Технологии Java

Введение в XML

http://kgeorgiy.info/courses/java-advanced/

Содержание

1. XML
2. Пространства имен
3. SAX
4. DOM
5. Заключение

Часть 1

XML

eXtensible Markup Language

- XML – текстовый язык хранения структурированных данных
- Предшественники XML
  - Standard Generalized Markup Language (SGML)
  - Hyper Text Markup Language (HTML)

Составляющие XML-документа

- Элементы (element) – задают структуру элемента
- Атрибуты (attribute) – дополняют информацию об элементе
- Символьные данные (character data) – текст внутри элемента
- Указания по обработке (processing instruction) – применяются разборщиками и другими программами
- Комментарии (comment) – игнорируются

Элемент

- Структура
  - Имя
  - Дети
  - Атрибуты
- Синтаксис

  <ИмяЭлемента Атрибуты>

  Дети

  </ИмяЭлемента>
- Сокращенный

  <ИмяЭлемента Атрибуты/>

Атрибут

- Структура
  - Имя
  - Значение
- Синтаксис

  имя = "значение"

  или

  имя = 'значение'

Пример: элементы и атрибуты

- Описание книги

```
<book
    caption        = "Рефакторинг"
    isbn           = "5-93286-045-6"
    publisher      = "Символ-Плюс"
    pages          = "430"
>
    <author name='Мартин' last-name='Фаулер'/>
</book>
```

XML-идентификаторы

- Первый символ
  - Буква, `_` или `:`
- Последующие символы
  - Первый символ, цифра, `-`или `.`
- Примеры
  - **hello**
  - **HelloWorld**
  - **HelloWorld156Times**
  - **hello-world**
  - **hello.world**

Символьные данные

- Простые символы
  - Пример: Привет!
- Именованные ссылки
  - **&**название**;**
  - Пример: Пусть a < b и b < c, тогда a < c
- Указание кода символа
  - **&#**НомерСимвола**;**
  - **&#x**ШестнадцатеричныйНомерСимвола**;**
  - Пример: **A A**

Именованные ссылки

- Значения по умолчанию

&amp;	&	ampersand
&lt;	<	less than
&gt;	>	greater than
&quot;	"	quotes
&apos;	'	apostrophe

Пробелы и переводы строк

- Пробельные символы
  - #x20 — Пробел
  - #x9 — Табуляция
  - #xA — Перевод строки
  - Другие переводы строк
- Другие переводы строк
  - #xD — возврат каретки
  - #x85 — перевод строки (IBM)
  - #x2028 — перевод строки (Unicode)
  - #xD #xA — перевод строки (DOS)
  - #xD #x85

Блоки символьных данных

- Синтаксис
  - <![CDATA[символьные данные]]>
- Примеры
  - <![CDATA[Внутри блока символьных данных спец символы можно писать непосредственно: ? < > ' "]]>
  - <![CDATA[Но, иногда, приходится делать и так:]]]]><![CDATA[>]]>

Указания по обработке

- Структура
  - Имя
  - Значение
- Синтаксис
  - **<?**имя значение**?>**

Комментарии

- Синтаксис
  - **<!--** комментарий **-->**
- Примеры

```
<!-- простой комментарий -->
<!--
    многострочный
    комментарий
-->
```

Общая структура XML-документа

- Пролог
  - Заголовок
  - Тип документа
  - Комментарии и указания по обработке
- Корневой элемент
- Эпилог
  - Комментарии и указания по обработке

Заголовок XML-файла (1)

- Позволяет указывать версию языка XML и кодировку, в которой записан файл
- Синтаксис
  - <?xml version="версия" encoding="кодировка"?>
- Пример

  Версия 1.1, кодировка 1251 (Windows Russian)

  <?xml version="1.1" encoding="WINDOWS-1251"?>

Заголовок XML-файла (2)

- Версии XML
  - 1.0 (пять редакций), версия по умолчанию
  - 1.1 (две редакции)
- Примеры
  - UTF-8 – версия по умолчанию
  - UTF-16 – при наличии byte-order mark
  - Другие кодировки могут не поддерживаться
    - WINDOWS-1251
    - Cp866

Часть 2

Пространства имен

Пространства имен

- Позволяют одновременно использовать одинаковые имена, придуманные разными людьми
- Пространство имен идентифицируется URI

Указание пространства имен

- Структура
  - Полное имя (qualified name)
  - Пространство имен (namespace)
  - Локальное имя (local name)
  - Префикс (prefix)
- Имя имеет вид
  - префикс**:**локальноеИмя
- Пространства имен связываются с префиксами с помощью атрибутов вида
  - **xmlns:префикс = "**пространство имен**"**

Пример: пространства имен

- Описание книги

```
<library:book
    xmlns:library  = "http://example.com/MyLibrary"
    xmlns:isbn     = "http://example.com/isbn"
    xmlns:issn     = "http://example.com/issn"
    isbn:number  = "5398866"
    issn:number  = "unknown"
    …
>
    <library:author library:name="Мартин" .../>
   …
</library:book>
```

Область действия префикса

- От элемента для которого определено отображение префикса вниз по дереву, до элементов для которых указано новое отображение этих префиксов
- Действие префикса можно отменить, указав отображение на пустую строку

Пространство имен по умолчанию

- Применяется для элементов для которых не указано пространство имен
- Объявление
  - **xmlns="**пространство имен"

Пример: пространства имен по умолчанию

- Описание книги

```
<book
    xmlns              = "http://example.com/MyLibrary"
    xmlns:library      = "http://example.com/MyLibrary"
    xmlns:isbn         = "http://example.com/isbn"
    xmlns:issn         = "http://example.com/issn"
    isbn:number        = "5398866"
    issn:number        = "unknown "
    …
>
     <author library:name="Мартин" .../>
   …
</book>
```

Часть 3

SAX

Simple API for XML

- Представляет XML-документ в виде последовательности событий
- Пакеты
  - org.xml.sax – модель SAX
  - java.xml.parsers – разборщики

Разбор XML

- Интерфейс XMLReader
- Методы
  - parse(InputSource) – разобрать XML-документ
  - setContentHandler(ContentHandler) – устанавливает приемник событий
  - setErrorHandler(ErrorHandler) – установить обработчик ошибок
  - setFeature(name, value) – установить настройку
  - setProperty(name, value) – установить свойство

Источники данных

- Класс InputSource
- Конструкторы
  - InputSource(InputStream) – из байтового потока
  - InputSource(Reader) – из символьного потока
  - InputSource(systemId) – по URL

Обработчик событий (1)

- Интерфейс ContentHandler
- Класс DefaultHandler
- Методы
  - setDocumentLocator(Locator locator) – установить источник местоположения
  - startDocument() – начало документа
  - endDocument() – конец документа
  - startElement(ns, localName, qName, Attributes) – открывающий тег элемента
  - endElement(ns, localName, qName) – закрывающий тег элемента

Обработчик событий (2)

- Методы
  - characters(char[] ch, offset, len) – последовательность символов
  - ignorableWhitespace(char[] ch, offset, len) – последовательность пробельных символов
  - processingInstruction(prefix, data) – рекомендация по обработке
  - startPrefixMapping(prefix, uri) – начало области использование префикса
  - endPrefixMapping(prefix, uri) – окончание области использование префикса

Атрибуты

- Интерфейс Attributes
- Методы
  - getLength() – количество атрибутов
  - getLocalName(index) – локальное имя
  - getQName(index) – полное имя
  - getURI(index) – пространство имен
  - getValue(index) – получить значение по индексу
  - getValue(qName) – получить значение по полному имени
  - getValue(ns, localName) – получить значение по пространству имен и локальному имени

Информация о местоположении

- Интерфейс Locator
- Методы
  - getLineNumber() – номер строки
  - getColumnNumber() – номер столбца
  - getSystemId() – URL разбираемого файла

Обработка ошибок

- Интерфейс ErrorHandler
- Методы
  - error(SAXParseException) – сообщение об исправимой ошибке
  - fatalError(SAXParseException) – сообщение о неисправимой ошибке
  - warning(SAXParseException) – сообщение о предупреждении

Исключения

- Класс SAXException
- Методы
  - getLineNumber() – номер строки
  - getColumnNumber() – номер столбца
  - getSystemId() – URL разбираемого файла

Создание SAXParser

- Класс SAXParserFactory
- Методы
  - static newInstance() – создать фабрику
  - newSAXParser() – создать разборщик
  - setFeature(uri, value) – установить настройку
  - setProperty(name, value) – установить свойство
  - setNamespaceAware(value) – установить поддержку пространств имен
- Класс SAXParser implements XMLReader

Часть 4

DOM

Document Object Model

- Представляет XML-документ в виде дерева узлов
- Пакеты
  - org.w3c.dom – модель DOM
  - java.xml.parsers – разборщики

Узлы

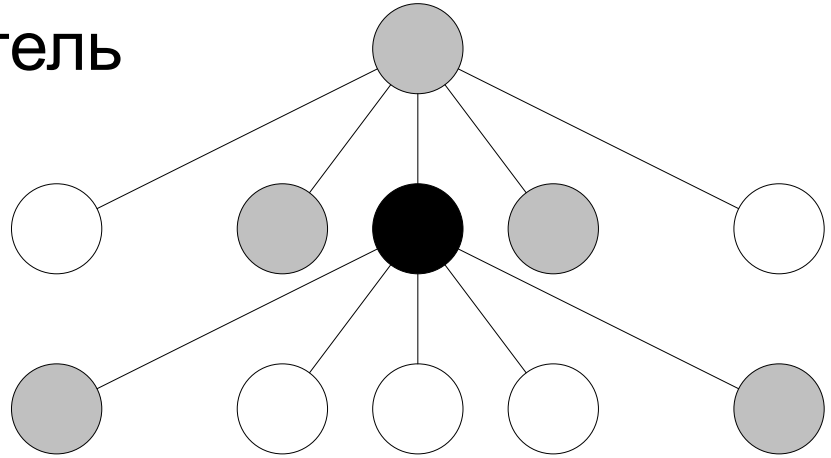
- Интерфейс Node
- Структура узла
  - getLocalName() – локальное имя
  - getNamespaceURI() – пространство имен
  - getPrefix() – префикс
  - getNodeName() – имя узла
  - getNodeValue() – значение узла
  - getNodeType() – тип узла

Типы узлов

Интерфейс	Описание	Имя	Значение
Attr	Атрибут	Имя	Значение
CDATASection	Блок символов	#cdata-section	Содержание
Comment	Комментарий	#comment	Содержание
Document	Документ	#document	
Element	Элемент	Имя	
ProcessingInstruction	Рекомендация по обработке	Имя	Значение
Text	Текст	#text	Содержание

Навигация по узлам

- Интерфейс Node
- Структура узла
  - getNextSibling() – предыдущий брат
  - getPreviousSibling() – следующий брат
  - getFirstChild() – первый ребенок
  - getLastChild() – последний ребенок
  - getParentNode() – родитель



Атрибуты

- Методы интерфейса Node
  - hasAttributes() – проверить наличие атрибутов
  - getAttributes() – получить атрибуты
- Интерфейс NamedNodeMap
- Методы
  - getLength() – количество элементов
  - item(index) – узел по индексу
  - getNamedItem(name) – узел по имени
  - getNamesItemNS(namespace, localName) – узел по имени и пространству имен

Вложенные узлы

- Методы интерфейса Node
  - hasChildNodes() – проверить наличие детей
  - getChildNodes() – получить детей
- Интерфейс NodeList
- Методы
  - getLength() – количество элементов
  - item(index) – элемент по индексу

Элементы

- Интерфейс Element
- Методы
  - Работа с атрибутами
    - getAttribute(name) – получить значение атрбута
  - Работа с вложенными элементами
    - getElementsByTagName(name) – получить всех потомков с заданным именем

Разбор XML в DOM

- Класс DocumentBuilder
- Методы
  - parse(File | InputStream | InputSource | URI) – построить документ
  - isNamespaceAware() – поддерживает ли пространства имен

Создание DocumentBuilder

- Класс DocumentBuilderFactory
- Методы
  - static newInstance() – создать фабрику
  - newDocumentBuilder() – создать DocumentBuilder
  - setFeature(uri, value) – установить настройку
  - setNamespaceAware(value) – установить поддержку пространств имен
  - setIgnoringComments(value) – установить игнорирование комментариев
  - setIgnoringElementContentWhitespace(value) – пропуск текстовых узлов из одних пробелов

Построение XML через DOM

- Интерфейс Document
  - createXXX(…) – создает элемент соответствующего типа
- Интерфейс Node
  - appendChild(node) – добавляет узел
  - removeChild(index) – удаляет узел

Вывод DOM в файл

- Класс TransformerFactory
- Методы
  - newInstance() – создать экземпляр фабрики
  - newTransformer() – создать пустое преобразование
- Класс Transformer
- Метод
  - transform(Source, Result) – преобразовать документ
- Класс DOMSource
- Класс StreamResult

Часть 6

Заключение

Ссылки (1)

- Extensible Markup Language 1.1 // http://www.w3.org/TR/2004/REC-xml11-20040204/
- XML Information Set // http://www.w3.org/TR/2004/REC-xml-infoset-20040204/
- Namespaces in XML 1.1 // http://www.w3.org/TR/2004/REC-xml-names11-20040204/

Ссылки (2)

- Document Object Model Level 3 Core Specification // http://www.w3.org/TR/2004/REC-DOM-Level-3-Core-20040407
- SAX Project // http://www.saxproject.org/
- Java API for XML Processing // http://java.sun.com/xml

Вопросы