ЛАБОРАТОРНАЯ РАБОТА 12

**Описание схемы XML-документа на языке XSD**

*Цель работы:*формирование практических умений описа­ния структуры XML-документа на языке XSD.

**Теоретические сведения**

Язык XML Schema предоставляет набор встроенных типов дан­ных, которые могут использоваться разработчиками для определения содержимого документа (например: *float*, *string,boolean* и т.д.). Описание возможных стандартных типов приведено в прил. 1.

Кроме предоставления встроенных типов, в XML Schema, как и во многих языках программирования, разработчикам раз­решено определять свои собственные типы, обычно называемые *определенными пользователем типами* (UDT). При описании UDT можно также *определить* для них и пространство имен, чтобы не путать их с другими UDT, случайно использующими такое же имя.

Имена, определенные в схеме, принадлежат так называе­мому *целевому пространству имен*. Само по себе пространство имен является фиксированным, произвольным именем, которое должно соответствовать синтаксису URL (например,

*http://example.org/publishing*). В то же время, объявление про­странства имен не ссылается на файл с описанием схемы, а лишь указывает его имя.

Определения и объявления в схеме могут ссылаться также на имена, которые принадлежат другим пространствам имен, на­зываемым *исходными*. В каждой схеме может быть определено одно целевое пространство имен и возможно использование множества исходных пространств имен.

Рассмотрим особенности объявления и использования пространств имен на основе упрощенного шаблона XML-схемы (пример 3.1).

**Пример 3.1.**

*<xsd:schematargetNamespace="http://example.org/publishing" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:tns="http://example.org/publishing"><!--определениетипов--><xsd:simpleType name="AuthorType"><!--определение области значений типа--> ...</xsd:simpleType> <!--глобальное объявление элементов и их атрибутов--> <xsd:element name="author" type="tns:AuthorType"/> <xsd:element name="year" type="xsd: gYear"/> ...*<*/xsd:schema*>

Элемент ***xsd:schema*** определяет содержимое пространства имен, а атрибут ***targetNamespace*** – его имя. Таким образом, целе­вым пространством имен является *http://example.org/publishing*, оно содержит имена *AuthorType, year*и*author*. Имена *schema, element, simpleType*и*gYear* принадлежат стандартному исходному пространству имен

*http://www.w3.org/2001/XMLSchema*.

Имена пространства имен сокращаются при помощи конст­рукции с зарезервированным словом *xmlns*. Так, в примере 3.1 целевому пространству имен назначается псевдоним *tns*, а исходному – *xsd*. Затем в документе указание конкретного имени предваряется псевдонимом пространства имен, к которому оно принадлежит (например, *tns:AuthorType*).

*Определение элемента* заключается в определении его имени и модели (типа) контента.

Шаблон определения элемента выглядит следующим образом:

*<xsd:element name="…" type="…" [minOccurs="…"] [maxOccurs="…"] [default="…"| fixed="…"]>*

Обязательными являются только параметры *name*и *type*, определяющие соответственно имя и тип элемента. Параметры *minOccurs* и *maxOccurs*определяют минимальное или макси­мальное количество вхождений элемента. Например, если эле­мент является обязательным, значение *minOccurs* больше или равно 1. Параметр *maxOccurs*может кроме числового значения принимать также значение *unbounded*, что указывает на отсутствие ограничения максимального числа появлений. Значение по умолчанию для *minOccurs* и для *maxOccurs* равно 1.

*Атрибут элемента* определяется по следующему шаблону:

*<xsd:attribute name="…" type="…" [use="…"] [default="…"| fixed="…"]>*

Атрибуты, в отличие от элементов, могут появиться только однажды или ни разу. Поэтому и синтаксис для определения по­явления атрибутов отличается от синтаксиса для определения числа появлений элементов. В частности, атрибуты могут быть объявлены с параметром *use*. В зависимости от значения этого параметра атрибут обязателен (*use="required"*), необязателен (*use="optional"*) или запрещен (*use="prohibited"*).

Значения по умолчанию и атрибутов, и элементов могут быть объявлены с использованием параметра *default*, хотя этот параметр в том или ином случае работает по-разному. Атрибут со значением, определенным по умолчанию, может появляться или не появляться в документе. Если атрибут не появляется в документе, то обработчик схемы обеспечивает атрибут со значе­нием, равным значению *default*. Таким образом, значения по умолчанию для атрибутов имеют смысл, только если сами атри­буты являются необязательными, поэтому будет ошибкой опре­делить значение по умолчанию вместе с параметром *use,* отлич­ным от *use="optional"*.

Значение по умолчанию для элементов обрабатывается не­много по-другому. Если элемент появляется в документе, но не содержит какого-либо значения, то в качестве его значения под­ставляется значение по умолчанию. Однако, если элемент не по­является в документе, то обработчик схемы не обеспечивает его значения вообще. В общем, различия между значениями по умолчанию элемента и атрибута в следующем: заданное по умолчанию значение атрибута применяется тогда, когда атрибут отсутствует, а заданное по умолчанию значение элемента при­меняются тогда, когда элемент присутствует в документе, но не имеет значения (пуст).

Атрибут *fixed*используется в объявлениях и атрибутов, и элементов. Он используется, чтобы указать, что атрибут или элемент принимают фиксированные значения (независимо от их наличия в документе). Обратите внимание, что понятия «фикси­рованное значение» и «значение по умолчанию» являются взаимоисключающими, поэтому объявление не может одновременно содержать атрибуты *fixed*и *default*.

Тип элемента может быть *простым* или *комплексным*(*сложным*). Элемент простого типа не может содержать другие элементы или атрибуты. Комплексный тип может использоваться для встраивания элементов в другие элементы или ассоциирования атрибутов с элементом.

*Предопределенный простой тип* ограничивает значения по их базовому типу и создается с помощью элемента*simpleType*. Для ограничения диапазона значений используется вложенный элемент *restriction*. Он позволяет определить, как именно нужно ограничивать базовый тип, сужая один или более из его *аспек-тов* (*facets*).

Например, чтобы определить ограничения для элемента, со­держащего год издания книги или журнала, может использовать-ся следующий код:

*<xsd:simpleType name="YearType"> <xsd:restriction base="xsd:gYear"> <xsd:minInclusive value="1900"/> <xsd:maxInclusive value="2012"/> </xsd:restriction> </xsd:simpleType>*

Таким образом, допустимые значения года издания нахо­дятся в интервале от 1900 до 2012.

Возможные аспекты базового типа XSD представлены в прил. 2.

Большинство аспектов неприменимы ко всем типам (некоторые имеют смысл только в определенных типах). Элемент *pattern* применяется, если ограничением выступает регулярное выражение.

Например, чтобы создать ограничение для элемента, содер­жащего информацию об авторе, нужно определить следующий простой тип:

*<xsd:simpleType name="AuthorType"> <restriction base=" xsd:string"> <pattern value="^[а-яА-Я. ]+ "/> </restriction> </xsd:simpleType>*

Базовым является тип *string*(строка), ограничение по базовому типу создается с использованием регулярных выражений. В примере информация об авторе может содержать один или более символов, включающих строчные или прописные буквы русского алфавита, точку, а также пробел. Символ "^" указывает на начало строки.

Развернутое описание синтаксиса регулярных выражений приведено в прил. 3.

*Комплексный* (*составной*) *тип* определяется для элементов, содержащих вложенные элементы, и элементов с атрибутами.

Чтобы определить новый комплексный тип в целевом про­странстве имен схемы, используется элемент ***xsd:complexType*:**

*<xsd:complexTypename="…"><!--описание модели содержимого комплексного типа --></xsd:complexType>*

Описание модели содержимого комплексного типа обычно включает объявления элементов, ссылки на элементы (<*xsd:ele-mentref="имя элемента, объявленного ранее"*>) и объявления атрибутов.

При описании модели содержимого также используются конструкции ***xsd:sequence*** *,****xsd:choice*** и ***xsd:all*, называемые со­ставителями. Контейнер** *xsd:sequence* определяет порядок указа­ния дочерних элементов, контейнер *xsd:choice* указывает на необходимость выбора только одного дочернего элемента из указанных, а контейнер *xsd:all* предписывает, что дочерние элементы могут размещаться в произвольном порядке. Описания атрибутов помещаются не в составителе, а после него в конце определения составного типа.

В примере 3.2 приведена возможная XSL-схема для доку­мента из примера 1.1.

**Пример 3.2 (***sch.xsd***)**

*<xsd:schema targetNamespace="http://example.org/publishing" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:tns="http://example.org/publishing">*

*<xsd:element name="documents" type="tns: DocsType">*

*<xsd:element name="author" type="tns:AuthorType"/><xsd:element name="title" type="xsd:string"/><xsd:element name="year\_publication" type="tns:YearType "/>*

*<xsd:complexType name="DocsType">*

*<xsd:sequence>*

*<xsd:element name="books"/>*

*<xsd:complexType>*

*<xsd:element name="book" type="tns: BookType">*

*</xsd:complexType>*

*<xsd:element name="articles"/>*

*<xsd:complexType>*

*<xsd:element name="article" type="tns: ArticleType"/>*

*</xsd:complexType>*

*</xsd: sequence>*

*</xsd:complexType>*

*<xsd:complexType name="BookType">*

*<xsd:sequence>*

*<xsd:element ref="author"/>*

*<xsd:element ref="title"/>*

*<xsd:element name="translate" type="tns:AuthorType"/>*

*<xsd:element ref=" year\_publication"/>*

*</xsd: sequence>*

*<xsd:attribute name="id" type="xsd:string" use="required"/>*

*</xsd:complexType>*

*<xsd:complexType name="ArticleType">*

*<xsd:sequence>*

*<xsd:element ref="author"/>*

*<xsd:element ref="title"/>*

*<xsd:element ref=" year\_publication"/>*

*</xsd: sequence>*

*<xsd:attribute name="id" type="xsd:string" use="required"/>*

*</xsd:complexType>*

*<xsd:simpleType name="AuthorType" ><restriction base="xsd:string"> <pattern value="^[а-яА-Я. ]+ "/></restriction></xsd:simpleType><xsd:simpleType name="YearType"><xsd:restriction base="xsd:gYear"> <xsd:minInclusive value="1900"/> <xsd:maxInclusive value="2012"/></xsd:restriction></xsd:simpleType></xsd:schema >*В начале документа определяется корневой элемент и эле­менты, на которые далее будет выполняться ссылка. Затем по порядку определяются комплексные и простые типы элементов.Если элементы документа не принадлежат никакому про­странству имен и записаны без префикса, то для связи доку­мента XML со схемой XSD в корневом элементе документа за­писывается атрибут *noNamespaceSchemaLocation*:*<documents xmlns:xsi="http://www.w3.org/2001/XMLSchema" xsi:noNamespaceSchemaLocation="sch.xsd">*Если же элементы документа относятся к некоторому про­странству имен, то применяется атрибут *schemaLocation*, в кото­ром через пробел перечисляются пространство имен и располо­жение файла со схемой, описывающей это пространство имен:*<documents xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"xsi: schemaLocation= "http://some.firm.com /someNames A.xsd http://some.firm.com /anotherNames B.xsd” xmlns:pr1= http://some.firm.com/someNames xmlns:pr2= “http://some.firm.com /anotherNames">*Значением атрибута *xsi:schemaLocation* является список пар имен пространства имен и размещения URI, разделенных про­белами, который показывает список, где искать определенный файл схемы. Таким образом, в документе можно использовать имена, определенные в схемах *A.xsd* и *B.xsd*, снабжая их префиксами *pr1* и *pr2* соответственно.

**Задание**

Разработайте XSD-схему XML-документа, созданного в лабораторной работе 10, которая должна включать:

- описание элементов;

- описание атрибутов элементов;

- определение типа элементов;

- определение последовательности использования эле­ментов.

**Контрольные вопросы**

1. В чем отличие XSD от DTD?

2. Как определить эле­мент в XSD?

3. Как определить атри­бут элемента в XSD?

4. Как подключить XSL-схему к XML-документу?

5. В каких случаях исполь­зуется сложный (комплексный) тип данных?