XSD-XML

XML

Если с английского расшифровать аббревиатуру **XML**, то получится «e**X**tensible **M**arkup **L**anguage» — расширяемый язык разметки. Давайте рассмотрим это понятие. Язык разметки — это набор символов, который используют, чтобы обозначить, какую структуру должен иметь текст и как именно отображаться на странице сайта.

Язык XML — это метаязык, с помощью которого можно сделать не только саму разметку данных, но и описание всех её языков. С помощью XML разработчик может спроектировать собственную разметку, которая лучше всего будет подходить под текущий проект или задачу. Благодаря такому свойству этот язык называют расширяемым. Единственное условие — разработчик должен учитывать синтаксические правила языка, ведь XML имеет конкретную грамматику: словарь тегов и их атрибутов, а также набор правил.

XML — формат, ориентированный на текст. Он базируется на Unicode — стандарте кодирования символов, который включает в себя знаки почти всех письменных языков мира. Изначально этот формат придумали для более удобного хранения и передачи данных. Он имеет несколько основных преимуществ:

- **доступность**: этот формат могут прочитать как электронные устройства (компьютеры, телефоны и другие), так и человек — разработчик, программист и даже не специалист. Если нужно, XML-документы можно читать и менять с помощью стандартных инструментов редактирования текстов;
- **совместимость**: благодаря тому, что XML хранит все данные в текстовом формате, их удобно передавать — дополнительная конвертация не нужна. Также можно использовать одну систему для генерации данных и разметки, а затем обрабатывать эти данные в любых других системах вне зависимости от клиентской платформы или операционной системы;
- **универсальность**: с его помощью можно структурировать, трансформировать и запрашивать данные. Также XML можно читать не только в API (правилах взаимодействия одной компьютерной программы с другой), но и непосредственно в коде.

Где используется XML

XML используется везде, где требуется выделить логическое содержимое документа для обработки. Формат рекомендован Консорциумом Всемирной паутины (W3C), поэтому применяется в API, когда ответ от сервера поступает в виде XML-файлов.

XML позволяет:

- записывать иерархию — «один подчиняется другому»;
- размечать текст по смыслу от важного к второстепенному;
- хранить типовые данные — скрипты, настройки программ, названия чего-либо;
- размечать текст для машинного обучения;
- хранить результаты работы текстовых редакторов.

Иерархии с данными XML могут использоваться в разных языках программирования:

- XHTML — при отображении страниц в интернете;
- OWL и RDF — при описании структуры и ресурсов каталогов;
- SVG — при описании изображений в векторном формате;
- WSDL — при обращении к удаленным веб-сервисам и программам;
- XAML — при описании интерфейса приложений.

Структура XML-документа

XML документ состоит из частей, называемых элементами. Элементы составляют основу XML-документов. Они образуют структуры, которые можно обрабатывать программно или с помощью таблиц стилей. Элементы размечают именованные разделы информации. Элементы строятся с помощью тегов разметки, обозначающих имя, начало и конец элемента. Элементы могут быть вложены друг в друга, на верхнем уровне находится элемент, называемый элементом документа или корневым элементом, в котором содержатся остальные элементы. Например:

```
<?xml version="1.0"?>
<planets>
        <planet ID="1">
                <name>Mercury</name>
         </planet>
         <planet ID="2">
                <name>Venus</name>
         </planet>
<!-- There are more planets. -->
</planets>
```

Документ XML может располагаться в одном или нескольких файлах, причем некоторые из них могут находиться на разных машинах. В XML используется специальная разметка для интеграции содержимого разных файлов в один объект, который можно охарактеризовать как логическую структуру. Благодаря тому, что документ не ограничен одним файлом, XML позволяет создавать документ из частей, которые могут располагаться где угодно.

Документ XML обычно содержит следующие разделы:

• XML-декларация (Объявление XML);

• Пролог;

• Элементы;

• Атрибуты;

• Комментарии.

Символы разметки

Разметка всегда начинается символом < и заканчивается символом >.

Вместе с символами < и > (угловые скобки) специальную роль отведена также символу & (амперсанд). Границы элементов, инструкций обработки и некоторых других последовательностей обозначаются угловыми скобками. Амперсанд же помогает при замене текста при помощи сущностей.

Объявление XML (Декларация XML-документа)

При объявлении XML указывается версия языка, использованная для написания документа. Спецификация предписывает начинать документ с объявления XML, потому что от версии языка зависит трактование содержимого документа.

И если в первой (1.0) версии языка объявление было не обязательным, то в новых версиях это обязательно, и если объявление отсутствует, то под этим понимается версия 1.0. В объявлении также может содержаться информация о кодировке документа.

Пример:

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

Синтаксические правила для декларации XML

- Декларация XML чувствительна к регистру и должна начинаться с « <? Xml> », где « xml » пишется строчными буквами.
- Декларация XML строго должна быть первым утверждением в документе XML.
- Протокол HTTP может переопределить значение *кодировки,* которое вы указали в объявлении XML.

Пролог XML-документа

Прологом называются данные, расположенные после открывающего тега документа или после корневого элемента. Он включает сведения, относящиеся к документу в целом — кодировка символов, структура документа, таблицы стилей.

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="book.xsl"?>
<!DOCTYPE book SYSTEM "schema.dtd">
<!--Some comments-->
```

В языке XML есть возможность включения в документ инструкций, которые несут определенную информацию для приложений, которые будут обрабатывать тот или иной документ. Инструкции по обработке в XML создаются следующим образом.

<?Приложение Содержимое?>

В XML инструкции по обработке заключаются в угловые кавычки со знаком вопроса. В первой части процессинговой инструкции определяется приложение или система, которой предназначена вторая часть этой инструкции или ее содержимое. При этом инструкции по обработке действительны только для тех

приложений, которым они адресованы. Примером процессинговой инструкции может быть следующая инструкция:

<?serv cache-document?>

Теги

Тег (tag) — конструкция разметки, которая содержит имя элемента. Теги могут быть открывающимися (начальными) и закрывающимися (конечными). Также существуют теги пустого элемента, которые совмещают в себе открывающийся и закрывающийся.

Примеры:

- Начальный тег: <tag1>
- Конечный тег: </tag1>
- Тег пустого элемента: <empty_tag1 />

XML – Элементы

Элементы XML могут быть определены как строительные блоки XML. Элементы могут вести себя как контейнеры для хранения текста, элементов, атрибутов, объектов мультимедиа или всего этого.

Каждый документ XML содержит один или несколько элементов, область действия которых ограничена начальным и конечным тегами или для пустых элементов – тегом пустого элемента.

Синтаксис

Ниже приведен синтаксис для написания элемента XML:

<element-name attribute1 attribute2>

....content

</element-name>

где,

- **element-name** – это имя элемента. *Имя* его регистра в начальных и конечных тегах должно совпадать.
- **attribute1, attribute2** – это атрибуты элемента, разделенные пробелами. Атрибут определяет свойство элемента. Он связывает имя со значением, которое представляет собой строку символов. Атрибут записывается как –

name = "value"

За именем следует знак = и строковое *значение* в двойных (“”) или одинарных (’) кавычках.

Пустой элемент

Пустой элемент (элемент без содержимого) имеет следующий синтаксис –

<name attribute1 attribute2.../>

Ниже приведен пример документа XML с использованием различных элементов XML:

<?xml version = "1.0"?>

<contact-info>

<address category = "residence">

<name>Tanmay Patil</name>

<company>TutorialsPoint</company>

<phone>(011) 123-4567</phone>

</address>

</contact-info>

Вложенность элементов (Nesting of Elements) − XML-элемент может содержать несколько XML-элементов в качестве своих дочерних элементов, но дочерние элементы не должны перекрываться. т.е. Конечный тег элемента должен иметь то же имя, что и у самого последнего несогласованного начального тега.

В следующем примере показаны неправильные вложенные теги −

<?xml version = "1.0"?>

<contact-info>

<company>TutorialsPoint

</contact-info>

</company>

В следующем примере показаны правильные вложенные теги −

<?xml version = "1.0"?>

<contact-info>

<company>TutorialsPoint</company>

<contact-info>

Корневой элемент − документ XML может иметь только один корневой элемент. Например, ниже приведен неправильный XML-документ, поскольку оба элемента x и y находятся на верхнем уровне без корневого элемента −

<x>...</x>

<y>...</y>

В следующем примере показан правильно сформированный XML−документ -

<root>

<x>...</x>

<y>...</y>

</root>

Чувствительность к регистру − имена XML-элементов чувствительны к регистру. Это означает, что имя начального и конечного элементов должно быть точно в одном и том же регистре.

Правила XML-элементов

Следующие правила должны соблюдаться для элементов XML –

- *Имя* элемента может содержать любые буквенно-цифровые символы. Единственные знаки препинания, допускаемые в именах, – это дефис (-), знак ниже (_) и точка (.).
- Имена чувствительны к регистру. Например, Адрес, адрес и АДРЕС – это разные имена.
- Начальный и конечный теги элемента должны быть идентичны.
- Элемент, который является контейнером, может содержать текст или элементы, как видно из приведенного выше примера.

Атрибуты

Еще одной частью элементов XML являются атрибуты. У элемента может быть несколько уникальных атрибутов. Атрибуты позволяют нам указать больше

информации об элементе. Правильнее сказать, что атрибуты определяют свойства элементов.

Атрибут всегда является парой имя=значение:

name = "value"

Пример атрибута в теге:

<tag1 name = "value">element</tag1>

Значение атрибута должно быть в двойных ("") или в одинарных кавычках ('). Атрибуты используются только в начальном теге и теге пустого элемента.

Синтаксис

Атрибут XML имеет следующий синтаксис –

<element-name attribute1 attribute2 >

....content..

< /element-name>

где *attribute1* и *attribute2* имеет следующую форму –

name = "value"

значение должно быть в двойных (“”) или одинарных (’) кавычках. Здесь *attribute1* и *attribute2* являются уникальными метками атрибутов.

Атрибуты используются для добавления уникальной метки к элементу, размещения метки в категории, добавления логического флага или иного связывания его с какой-либо строкой данных. Следующий пример демонстрирует использование атрибутов –

<?xml version = "1.0" encoding = "UTF-8"?>

<!DOCTYPE garden [

<!ELEMENT garden (plants)*>

<!ELEMENT plants (#PCDATA)>

<!ATTLIST plants category CDATA #REQUIRED>

]>

<garden>

<plants category = "flowers" />

<plants category = "shrubs">

</plants>

</garden>

Атрибуты используются для различения элементов с одинаковыми именами, когда вы не хотите создавать новый элемент для каждой ситуации. Следовательно, использование атрибута может добавить немного больше деталей при дифференциации двух или более похожих элементов.

В приведенном выше примере мы классифицировали растения, включая категорию атрибутов и присваивая различные значения каждому из элементов. Следовательно, у нас есть две категории *растений* , один *цветы* и другой *цвет* . Таким образом, у нас есть два растительных элемента с разными атрибутами.

Вы также можете заметить, что мы объявили этот атрибут в начале XML.

Типы атрибутов

В следующей таблице перечислены типы атрибутов –

Тип атрибута	Описание
StringType	Он принимает любую буквенную строку в качестве значения. CDATA – это тип StringType. CDATA – это символьные данные. Это означает, что любая строка символов без разметки является законной частью атрибута.
TokenizedType	Это более ограниченный тип. Ограничения достоверности, отмеченные в грамматике, применяются после нормализации значения атрибута. Атрибуты TokenizedType представлены как –ID – используется для указания элемента как уникального.IDREF – используется для ссылки на идентификатор, который был назван для другого элемента.IDREFS – используется для ссылки на все идентификаторы элемента.ENTITY – указывает, что атрибут будет представлять внешнюю сущность в документе.ENTITIES – указывает, что атрибут будет представлять внешние объекты в документе.NMTOKEN – это похоже на CDATA с ограничениями на то, какие данные могут быть частью атрибута.NMTOKENS – Это похоже на CDATA с ограничениями на то, какие данные могут быть частью атрибута.
EnumeratedType	У этого есть список предопределенных значений в его объявлении. из которых он должен назначить одно

	значение. Существует два типа перечисляемых атрибутов:NotationType – объявляет, что элемент будет ссылаться на NOTATION, объявленный где-то еще в документе XML.Перечисление – Перечисление позволяет вам определить определенный список значений, которым должно соответствовать значение атрибута.

Это более ограниченный тип. Ограничения достоверности, отмеченные в грамматике, применяются после нормализации значения атрибута. Атрибуты TokenizedType представлены как –

ID – используется для указания элемента как уникального.

IDREF – используется для ссылки на идентификатор, который был назван для другого элемента.

IDREFS – используется для ссылки на все идентификаторы элемента.

ENTITY – указывает, что атрибут будет представлять внешнюю сущность в документе.

ENTITIES – указывает, что атрибут будет представлять внешние объекты в документе.

NMTOKEN – это похоже на CDATA с ограничениями на то, какие данные могут быть частью атрибута.

NMTOKENS – Это похоже на CDATA с ограничениями на то, какие данные могут быть частью атрибута.

У этого есть список предопределенных значений в его объявлении. из которых он должен назначить одно значение. Существует два типа перечисляемых атрибутов:

NotationType – объявляет, что элемент будет ссылаться на NOTATION, объявленный где-то еще в документе XML.

Перечисление – Перечисление позволяет вам определить определенный список значений, которым должно соответствовать значение атрибута.

Правила атрибутов элемента

Ниже приведены правила, которые необходимо соблюдать для атрибутов:

- Имя атрибута не должно появляться более одного раза в одном и том же начальном теге или теге пустого элемента.

- Атрибут должен быть объявлен в определении типа документа (DTD) с помощью объявления списка атрибутов.
- Значения атрибутов не должны содержать прямых или косвенных ссылок на сущности для внешних сущностей.
- Текст замены любого объекта, на который прямо или косвенно ссылается значение атрибута, не должен содержать знак «меньше» (**<**)

Кодировка пяти служебных символов (<,>,’,”,&)

Очевидно, что символы <, > и & не могут быть использованы в символьных данных и в значениях атрибутов в таком виде. Для их представления нужно использовать специальные последовательности. Также специальные последовательности используются при написании апострофов и кавычек внутри значений атрибутов:

Символ	Замена
<	<
>	>
&	&
'	'
"	"

Кроме того, для написания символа \ необходимо использовать \\.

Секция CDATA

Секция CDATA не является логической единицей текста. Эта секция может встречаться в документе в том месте, где синтаксис позволяет нам размещать символьные данные.

Секция начинается с <![CDATA[и завершается]]>. Между этой разметкой размещаются символьные данные, и при этом символы <, > и & могут быть

использованы в их непосредственной форме.

Комментарии

Комментарии (comment) не относятся к символьным данным документа. Комментарий начинается с <!-- и заканчивается -->, внутри комментария не может быть использована последовательность символов --. Также внутри комментария символ амперсанда не используется в качестве разметки.

Пример:

<!-- это комментарий >

Любой текст между символами **<! –** и **->** считается комментарием.

Правила XML-комментариев

Следующие правила должны соблюдаться для XML-комментариев –

- Комментарии не могут появляться до объявления XML.
- Комментарии могут появляться в любом месте документа.
- Комментарии не должны появляться в значениях атрибутов.
- Комментарии не могут быть вложены в другие комментарии.

Имена

В языке XML все имена могут содержать только буквы, входящие в секцию букв кодировки Unicode, цифры (арабские), точки, двоеточия, дефисы и знаки подчеркивания. Начинаться имена могут с буквы, двоеточия или символа подчеркивания. Обрати внимание, что строка XML в любом регистре не может быть началом для имени.

Пример

Посмотрим на Java-класс и объект этого класса. Потом попробуем сериализовать объект в формат XML. Код класса:

public class Book {

private String title;

private String author;

private Integer pageCount;

```
private List<String> chapters;
public Book(String title, String author, Integer pageCount, List<String> chapters) {
this.title = title;
this.author = author;
this.pageCount = pageCount;
this.chapters = chapters;
}
// геттеры/сеттеры
}
```

и создание объекта:

```
Book book = new Book("My Favorite Book", "Amigo", 999, Arrays.asList("Chapter 1", "Chapter 2", "Chapter 3", "Chapter 4", "Chapter 5", "Chapter 6"));
```

Вот так выглядит пример валидного XML-представления Java-объекта, который содержит 4 поля, одно из которых представляет собой коллекцию (Java-код выше):

```
<Book>
<title>My Favorite Book</title>
<author>Amigo</author>
<pageCount>999</pageCount>
<chapters>
<chapters>Chapter 1</chapters>
<chapters>Chapter 2</chapters>
<chapters>Chapter 3</chapters>
<chapters>Chapter 4</chapters>
<chapters>Chapter 5</chapters>
<chapters>Chapter 6</chapters>
</chapters>
</Book>
```