствует ясному разделению труда между автором страниц (их содержания) и программистом.

Данная статья предполагает, что читатель знаком на средне-продвинутом уровне с JSP, XML, и DOM, если же нет, то вы можете начать с изучения материала указанного в конце статьи в раздеде "Ресурсы" для лучшего понимания .

Замечание:

Библиотека тегов XPath находится все еще в стадии разработки и следовательно не доступна как самостоятельная библиотека. Те не менее вы можете загрузить WAR файл XPath-JSP Тест- приложения забрать из него нужные классы для создания своей самостоятельной библиотеки тегов.

1. Синтаксис XPath

XPath, рекомендация W3C от Ноября 1999, описывают простую нотацию для указания и выборки отдельных частей XML документа. XML документ — это дерево элементов в котором путь от корневого узла до любого вложенного уникальный. Та вот XPath как раз и определяет этот путь.

Давайте рассмотрим пример XML документа и несколько выражений XPath используемых для выборки отдельных его частей.

Данный документ представляет пользователя (user) который имеет свой код (userid) и пароль (password). Также пользователь может иметь несколько ролей, в нашем случае их две: `Администратор домена (Domain Administrator)` и `Администратор справочной (Help Desk Administrator)`:

```
<user>
  <userid>такой-то</userid>
  <password>какой-то</password>
  <roles>
  <role id="admin">
   Администратор домена</role>
  <role id="helpdesk">
   Администратор справочной</role>
```

```
  </roles>
 </user>
```

В своей основе синтаксис XPath очень похож на адресацию файлов в файловых системах. В том случае, если в путь начинается с /, это говорит о том, что это путь к нужному элементу — абсолютный:

```
/user/userid/text() извлекает 'такой-то'

 /user/roles/role извлекает все элеметы 'role'

//role извлекает все элеметы 'role' [ аналогично предыдущему примеру]

 //role[@id='admin']/text() извлекает 'Администратор домена'

 /user/roles/role[1] извлекает первый элемент role
                          т.е. 'Администратор домена'

 /user/roles/role[last()] извлекает последний элемент role
                          т.е. 'Администратор справочной'
```

Приведенные примеры иллюстрируют простые примеры XPath. Для более детального ознакомления с синтаксисом XPath — обратитесь к учебному курсу по XPath в разделе ресурсов.

2. Архитектура веб приложения использующего библиотеку тегов XPath

Увидев как можно использовать XPath выражения на XML документе, давайте обсудим архитектуру веб приложения использующего библиотеку тегов XPath изображенную на рис.1

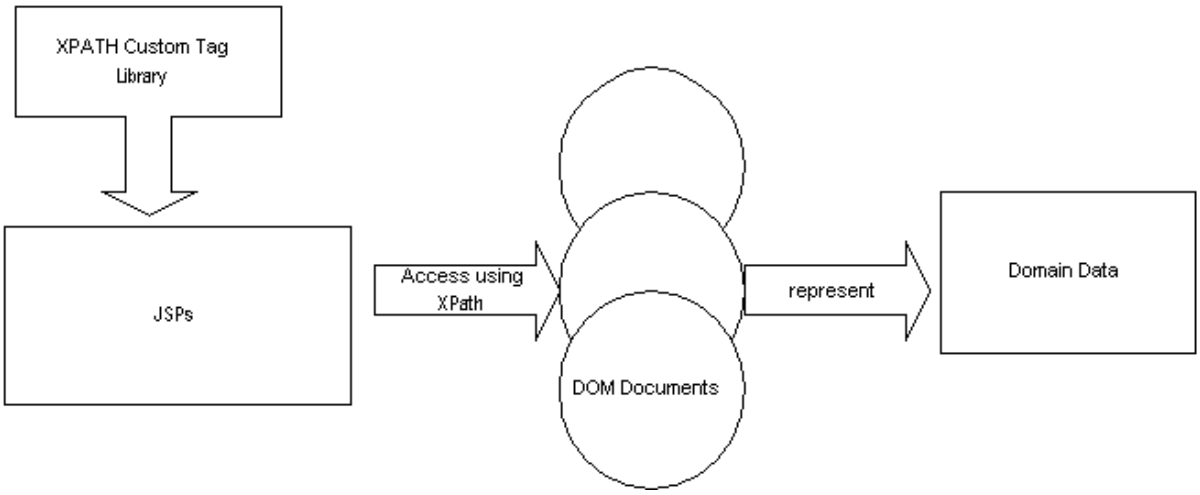

Рисунок 1. Веб приложение использующее библиотеку тегов XPath.

В этом приближении, презентационный уровень (JSP) обращается к данным используя XPath выражения. Данные представлены в виде DOM экземпляров.

Авторы страниц используют ПБТ XPath для вставки XPath выражений в JSP страницы. DOM документ принадлежит разработчикам и они ответственны за его наполнение и состояние. Также, авторам страниц можно дать и DTD DOM документа для того, чтоб они имели представление о структуре документа, что будет не лишним при написании выражений XPath.

Запомните, что выражения XPath служат лишь только для доступа к данным и не могут быть использованы для их (данных) модификации в DOM экземпляре документа.

3. Примеры выражений XPath в JSP.

Сейчас мы создадим пример для иллюстрации различных опций ПБТ XPath. В этом примере мы предположим, что работаем с клиентскими заказами. Пример XML документа `Order(Заказ)` выглядит так:

`Order.xml`

```
<ORDER>                                 <!-- ЗАКАЗ -->
   <SHIPTO>                             <!-- ДОСТАВИТЬ К -->
```

```
      <NAME>ALICE SMITH</NAME>           <!-- ИМЯ -->
      <STREET>123 MAPLE STREET</STREET>  <!-- УЛИЦА -->
      <CITY>MILL VALLEY</CITY>           <!-- ГОРОД -->
      <STATE>CA</STATE>                  <!-- ШТАТ -->
      <ZIP>90952</ZIP>                   <!-- ПОЧТОВЫЙ КОД -->
   </SHIPTO>

   <DATE>12-31-2000</DATE>               <!-- ДАТА -->

   <!-- Теперь идут позиции заказа -->

   <ITEM>                                          <!-- ПОЗИЦИЯ -->
      <TITLE>Twelve Songs of Christmas</TITLE>     <!-- НАИМЕНОВАНИЕ -->
      <ARTIST>JIM REEVES</ARTIST>                  <!-- АРТИСТ -->
      <PRICE>15.95</PRICE>                         <!-- ЦЕНА -->
   </ITEM>
   <ITEM>
      <TITLE>First Piano Concerto</TITLE>
      <ARTIST>>Janos</ARTIST>
      <PRICE>12.95</PRICE>
   </ITEM>

<!-- Остальные позиции здесь -->
 ...  ... ...
</ORDER>
```

XML документ `ORDER(Заказ)` состоит из `SHIPTO(Доставить к)`, `DATE(Дата)`, и нескольких `ITEM(Позиция)` элементов. Приложение держит в памяти DOM представление этого документа. Следующие примеры иллюстрирую применение выражений XPath в JSP, которые извлекают нужные данные из этого DOM.

4. Пример 1: Простой доступ и подстановка

В нашем первом примере, JSP использует XPath для доступа к DOM экземпляру `ORDER`, получения и подстановки в JSP:

viewOrder.jsp

```
<%@ page errorPage="error.jsp" %>
```

```
<%@ taglib uri="/src/tags/taglib.tld" prefix="mytag" %>

<html>

  <head>
    <title>Просмотр заказов </title>
  </head>

   <body bgcolor="white" link="#666699" vlink="#666699">

    <B> Дата заказа: </B> <mytag:getvalue select="/ORDER/DATE" /> <BR>

 <HR>
<!-- тег ifdef проверяет истинность Xpath выражения -->
<mytag:ifdef select="/ORDER/SHIPTO" >

    <B> Доставить к: </B> <BR/>
    <mytag:getvalue select="/ORDER/SHIPTO/NAME" /> <BR>
    <mytag:getvalue select="/ORDER/SHIPTO/STREET" /> <BR>
    <mytag:getvalue select="/ORDER/SHIPTO/CITY" /> <BR>
    <mytag:getvalue select="/ORDER/SHIPTO/STATE" /> <BR>
    <mytag:getvalue select="/ORDER/SHIPTO/ZIP" /> <BR>

</mytag:ifdef>

 <HR>

    <!-- Тег – iterate – производит итерации
     по повторяющимся элементам "ITEM"
     Тег – getvalue – подставляет результат
     выражения XPath для данной итерации       -->

    <mytag:iterate select="/ORDER/ITEM" >

<p>
   <B> Артист: </B> <mytag:getvalue select="./ARTIST" />
   <B> Наименование : </B> <mytag:getvalue select="./TITLE" />
   <B> Цена : </B> <mytag:getvalue select="./PRICE" />

     <BR>
```

```
    </mytag:iterate>

    </body>

    </html>
```

Выше приведенный код, использует 3 тега из ПБТ XPath : `getvalue`, `ifdef`, и `iterate`. Все эти теги имеют атрибут `select` в котором указывается выражение XPath .

Тег `getvalue` применяет указанное в атрибуте `select` выражение XPath на DOM представление документа Order и подставляет результат применения в результирующий документ.

Тег логического выражения `ifdef` выполняет свое тело лишь в том случае, если выражение XPath в его атрибуте `select` является истинным.

`iterate`, — тег итерации, выполняет свое тело для каждой итерации. В нашем случае, атрибут `select` указывает на узел по которому необходимо итерировать. Заметьте, что тег `getvalue` , при использовании его внутри тега `iterate`, указывает на относительный путь XPath выражения в своем атрибуте `select`.

Приведенный выше пример иллюстрирует реализацию трех возможностей используя выражения XPtah в JSP, а именно простой доступ к данным, подстановку, контроль над ходом выполнения. Следующий пример показывает подстановку значений аттрибутов.

5. Пример 2: Подстановка значений аттрибутов

В нашем втором примере мы рассмотрим как с помощью выражений XPath в JSP получить доступ к данным тега `SHIPTO` из DOM экземпляра которые в последствии будут изменяться в HTML форме. Обратите внимание на использование префикса `template:` при вычислении значения аттрибута value:

editShipTo.jsp

```
<%@ taglib uri="/src/tags/taglib.tld" prefix="mytag" %>
```

```
   <html>
    .....
    .....

      <form action="/editshipto.do" method="POST" >

      SHIP TO: <BR/>

     <!-- Подстановка(вычисление) значения аттрибута value
     производится посредством префикса 'template:'
     и заключением в тег XPathTemplate всего
     содержимого блока вычисления -->
     <mytag:XPathTemplate>

 NAME: <input type="text" name="name"
  template:value="/ORDER/SHIPTO/NAME"/> <BR/>

 STREET: <input type="text" name="street"
  template:value="/ORDER/SHIPTO/STREET"/> <BR/>

 CITY : <input type="text" name="city"
  template:value="/ORDER/SHIPTO/CITY"/> <BR/>

 STATE: <input type="text" name="state"
  template:value="/ORDER/SHIPTO/STATE"/> <BR/>

 ZIP: <input type="text" name="zip"
  template:value="/ORDER/SHIPTO/ZIP"/>

     </mytag:XPathTemplate>

</form>
  .....
  .....
    </html>
```

Приведенный выше код иллюстрирует использование тег-контейнера `XPathTemplate`, который позволяет выражениям XPath внедряться непосредственно в текст документа и вычислять (подсталять) результаты XPath выражений. Для этого, как вы уже заметили, используется префикс `template:` добавляемый к имени

вычисляемого атрибута.

Весь текст заключенный в тег-контейнер `XPathTemplate` просматривается на наличие элементов с именами атрибутов с префиксом `template:`. Если такой элемент находится, тогда XPath выражение применяется на DOM экземпляр документа и перезаписывает атрибут value уже без `template:` префикса.

В приведенном выше примере мы рассмотрели как используются выражения XPath в значениях атрибутов и как работает тег-контейнер `XPathTemplate`, подставляющий вычисляемые значения в документ. Далее, мы рассмотрим другие теги доступные в данной версии ПБТ XPath.

6. Библиотега пользовательских тегов XPath

Теперь мы рассмотрим различные теги и их атрибуты в БПТ XPath. К каждому тегу вы найдете его имя, описание и детали атрибутов.

6.1. Тег: XPathTemplate

Описание:

- Тег-контейнер/супер-тег позволяющий использовать выражения XPath в атрибутах документа
- Тело этого тега должно быть в коде HTML или well-formed (правильно-сформированным) XML
- Для подстановки значения атрибута необходимо перед его именем посавить префикс `template:` Т.е. конструкция вида `template:имя_атрибута="XPath_выражение"` после обработки заменится на `имя_атрибута="значение_XPath_выражения"`
- Для подстановки значений элементов используйте теги `iterate` или `getvalue` описание которых приведено ниже

Атрибуты:

- Имя атрибута: content (тело,контент,содержание)
- Значение атрибута: html (по-умолчанию) или xml

6.2. Тег: iterate

Описание:

- Позволяет реализовать итерации по выражению XPath, которое возвращает множество нодов, по которому, собственно и производятся итерации.
- Используется в сочетании с тегом `getvalue`

Атрибуты:

- Имя атрибута: select
- Значение атрибута: выражение XPath

6.3. Тег: getvalue

Описание:

- Подставляет результата XPath выражения
- Реализация этого тега проверяет заключен ли этот тег в теле тега `iterate` и подставляет соответствующее значение XPath выражения

Атрибуты:

- Имя атрибута: select
- Значение атрибута: выражение XPath

6.4. Тег: ifdef

Описание:

- Проверяет истинность выражения XPath примененное на DOM документа
- В случае если true (верно) тело этого тега обрабатывается, иначе — нет.

Атрибуты:

- Имя атрибута: select
- Значение атрибута: выражение XPath

6.5. Тег: ifnodef

Описание:

- Проверяет НЕ истинность выражения XPath примененное на DOM документа
- В случае если true (верно) тело этого тега обрабатывается, иначе — нет.

Атрибуты:

- Имя атрибута: select
- Значение атрибута: выражение XPath

6.6. Тег: gencontent

Описание:

- Создает экземпляр DOM документа в пределах сессии из XML находящегося в теле этого тега
- В случае если атрибут `debug` установлен в `on`, этот тег сериализует DOM экземпляр документа в стандартный поток вывода

Атрибуты:

- Имя атрибута: debug
- Значение атрибута: on или off

7. Представление данных в виде DOM документов

Предыдущие разделы иллюстрировали как можно посредством XPath выражений обеспечивать простой доступ к данным из DOM экземпляра документа. Теперь же давайте обсудим 2 основные технологии создания DOM экземпляра документа для предствления данных.

Данные могут быть представлены в виде DOM документ двумя путями:

1. Классами данных, например `Order.java`, может возвращать DOM документ отражающий его (заказа) свойста, элементы и т.д. путем вызова метода `toDOM()`
2. Набором обьектов JavaBean преобразованных к DOM экземплярам докуметов во время выполнения

Из двух, второй вариант более элегантен, но превносит дополнительные потери в производительности, поскольку преобразование из JavaBean в DOM документ это дополнительные накладные расходы. Java библиотеки JOX предоставляют возможность представления JavaBeans в XML и обратно.

8. Отображение выражений XPath на DOM экземпляры.

Итак как же выражения XPath отображаются на DOM экземпляр документа ?

После создания DOM представления данных необходимых для jsp, DOM экземпляр помещается в рамки приложения (application scope), запроса (request scope), сессии (session scope), или страницы (page scope). `name` или `key` свойства экземпляра DOM документа точно такие же как и у элемента document.

Например DOM представление файла `Order.xml` будет сохранено в одной из рамок (scope) с именем `Order`, поскольку это же имя имеет элемент document в `Order.xml`.

Во время выполнения, из XPath выражения выделяется root(корневой) элемент. Например, если XPath выражение это `/ORDER/SHIPTO/NAME`, тогда root элемент будет `ORDER`. Тег-обработчик просматривает DOM представление документа в поисках `ORDER` в рамках (scope) страницы, сессии, запроса, и приложения.

После того как подходящее DOM представление найдено, на него применяется выражение XPath и затем, результат выполнения подставляется в место вызова.

Эта семантика скрыта от автора JSP страниц, поскольку спрятана в реализации ПБТ.

9. Тестовое XPath-JSP Приложение

Я включил тестовое XPath-JSP приложение, так что вы сможете попрактиковаться и проверить как вы усвоили то что обсуждалось в предыдущих параграфах.

В этом приложениии показано как создать обьект DOM документа используя тег `gencontent` в JSP. В нем также показано как можно получить доступ к различным частям DOM документа посредством ранее расмотренных XPath выражений из JSP страницы.

Вы также увидете как обрабытываются и отображаются выражения XPath на DOM документа.

9.1. Инсталяция XPath-JSP Тестового Приложения

Перед загрузкой и инсталяцией убедитесь, что вы уже имеете следующие программные пакеты:

- **Java Development Kit**
  Вам нужно загрузить и проинсталировать Java 2 (версии 1.2 или старше) релизация

JDK для вашей операционной системы. Посмотрите на раздел ресурсов по информации о загрузке.

- **Контейнер Сервлетов**
  Вам нужно загрузить и проинсталировать контейнер сервлетов совместимый с со спецификацией на сервлеты (Servlet API Specification) (версии 2.2 или старше) и спецификацией на JSP (версии 1.1 или старше). Один из популярных продуктов подходящий для наших требований — это Apache's Tomcat (версии 3.1 или старше). ПРИМЕЧАНИЕ: это Тестовое Приложение проверялось *только* на Apache Tomcat версии 3.1. **от переводчика.:** данный тестовый пример работал без проблем и с Resin-1.2.3

После успешной инсталяции перечисленных выше продуктов, загрузите XPath-JSP Тестовое Приложение, которое содержит архив веб приложения (Web application archive (WAR)). Оно должно работать на под любым сервлет контейнером совместимым с сервлет спецификацией 2.2 или старшей.

Предупреждение:
Данный продукт содержит програмное обеспечение разработанное Apache Software Foundation (http://www.apache.org/).

Предупреждение:
Данный продукт содержит програмное обеспечение разработанное OpenXML Inc. (http://www.openxml.org/).

При развертке файла `xpath-jsp.war` структура директорий должна быть следующей:

```
xpath-jsp
|   error.jsp (страница-обработчик ошибок JSP)
|   editShipTo.jsp (пример использования XPATH выражений в JSP)
|   viewOrder.jsp (пример использования XPATH выражений в JSP)
|
+---META-INF
|       MANIFEST.MF
|
+---test-DOMs
|       dom.jsp (тестовая jsp страница, которую вы можете
|                использовать для создания
```

```
|                DOM документа в рамках сессии)
|
\---WEB-INF
    |   taglib.tld
    |   web.xml
    |
    +---classes (Классы пользовательской
    |            библиотеки тегов XPath- JSP)
    |   +---tags
    |   |       ex1.html
    |   |       GenContentTag.class
    |   |       IFDef.class
    |   |       IFDefTag.class
    |   |       IFNoDefTag.class
    |   |       IterateTag.class
    |   |       RefTag.class
    |   |       TemplateTag.class
    |   |       test.xml
    |   |
    |   \---util
    |           Util.class
    |           XPathQuery.class
    |
    \---lib
            openxml-1.2.jar (OpenXML HTML парсер)
            xalan.jar (Apache XSLT процессор)
            xerces.jar (Apache XML парсер)
```

9.2. Установка CLASSPATH

WAR файл XPath-JSP Тестового Приложения содержит Apache XML парсер `xerces.jar`. Чтобы все заработало необходимо, чтоб путь к файлу `xerces.jar` стоял первым из всех путей к XML парсерам в переменной окружения `CLASSPATH`.

В случае использования Tomcat 3.1, убедитесь, что `xerces.jar` стоит перед `xml.jar` XML парсером определенным по-умолчанию.

9.3. Запуск XPath-JSP Тестового Приложения

Для запуска тестового приложения следуйте следующим путем:

1. Зайдите в директорию `xpath-jsp/test-DOMs` и откройте файл `dom.jsp` .
2. Содержимое этого файла — это XML данные заключенные в тег `<gencontent/>`. Когда запускается эта страница то создается DOM экземпляр из тела этого тега. DOM экземпляр документа сохраняется в рамках сессии. В качестве "ключа"("key") или "имени"("name") используемого для сохранения экземпляра используется элемент document содержимого XML. В нашем случае его значение будет `Order`.
3. Заходим в директорию `xpath-jsp` и открываем `viewOrder.jsp` и `editShipTo.jsp` файлы.
4. Заметьте, что выражения применяются на DOM созданный из `dom.jsp`.
5. Для теста, откройте `dom.jsp` из браузера. Затем откройте `viewOrder.jsp` и `editShipTo.jsp` файлы.

Предупреждение:

Код для данного приложения все еще в стадии разработки, поэтому возможны сбои.

Ну вот и все,о чем хотелось упомянуть.

10. Некоторые соображения

Не существует легких путей. Понимая это, давайте рассмотрим "за" и "против" обьеденения возможностей XPath выражений в JSP.

ПБТ XPath-JSP представляет прекрасное решение для упрощения кода на уровне представления (presentation tier) и осводождая его от повторяющегося Java кода. Также позволяет разделять данных от вариантов их представления предлагая для этого небольшой набор тегов, которые давют возможность быстро и просто выбрать необходимую информацию из DOM документа. Хотя использование предполагает знание синтаксиса XPath, этот синтаксис не так уж сложен. DOM документы полностью обрабатываются "за сценой", чтопозволяет упростить код представления (presentation layer).

Производительность потенциально можеть быть ниже из-за накладных расходов при преобразовании данных в DOM и доступе посреством XPath. Тем не менее невозвожно предугадать точно насколько понизиться производительность так как она зависит от

многих обстоятельств и может быть практически мало заметной в реальных ситуациях.

За:

- Стандарт в доступе и подстановке благодаря использованию синтаксиса XPath
- Ясное разделение ролей разработчика и автора страниц.
- Автор страниц: эксперт в Пользовательском интерфейсе, использует XPath выражения.
- Разработчик: эксперт в серверной логике, создает и использует пользоваьельские библиотеки тегов и DOM документы
- XPath предлагает простой но мощный синтаксис прохождение по дереву поиска

Против:

- Представление данных в виде DOM документов — некоторая потеря в производительности .

11. Заключение

После прочтения данной статьи вы должны хоррошо понимать принципы построения страниц с использованием библиотеки XPath а также пределы ее использования.

Спасибо моему коллеге Andy Hakim за его комментарии и отклики.

12. Об авторе

Stanley Santiago старший программист в iPlanet Ecommerce Solutions, подразделение Sun/Netscape . Имеет за плечами 8-ми летний опыт работы с ООП проектами и интернет проектами на Java и XML.

13. От переводчика

Хотя представленная автором библиотека XPath предлагает небольшой набор тегов (по сравнению с предлагаемыми в спецификации по XPath W3C), однако для небольшого проекта этого будет вполне достаточно для манипулированием содержания XML фалов из JSP. Юрий Бакуменко

14. Ресурсы

Ресурсы JavaWorld

- XPath-JSP Тестовое приложение WAR : http://www.javaworld.com/javaworld/jw-01-2001/xpath/jw-0126-xpath.zip
- "XML document processing in Java using XPath and XSLT" by Andrй Tost (*JavaWorld,* September 2000) обьясняет как XPath и XSLT могут значительно упростить ваш Java код при обработке XML документов: http://www.javaworld.com/javaworld/jw-09-2000/jw-0908-xpath.html
- "XML JavaBeans, Part 1" by XML guru Mark Johnson (*JavaWorld,* February 1999) касается преобразования JavaBeans в XML и DOM: http://www.javaworld.com/javaworld/ jw-02-1999/jw-02-beans.html
- Вы найдете множество различной инфомации/статей касающейся XML в разделе **Java and XML** в нашем Topical Index: http://www.javaworld.com/javaworld/topicalindex/jw-ti-javaxml.html
- Для большего кол-ва статей о применении Java на стороне сервера, не пропустите раздел **Server-Side Java** в нашем Topical Index: http://www.javaworld.com/javaworld/topicalindex/jw-ti-ssj.html
- **Java in Enterprise** группа обуждения, модерируемая Qusay Mahmoud: http://www.itworld.com/jump/jw-0126-xpath/ forums.itworld.com/webx?14@@.ee6b80a
- Обмен отытом разработки XML в **XML & Java** обсуждении: http://www.itworld.com/jump/ jw-0126-xpath/forums.itworld.com/webx?230@@.ee6b78f!skip=103
- Для постоянного получения различных советов, подпишитесь *Java in the Enterprise* еженедельную рассылку (find it under the "Application Development Series" section): http://www.itworld.com/ cgi-bin/subcontent12.cgi

Другие важные ресурсы

- Сходите на страничку Sun по БПТ JSP : http://java.sun.com/ products/jsp/index.html
- Почитайте о DOM спецификации на сайте W3C : http://www.w3.org/DOM/
- Почитайте о XPath спецификации на сайте W3C : http://www.w3.org/TR/xpath.html
- Хорошие учебные пособия по XML и DOM находятся на XML Zone of IBM's developerWorks: http://www.ibm.com/developer/xml
- "Addressing Infosets with XPath" by Aaron Skonnard (*MSDN Magazine,* July 2000)

подробрая статья о XPath: http://msdn.microsoft.com/msdnmag/issues/0700/xml/xml0700.asp

- Почитайте о Apache Xerces XML парсере и Apache Xalan XSL процессоре: http://xml.apache.org/
- Для более детального описания синтаксиса XPath с примерами сходите на ZVON.org's XPath tutorial: http://www.zvon.org/HTMLonly/XPathTutorial/General/examples.html
- О связке JavaBeans и XML, почитайте "What is JOX?": http://www.wutka.com/jox.html