

Применение XSL-FO

by J. David Eisenberg

Данная серия документов подготовлена на основе материалов сайта Школы Консорциума W3C. Этот сайт является экспериментальным сервером, на котором содержание документов хранится в формате XML. Пользователям сайта эти документы доступны в виде HTML (преобразование на стороне клиента с помощью таблицы стилей XSLT) и в виде PDF (преобразование тех же документов в XSL-FO, а затем в формат PDF).

Содержание

- **Часть 1**²
 - Инициализация
 - Макеты страниц
 - Зона основного содержания
 - Направления
 - Задание размеров областей
- **Часть 2**³
 - Последовательности страниц
 - Дополнительная информация
 - Лицевая страница
 - Создание PDF-файла
 - Работа с основными страницами
- **Часть 3**⁴
 - Заголовки и абзацы
 - Списки
- **Часть 4**⁵
 - Списки определений
 - Таблицы
 - Заключение

Часть 1

Вступление

Спецификация расширяемого языка таблиц стилей XSL консорциума W3C, датированная 17 января 2001 года состоит из двух частей:

2: http://xml.nsu.ru/extra/usingxslfo_1.xml

3: http://xml.nsu.ru/extra/usingxslfo_2.xml

4: http://xml.nsu.ru/extra/usingxslfo_3.xml

5: http://xml.nsu.ru/extra/usingxslfo_4.xml

- XSLT: язык преобразований XML-документов, и
- XSL-FO: форматирующие объекты XSL - XML-словарь, задающий форматирующую семантику.

XSLT прост в изучении и использовании. С минимальными затратами времени с его помощью разработчики могут преобразовывать XML-файлы в HTML-файлы, которые пользователи могут просматривать в своих браузерах. Поэтому разработчики встретили появление XSLT с огромным энтузиазмом. Раздел сайта XML.com "[Transforming XML](http://xml.com/pub/q/transformingxml)"⁶ – прекрасное место для первого знакомства с XSLT.

XSL-FO сам по себе является основанным на XML языком разметки, который позволяет с большой точностью задавать макет и другую стилевую информацию, относящуюся к содержанию документов. Разметка XSL-FO достаточно сложна. Кроме того, она довольно громоздка – практически единственным разумным способом создания XSL FO-файла является генерация его из исходного документа с помощью XSLT. И, наконец, если вы имеете файл XSL-FO, его нужно как-то выводить на носитель. Пока существует очень немного средств реализации этого последнего шага. Поэтому XML-FO не так быстро завоевывает популярность, как XSLT.

Эта статья не раскроет вам XSL-FO во всех деталях, в ней мы поговорим лишь об основных особенностях XSL-FO. В качестве примера мы будем заниматься созданием небольшой брошюры, которая должна будет вставляться в CD-ROM для изучающих испанский язык. Для конечного преобразования файлов FO в PDF мы будем использовать [Apache Software Foundation's FOP tool](http://xml.apache.org/fop/)⁷.

Инициализация

Поскольку XSL-FO-файл является XML-документом, он должен начинаться со стандартной процессуальной инструкции XML и должен иметь корневой элемент:

```
<?xml version="1.0" encoding="utf-8"?>
<fo:root>
```

Структура основной части документа выглядит так:

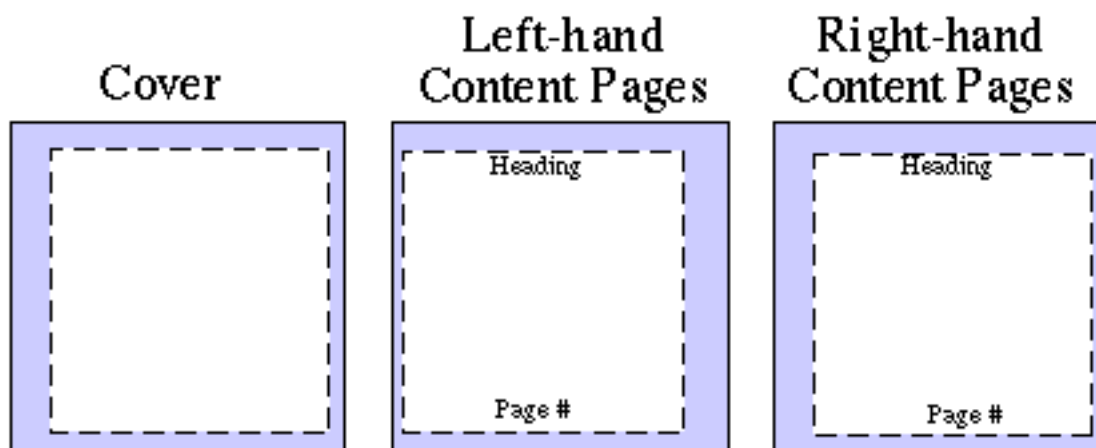
- **Набор мастеров макета**, который состоит из:
 - Описаний внешнего вида страниц, которые могут появляться в документе.
 - Описаний последовательностей, в которых идут эти форматы страниц.
- **Страниц и их содержимого**.

Макеты страниц

Вслед за первым тэгом FO-документа `<fo:root>`, нам нужно задать описания видов страниц, которые могут появляться в документе. Наш документ будет состоять из трех видов страниц, они показаны на рисунке. Для того, чтобы зарезервировать место для склейки страниц, обложка и правые страницы разворотов будут иметь слева отступ большего размера. Кроме того, основные страницы будут иметь области заголовков и футеры.

6: <http://xml.com/pub/q/transformingxml>

7: <http://xml.apache.org/fop/>



Начнем с задания высоты и ширины страниц, а также отступов. В качестве единиц измерения мы будем использовать сантиметры, но вы можете использовать любые единицы измерения, которые применяются в CSS, например, px (пиксели), pt (пункты), em, in, mm и так далее. Каждое из описаний вида страниц называется простой мастер-страницей (simple-page-master) и должно иметь атрибут `master-name`, он задает имя мастер-страницы, по которому позже мы будем к ней обращаться.

```
<fo:layout-master-set>
  <fo:simple-page-master master-name="cover"
    page-height="12cm"
    page-width="12cm"
    margin-top="0.5cm"
    margin-bottom="0.5cm"
    margin-left="1cm"
    margin-right="0.5cm">
  </fo:simple-page-master>

  <fo:simple-page-master master-name="leftPage"
    page-height="12cm"
    page-width="12cm"
    margin-left="0.5cm"
    margin-right="1cm"
    margin-top="0.5cm"
    margin-bottom="0.5cm">
  </fo:simple-page-master>

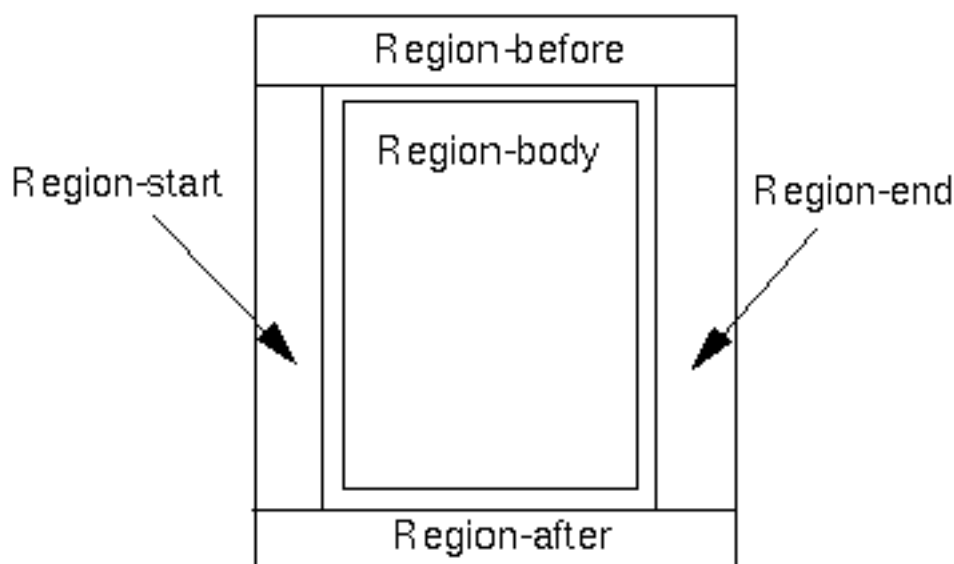
  <fo:simple-page-master master-name="rightPage"
    page-height="12cm"
    page-width="12cm"
    margin-left="1cm"
    margin-right="0.5cm"
    margin-top="0.5cm"
    margin-bottom="0.5cm">
  </fo:simple-page-master>

  <!-- more info will go here -->
</fo:layout-master-set>
```

Отступы здесь – это области, в которых не будет выводиться никакое печатное содержание.

Зона основного содержания

Все содержимое страниц будет выводиться в областях, отмеченных на приведенных выше схемах пунктирными линиями. Это – зоны основного содержания (они называются [page-reference-area](#)), каждая разделяется на пять областей, как показано ниже:



Направления

Прежде, чем мы продолжим, давайте немного поговорим о терминологии. Когда мы задаем отступы, мы используем слова [top](#), [bottom](#), [left](#) и [right](#), поскольку никто не станет спорить о том, какой край листа бумаги называть верхним краем, левым краем, и так далее. Но при разговоре о зоне основного содержания мы будем использовать другие слова, поскольку не во всех языках письмо ведется слева направо и сверху вниз.

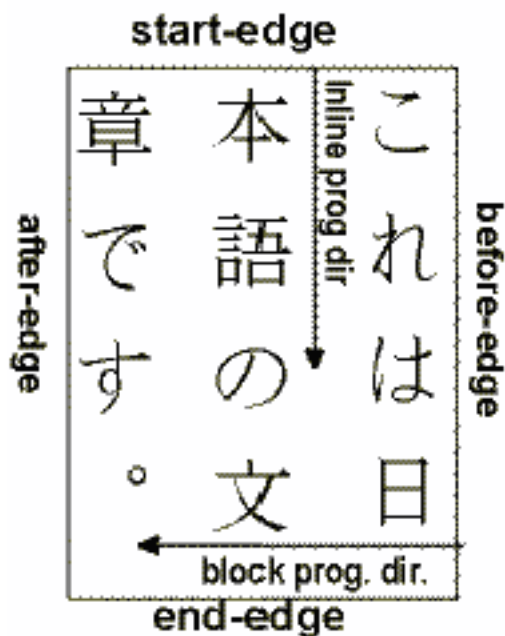
В FO считается, что страница состоит из двух классов элементов: **блоковых элементов** (например, абзацы), они всегда начинаются с новой строки, и **внутри-строчных элементов** (например, области полужирного шрифта или курсива), они не переводят строку. FO-параметр [block-progress-direction](#) – это направление, в котором абзацы последовательно размещаются на странице. [before-edge](#) предшествует абзацу, а [after-edge](#) следует за ним.

[inline-progress-direction](#) – это порядок, в котором символы размещаются в строке. [start-edge](#) предшествует строке, а [end-edge](#) следует за ней.

Как показано на рисунке выше, в иврите начальный край и конечный край расположены на прямо противоположных краях, нежели в английском языке (подобная ситуация и в арабском.)



В японском языке письмо идет как показано на следующем рисунке. Этот рисунок взят из спецификации по XSL.



Преимущество нового словаря разметки в его независимости от языка. Если вы хотите, чтобы заголовок располагался на противоположной стороне страницы, нежели обычный текст, вам нужно просто установить его параметр `text-align="end"` и результат будет примерно такой:

Интересный заголовок

Такие заголовки выглядят необычно и привлекут внимание читателя.

Если позже документ будет переводиться на арабский или японский язык, можно быть уверенным в том, что заголовок по-прежнему появится на «противоположной» стороне, хотя это будет совсем другая сторона. Не нужно будет проверять весь документ, чтобы менять в нем атрибуты `left` и `right` или переключать их в `top` или `bottom`.

Задание размеров областей

На обложке не нужен ни заголовок, ни футер, поэтому нам нужно задать лишь те параметры основной области `region-body`, которые в следующем листинге выделены полужирным:

```
<fo:simple-page-master master-name="cover"
  page-height="12cm"
  page-width="12cm"
  margin-top="0.5cm"
  margin-bottom="0.5cm"
  margin-left="1cm"
  margin-right="0.5cm">
  <fo:region-body
    margin-top="3cm" />
</fo:simple-page-master>
```

Правые и левые страницы разворотов имеют заголовки и футеры, поэтому нам нужно задать параметр `extent` (растяжение) пред-области `region-before` и после-области `region-after`.

```
<fo:simple-page-master master-name="leftPage"
  page-height="12cm"
  page-width="12cm"
  margin-left="0.5cm"
  margin-right="1cm"
  margin-top="0.5cm"
  margin-bottom="0.5cm">
  <fo:region-before extent="1cm"/>
<fo:region-after extent="1cm"/>
<fo:region-body
  margin-top="1.1cm"
  margin-bottom="1.1cm" />
</fo:simple-page-master>

<fo:simple-page-master master-name="rightPage"
  page-height="12cm"
  page-width="12cm"
  margin-left="1cm"
  margin-right="0.5cm"
  margin-top="0.5cm"
  margin-bottom="0.5cm">
  <fo:region-before extent="1cm"/>
<fo:region-after extent="1cm"/>
<fo:region-body
  margin-top="1.1cm"
  margin-bottom="1.1cm" />
</fo:simple-page-master>
```

Важное замечание: Отступы, которые вы задаете для основной области `region-body`, должны быть больше или равны растяжению `region-before` и `region-after` (и расширений областей `region-start` и `region-end`, если вы их используете – в настоящее время форматер FOP их не поддерживает.). Если вы сделаете так:

```
<fo:region-before extent="1cm"/>
<fo:region-after extent="1cm"/>
<fo:region-body
```

```
margin-top="0.20cm"  
margin-bottom="0.20cm" />
```

вы получите примерно такой результат:

The Actor - A Novel in Bad Style
“Why are you always stepping on my toes?” asked
Hubert. “Couldn’t you wait until I finish my introduction?”

Часть 2

Последовательности страниц

После задания мастер-страниц можно определить порядок, в котором будет использоваться набор этих мастер-страниц при генерации последовательности страниц.

Создаваемый нами документ состоит из лицевой страницы, за которой следуют страницы с основным содержанием документа. Таким образом, у нас имеется две последовательности страниц: лицевая страница (это последовательность страниц, состоящая только из одной страницы), и основные страницы (эта последовательность образована правыми и левыми страницами разворотов).

Можно задавать последовательности страниц, которые содержат мастер-страницу только для обложки, но мы ничего не выигрываем от их создания (другое дело, если как во многих книгах, несколько страниц может быть отнесено к лицевым – тогда игра стоит свеч). Поэтому мы сосредоточимся главным образом на задании последовательности мастер-страниц для основного содержания книги. В обычных условиях содержание книги размещается на левых страницах разворотов с четными номерами, за которыми следуют правые страницы с нечетными номерами. Это означает, что оборотная сторона лицевой страницы будет иметь номер два. Это задание показано ниже, для удобства показаны и номера строк:

```
1 <fo:page-sequence-master master-name="contents">  
2   <fo:repeatable-page-master-alternatives>  
3     <fo:conditional-page-master-reference  
4       master-name="leftPage"  
5       odd-or-even="even"/>  
6     <fo:conditional-page-master-reference  
7       master-name="rightPage"  
8       odd-or-even="odd"/>  
9   </fo:repeatable-page-master-alternatives>  
10 </fo:page-sequence-master>  
11
```

- **Строка 1** Определение и наименование данного мастера последовательности страниц
- **Строка 2** Последовательность образована мастер-страницами, которые должны по очереди вступать в действие в зависимости от заданных условий
- **Строки 3-5** Выбирается мастер-страница с именем `leftPage` если генерируемая страница имеет четный номер
- **Строки 6-8** Выбирается мастер-страница с именем `rightPage` если генерируемая страница

имеет нечетный номер

Дополнительная информация

Хотя это, вероятно, наиболее распространенная последовательность страниц, возможны и другие варианты. Если у вас документ имеет односторонний формат, в котором содержание размещается на страницах, соответствующих правым страницам разворотов, и при этом вы хотите определить максимальное количество страниц, вам следует применять следующий мастер последовательности страниц:

```
<fo:page-sequence-master master-name="example">
  <fo:repeatable-page-master-reference
    maximum-repeats="10" master-name="rightPage"/>
</fo:page-sequence-master>
```

Атрибут `maximum-repeats` может также применяться и в элементе `repeatable-page-master-alternatives`.

Атрибут `maximum-repeats` можно использовать для ограничения количества страниц, которое может быть сгенерировано в данной последовательности. Атрибут `maximum-repeats` также может применяться и в элементе `repeatable-page-master-alternatives`.

Другие условия, которые вы можете использовать в элементе `conditional-page-master-reference` таковы:

Условие	Описание
<code>page-position</code>	Данная мастер-страница используется в зависимости от того, где она размещается в последовательности страниц <code>page-sequence</code> . Допустимыми значениями являются <code>first</code> , <code>last</code> , <code>rest</code> (то есть, не первая страница), или <code>any</code>
<code>blank-or-not-blank</code>	Данная мастер-страница используется в зависимости от того, пустая это страница или нет. Допустимые значения <code>blank</code> или <code>not-blank</code> . Значение <code>blank</code> используется для поддержания четности, например, для генерации пустой страницы, чтобы глава всегда заканчивалась на нечетной странице

Лицевая страница

Теперь мастер-страницы и последовательности заданы, и мы можем начать размещение содержания на этих страницах. Это делается с помощью определения, когда используется та или иная последовательность страниц, а также в какую область должна помещаться информация. Вот описание лицевой страницы. Для вывода символа копирайта мы используем числовую ссылку на сущность `©`:

```
1 <fo:page-sequence master-name="cover">
2 <fo:flow flow-name="xsl-region-body">
3   <fo:block font-family="Helvetica" font-size="18pt"
4     text-align="end">
5     Spanish Review Handbook
6   </fo:block>
7   <fo:block font-family="Helvetica" font-size="12pt"
8     text-align="end" space-after="36pt">
9     Copyright &#169; 2001 J. David Eisenberg
10  </fo:block>
```



```

11     <fo:block text-align="end">
12         A Catcode Production
13     </fo:block>
14 </fo:flow>
15 </fo:page-sequence>

```

■ **Строка 1** Задание последовательности страниц, в которой будет размещаться содержание. Обратите внимание: ее легко перепутать с элементом `<fo:page-sequence-master>`; здесь слово `master` относится к имени атрибута, а не к имени элемента!

■ **Строка 2** Следующее содержание идет в область `xsl-region-body` страницы.

■ **Строки 3-6** Данное текстовое содержание (Spanish Review Handbook) должно идти с новой строки (`<fo:block>`) с заданным кеглем и гарнитурой. Обратите внимание, что атрибут `text-align` имеет значение `end`.

■ **Строки 7-10** Еще один блок для информации о копирайте, применяется другой шрифт. Для резервирования дополнительного пустого пространства после этой информации используется атрибут `space-after`.

■ **Строки 13-14** Еще один блок с издательской информацией.

■ **Строки 14-15** Конец определения содержимого страницы.

Создание PDF-файла

Теперь у нас есть некоторое определенное содержание и мы можем обработать эту страницу для вывода на печать. Если вы хотите попробовать это сделать сами, скачайте [Apache Software Foundation's FOP](http://xml.apache.org/fop/)⁸ и установите в соответствии с инструкциями. Вам понадобится:

- Java 1.1.x или более поздняя версия;
- XML-парсер, который поддерживает SAX и DOM;
- Для примеров других статей этой серии вам понадобится парсер XSLT (если вы скачаете [Xalan](http://xml.apache.org/xalan-j/)⁹, вы сразу получите Xerces, (парсер XML) и Xalan (парсер XSLT));
- Библиотека SVG, которая находится в файле [w3c.jar](#), он идет вместе с FOP.

Например, для Linux-систем вам нужно разместить все файлы.jar в подходящую директорию и создать скрипт с именем `fop.sh`, который выглядит так:

```

java -cp \
/usr/local/xml-jar/fop.jar:/usr/local/xml-jar/w3c.jar:\
/usr/local/xml-jar/xml.jar:/usr/local/xml-jar/xerces.jar:\
/usr/local/xml-jar/xalan.jar:/usr/local/xml-jar/bsf.jar \
org.apache.fop.apps.CommandLine $1 $2

```

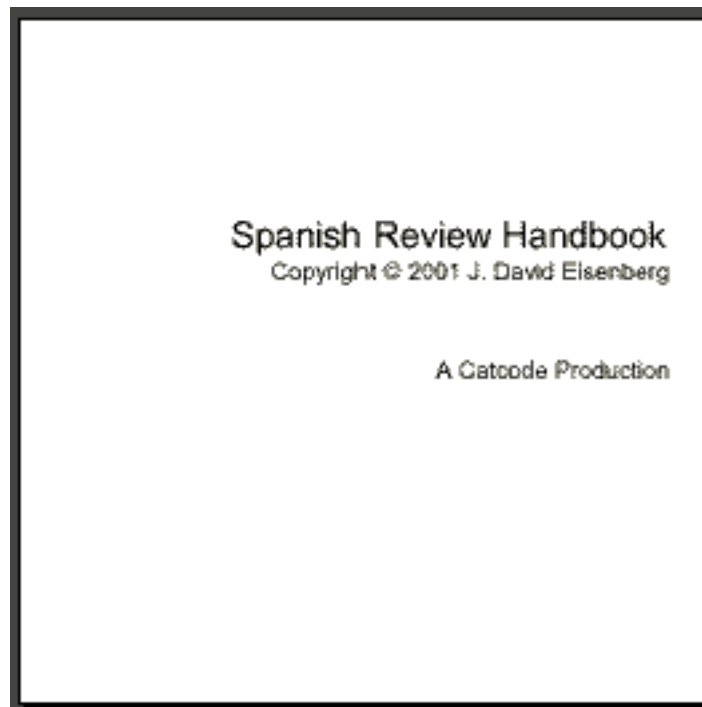
Скрипт вызывается командой вида `fop.sh spanish1.fo spanish1.pdf`. Результатом будет создание PDF-файла. Чтобы просмотреть этот файл, нужен просмотрщик PDF. [Adobe](http://www.adobe.com/)¹⁰ Acrobat Reader работает на Linux, Macintosh, и Windows. Пользователи Linux могут также использовать [xpdf](http://www.foolabs.com/xpdf/)¹¹, просмотрщик PDF для X-Window. Вот результирующий документ в уменьшенном виде:

8: <http://xml.apache.org/fop/>

9: <http://xml.apache.org/xalan-j/index.html>

10: <http://www.adobe.com/>

11: <http://www.foolabs.com/xpdf/>



Тут совершенно очевидно не хватает графических элементов, которые бы украсили обложку. Графика добавляется элементом `external-graphic`, чей атрибут `src` указывает URI нужного изображения. Добавочные элементы выделены на следующем листинге:

```
<fo:block font-family="Helvetica" font-size="12pt"
  text-align="end" space-after="36pt">
  Copyright #169; 2001 J. David Eisenberg
</fo:block>
<fo:block text-align="end">
  <fo:external-graphic src="file:images/catcode_logo.jpg"
    width="99px" height="109px"/>
</fo:block>
<fo:block>
  A Catcode Production
</fo:block>
```

Ну вот, стало лучше, не правда ли?



Работа с основными страницами

Начнем работу с основными страницами документа. Нам нужно поместить информацию в области страницы `xsl-region-before`, `xsl-region-after` и `xsl-region-body`.

```
1 <fo:page-sequence master-name="contents" initial-page-number="2">
2   <fo:static-content flow-name="xsl-region-before">
3     <fo:block font-family="Helvetica" font-size="10pt"
4       text-align="center">
5       Spanish Review Handbook
6     </fo:block>
7   </fo:static-content>
8
9   <fo:static-content flow-name="xsl-region-after">
10    <fo:block font-family="Helvetica" font-size="10pt"
11      text-align="center">
12      Página <fo:page-number />
13    </fo:block>
14  </fo:static-content>
15
16  <fo:flow flow-name="xsl-region-body">
17    <fo:block font-size="14pt">
18      Watch this space!
19    </fo:block>
20  </fo:flow>
21 </fo:page-sequence>
```

■ **Строка 1** Начало новой последовательности страниц с применением последовательности с именем мастера `contents`. Начало нумерации страниц с цифры 2.

■ **Строки 2-7** В настоящее время преобразователи FO в PDF требуют, чтобы содержание

области заголовка страниц было одинаково для всех страниц, поэтому для заполнения области `xsl-region-before` нужно задать статическое содержимое `<fo:static-content>`, а не переменное `<fo:flow>`.

■ **Строки 9-14** Области футеров также должны содержать `<fo:static-content>`. Обратите внимание: В строке 12 показано, как следует вставлять номер текущей страницы с помощью элемента `<fo:page-number/>`. Ссылка на сущность `á` представляет символ `á`.

■ **Строки 16-20** Задание основного содержания для заполнения области `xsl-region-body` в этой последовательности страниц.

Вот так будет выглядеть результат:



Текущий вариант PDF-документа¹²

В следующей части мы поговорим о том, как XSLT может значительно облегчить создание FO-элементов. Кроме того, вы узнаете, как задавать в документах таблицы и списки.

Часть 3

Заголовки и абзацы

Разобравшись с обложкой и с общей конструкцией внутренних страниц, мы теперь готовы приступить к основному информационному наполнению брошюры. Начнем со страницы, которая должна содержать следующий текст:

¹²: http://xml.nsu.ru/extra/usingxslfo_samples/spanish.pdf

Introduction

This handbook covers the major topics in Spanish but is by no means complete.

Accents

When we pronounce English words, one syllable is usually emphasized (**stressed**, in linguistic terms). The stressed syllable is underlined in the following words: computer, language, development, succeeds. Spanish words also have a stressed syllable, and there are rules for determining which syllable carries the emphasis.

Заголовки и абзацы будут образованы элементами `<fo:block>`, а выделенные полужирным шрифтом и подчеркнутые слова будут заключаться в элементы `<fo:inline>`. Начнем с описания первого заголовка:

```
<fo:block
  font-size="14pt" font-family="sans-serif"
  font-weight="bold" color="green"
  space-before="6pt" space-after="6pt">
  Introduction
</fo:block>
```

Свойства `space-before` и `space-after` – лишь два из многих возможных свойств блочных элементов. Многие из них точно совпадают с аналогичными свойствами, которые применяются в каскадных таблицах стилей CSS:

■ Свойства шрифта

`font-family`, `font-weight`, `font-style` (`italic`), `font-size`, `font-stretch`, `font-variant` (`small-caps`)

■ Свойства фона

`background-color`, `background-image`, `background-repeat`, `background-attachment` (`scroll` или `fixed`)

■ Свойства границ

`border-location-info` где:

`location` может быть `before`, `after`, `start`, `end`, `top`, `bottom`, `left`, или `right`

`info` может быть `style`, `width`, или `color`

■ Свойства отбивок

`padding-location` где:

`location` может быть `before`, `after`, `start`, `end`, `top`, `bottom`, `left`, или `right`

■ Свойства отступов

`margin-location` где:

`location` может быть `top`, `bottom`, `left` или `right`

■ Свойства выравнивания текста

`text-align` и `text-align-last` (для последней строки текста в блоке); значениями может быть `start`, `end`, `left`, `right` или `center`

■ Свойства абзацных смещений

`text-indent` (для первой строки), `start-indent`, `end-indent`

■ Другие свойства

`wrap-option` (`no-wrap` или `wrap`); `widows` и `orphans` (определение того, сколько строк должно быть оставлено наверху или внизу страницы) `break-after` и `break-before` (когда должна разбиваться страница или колонка); `reference-orientation` (повернутый текст, приращения по 90 градусам)

Имея такое изобилие возможностей, мы можем получить весьма сложное определение блока для каждого абзаца текста. Следующее определение использует запись в виде "составного типа данных" (точка в качестве разделителя), оно предоставляет формату некоторую свободу в выборе размеров пустого пространства до и после абзаца:

```
<fo:block
  text-indent="1em"
  font-family="sans-serif" font-size="12pt"
  space-before.minimum="2pt"
  space-before.maximum="6pt"
  space-before.optimum="4pt"
  space-after.minimum="2pt"
  space-after.maximum="6pt"
  space-after.optimum="4pt">
This handbook covers the major topics in Spanish, but is by
no means complete.
</fo:block>
```

Если в документе имеется двенадцать или тринадцать заголовков и семьдесят или восемьдесят абзацев, вряд ли вам захочется вручную выписывать все необходимые форматирующие объекты. И здесь вступает в дело XSLT. Мы будем создавать документ в виде HTML, а затем с помощью XSLT трансформируем его в гораздо более сложный XSL-FO. Вот необходимый нам HTML:

```
<h3>Introduction</h3>
<p>
This handbook covers the major topics in Spanish, but is by
no means complete.
</p>
<h3>Accents</h3>
<p>
When we pronounce English words, one syllable is usually
emphasized (<b>stressed</b>, in linguistic terms).
The stressed syllable is underlined in the following
words: com<u>pu</u>ter, <u>lan</u>guage, de<u>vel</u>opment,
suc<u>ceeds</u>. Spanish
words also have a stressed syllable, and there are rules for
determining which syllable carries the emphasis.
</p>
```

А вот шаблоны, которые необходимы для преобразования заголовков и абзацев:

```
<xsl:template match="h3">
  <fo:block font-size="14pt" font-family="sans-serif"
    font-weight="bold" color="green"
    space-before="6pt" space-after="6pt">
  <xsl:apply-templates/>
  </fo:block>
</xsl:template>

<xsl:template match="p">
  <fo:block
```

```
text-indent="1em"
font-family="sans-serif" font-size="12pt"
space-before.minimum="2pt"
space-before.maximum="6pt"
space-before.optimum="4pt"
space-after.minimum="2pt"
space-after.maximum="6pt"
space-after.optimum="4pt">
<xsl:apply-templates/>
</fo:block>
</xsl:template>
```

А что делать с кодом инициализации страницы, который мы рассматривали в предыдущих частях статьи? Его нужно создавать шаблоном, который будет применяться к тэгам `<html>` и `<body>` исходного HTML-документа. Мы не станем здесь его рассматривать, но вы можете его открыть отдельно.

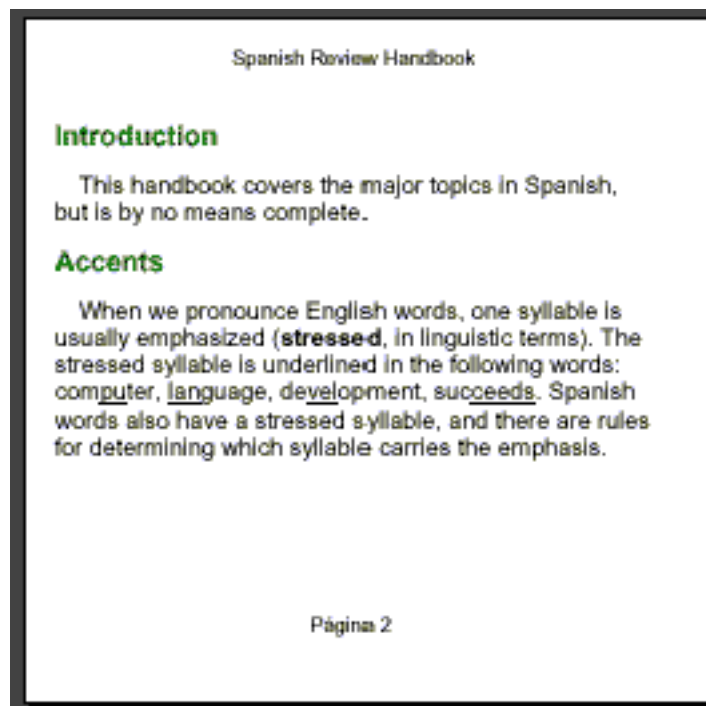
Остались тэги `` и `<u>`. Это внутри-строчные элементы и они будут создаваться форматирующим объектом `<fo:inline>` (заодно мы здесь учтем и тэг `<i>`).

```
<xsl:template match="b">
  <fo:inline font-weight="bold">
    <xsl:apply-templates/>
  </fo:inline>
</xsl:template>

<xsl:template match="u">
  <fo:inline text-decoration="underline">
    <xsl:apply-templates/>
  </fo:inline>
</xsl:template>

<xsl:template match="i">
  <fo:inline font-style="italic">
    <xsl:apply-templates/>
  </fo:inline>
</xsl:template>
```

После того, как мы создадим нужный HTML-файл, пропустим его через XSLT и FOP, мы получим следующую страницу:

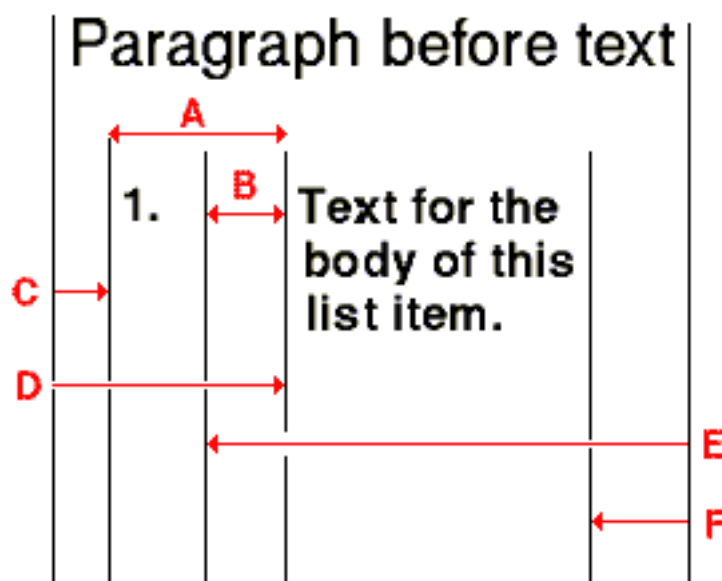


Списки

Теперь мы добавим в наш документ списки. Вот содержание, которое нужно добавить:

1. If a syllable has an accent mark, that syllable always gets the stress: *acción* (action), *teléfono*.
2. If the word ends with a vowel, **n**, or **s**, the next-to-last syllable gets the stress: *amigo*, *hablan* (they talk), *animales*.
3. All other words are accented on the last syllable: *hotel*, *similar*, *español*.

Для организации списков используется четыре элемента. Элемент `<fo:list-block>` содержит несколько пунктов списка `<fo:list-items>`. Каждый пункт списка состоит из элемента отметки пункта списка `<fo:list-item-label>` и элемента тела пункта списка `<fo:list-item-body>`. Отступы задаются с помощью атрибутов, показанных на следующей диаграмме:



- A. Свойство `provisional-distance-between-starts`
- B. Свойство `provisional-label-separation`
- C. Свойство `start-indent` объекта `list-item-label`
- D. Свойство `start-indent` объекта `list-item-body`
- E. Свойство `end-indent` объекта `list-item-label`
- F. Свойство `end-indent` объекта `list-item-body`

Теперь можно создать XSLT-шаблон для работы с упорядоченными списками. Мы зададим начальный отступ отметки пункта списка, а остальное предоставим формату FOP. Использование относительных единиц `em` при задании отступа позволяет получить хороший результат при любом размере шрифта.

```
<xsl:template match="ol">
  <fo:list-block
    space-before="0.25em" space-after="0.25em">
    <xsl:apply-templates/>
  </fo:list-block>
</xsl:template>

<xsl:template match="ol/li">
  <fo:list-item space-after="0.5ex">
    <fo:list-item-label start-indent="1em">
      <fo:block>
        <xsl:number/>.
      </fo:block>
    </fo:list-item-label>
    <fo:list-item-body>
      <fo:block>
        <xsl:apply-templates/>
      </fo:block>
    </fo:list-item-body>
  </fo:list-item>
</xsl:template>
```

Неупорядоченные списки организуются аналогичным образом, только вместо цифр мы используем буллеты. Соответствующий кусок определения пунктов списка в случае неупорядо-

ченного списка выглядит так:

```
<xsl:template match="ul/li">
  <fo:list-item>
    <fo:list-item-label start-indent="1em">
      <fo:block>
        &#x2022;
      </fo:block>
    <!-- и т.д. -->
  </fo:list-item>
</xsl:template>
```

Часть 4

Списки определений

Использование модели списка для создания списка определений, в котором определяемые термины и их определения должны уместиться на одной строке потребовало бы создания очень сложной таблицы стилей XSLT. (Можете посмотреть сами в спецификации XSL, раздел 6.8.1.1.) Поэтому мы просто разместим термины и их определения на различных строках, как это и делается обычно в HTML.

```
<xsl:template match="dl">
  <fo:block space-before="0.25em" space-after="0.25em">
    <xsl:apply-templates/>
  </fo:block>
</xsl:template>

<xsl:template match="dt">
  <fo:block><xsl:apply-templates/></fo:block>
</xsl:template>

<xsl:template match="dd">
  <fo:block start-indent="2em">
    <xsl:apply-templates/>
  </fo:block>
</xsl:template>
```

Здесь приведен фрагмент буклета, на котором имеется упорядоченный список и список определений. Обратите внимание, что текст самостоятельно перетекает с одной страницы на другую, для этого не нужно ничего делать специально.

words also have a stressed syllable, and there are rules for determining which syllable carries the emphasis.

1. If a syllable has an accent mark, that syllable always gets the stress: *acción* (action), *teléfono*.
2. If the word ends with a vowel, n, or s, the next-to-last syllable gets the stress: *amigo*, *hablan* (they talk), *animales*.

Página 2

Spanish Review Handbook

3. All other words are accented on the last syllable: *hotel*, *similar*, *español*.

Parts of Speech

Words fall into one of seven classes, known as parts of speech. We will deal with only four of them.

Nouns

A person, place, or thing. (*Juan*, *telephone*)

Verbs

An action or state of being. (*run*, *become*)

Adjectives

Таблицы

Когда мы начинаем говорить о глаголах, нам нужно показать классическую таблицу спряжений:

Singular	Plural
yo canto	nosotros cantamos
tú cantas	vosotros cantáis
él canta ella canta	ellos cantan ellas cantan

Таблица в XSL-FO образуется следующей иерархией элементов:

```
<fo:table-and-caption>
  <fo:table-caption>
  <fo:table>
    <fo:table-column>
    <fo:table-header>
      <fo:table-row>
        <fo:table-cell>
```

```

    <fo:table-body>
      <fo:table-row>
        <fo:table-cell>
      <fo:table-footer>
        <fo:table-row>
          <fo:table-cell>

```

Элемент `<fo:table>` соответствует HTML-тэгу `<table>`; элемент `<fo:table-body>` соответствует тэгу `<tbody>`. Единственное значимое различие состоит в добавлении элемента `<fo:table-column>`, который позволяет задавать ширину каждой колонки в таблице. Этот тэг можно применять для задания характеристик ячеек, принадлежащих одной и той же колонке. В текущей реализации FOP (январь 2001 года), элемент `<table-and-caption>` не поддерживается. Вам необходимо задавать ширину колонок с помощью атрибута `column-width` элемента `<fo:table-column>`. FOP пока не может автоматически определить необходимую ширину таблицы.

Следующий XSLT-код создает простые таблицы. Предполагается, что ширина всех колонок определяется в первом ряду таблицы и что все ширины заданы в пикселях. Здесь также предполагается, что в одном дюйме умещается 72 пикселя. Здесь не предусмотрена работа с ячейками, перекрывающими несколько рядов или колонок. Наберите больше воздуха, пример все равно достаточно длинный.

```

<!-- если объект table-and-caption поддерживается, он будет
охватывать таблицу -->

<xsl:template match="table">
  <xsl:apply-templates/>
</xsl:template>

<!-- Находим все атрибуты вида width=attribute всех элементов
<th> и <td> в первом ряде <tr> таблицы. Значения выражены в
пикселях, поэтому делим их на 72, чтобы получить дюймы -->

<xsl:template match="tbody">
  <fo:table>
    <xsl:for-each select="tr[1]/th|tr[1]/td">
      <fo:table-column>
        <xsl:attribute name="column-width"><xsl:value-of
          select="floor(@width div 72)"/>in</xsl:attribute>
      </fo:table-column>
    </xsl:for-each>

    <fo:table-body>
      <xsl:apply-templates />
    </fo:table-body>

  </fo:table>
</xsl:template>

<!-- Здесь не трудно: <tr> соответствует элементу <fo:table-row> -->

<xsl:template match="tr">
  <fo:table-row> <xsl:apply-templates/> </fo:table-row>
</xsl:template>

<!-- Обработка заголовочных ячеек таблицы. Их содержание должно

```

выводиться полужирным шрифтом и по умолчанию отцентрировано в ячейках. Обратитесь к тэгу <table> и посмотрите, задана ли толщина границ таблицы. -->

```
<xsl:template match="th">
  <fo:table-cell font-weight="bold" text-align="center">
    <xsl:if test="ancestor::table[1]/@border > 0">
      <xsl:attribute name="border-style">solid</xsl:attribute>
      <xsl:attribute name="border-width">1pt</xsl:attribute>
    </xsl:if>
    <fo:block>
      <xsl:apply-templates/>
    </fo:block>
  </fo:table-cell>
</xsl:template>
```

<!-- Обработка основных ячеек. Обратитесь к тэгу <table> и посмотрите, задана ли толщина границ таблицы. -->

```
<xsl:template match="td">
  <fo:table-cell>
    <xsl:if test="ancestor::table/@border > 0">
      <xsl:attribute name="border-style">solid</xsl:attribute>
      <xsl:attribute name="border-width">1pt</xsl:attribute>
    </xsl:if>
    <fo:block>
      <!-- установка того же выравнивания, что и в тэге <td> -->
      <xsl:choose>
        <xsl:when test="@align='left'">
          <xsl:attribute name="text-align">start</xsl:attribute>
        </xsl:when>
        <xsl:when test="@align='center'">
          <xsl:attribute name="text-align">center</xsl:attribute>
        </xsl:when>
        <xsl:when test="@align='right'">
          <xsl:attribute name="text-align">end</xsl:attribute>
        </xsl:when>
      </xsl:choose>
      <xsl:apply-templates/>
    </fo:block>
  </fo:table-cell>
</xsl:template>
```

Описание глаголов третьего лица в нашей таблице потребовало использования тэга
, он транслируется в FO следующим образом:

```
<xsl:template match="br">
  <fo:block><xsl:text>&#xA;</xsl:text></fo:block>
</xsl:template>
```

В итоге наша брошюра будет выглядеть так:

Spanish Review Handbook											
Regular Spanish verbs come in three groups:											
<ul style="list-style-type: none">• Verbs ending in -ar, like cantar (to sing).• Verbs ending in -er, like comer (to eat).• Verbs ending in -ir, like vivir (to live).											
Here is the pattern for conjugating cantar in the present tense.											
<table><tr><th>Singular</th><th>Plural</th></tr><tr><td>yo canto</td><td>nosotros cantamos</td></tr><tr><td>tú cantas</td><td>vosotros cantáis</td></tr><tr><td>él canta</td><td>ellos cantan</td></tr><tr><td>ella canta</td><td>ellas cantan</td></tr></table>	Singular	Plural	yo canto	nosotros cantamos	tú cantas	vosotros cantáis	él canta	ellos cantan	ella canta	ellas cantan	
Singular	Plural										
yo canto	nosotros cantamos										
tú cantas	vosotros cantáis										
él canta	ellos cantan										
ella canta	ellas cantan										
Página 4											

Содержание брошюры в виде HTML¹³

Окончательный вариант таблицы стилей XSLT¹⁴

Окончательный вариант PDF-документа¹⁵

Заключение

Как вы увидели, комбинация XSLT и FO позволяет преобразовывать XHTML или другие XML-документы в формат, разработанный для печати. В этой статье мы раскрыли только некоторые из возможностей, которые предоставляет XSL-FO. За дополнительной информацией обращайтесь к 15 главе XML Bible, by Eliotte Rusty Harold¹⁶.

Вы можете скачать архив ZIP¹⁷ материалов, использовавшихся в этой статье, включая XSL-файл, который содержит в себе и другие аспекты преобразования HTML в FO и который применялся для преобразования этой статьи в формат PDF.

Developed by [Metaphor](#) (c) 2002

13: http://xml.nsu.ru/extra/usingxslfo_samples/spanish2.html

14: http://xml.nsu.ru/extra/usingxslfo_samples/spanish.xsl

15: http://xml.nsu.ru/extra/usingxslfo_samples/spanish2.pdf

16: <http://www.ibiblio.org/xml/books/bible/updates/15.html>

17: http://xml.nsu.ru/extra/usingxslfo_samples/usingxslfo.zip