Школа

XML схемы

Добро пожаловать в школу XML-схем

Школа схем XML

В школе схем XML вы узнаете что такое XML-схема. Вы узнаете как XML-схема заменяет DTD и как применять язык XML-схем в своих приложениях. Изучайте схемы XML!

Описание элементов схем XML[1] (open link)

Здесь вы найдете список допустимых типов данных и элементов XML-схем.

Ресурсы по схемам XML[2] (open link)

Список ссылок на другие ценные ресурсы Интернет по XML-схемам.

Содержание

Введение в схемы XML[3] (open link)

В введении в язык схем XML рассказывается о том, что такое XML-схема.

XML-схемы: почему?[4] (open link)

В этом разделе объясняется, почему XML-схемы могут применяться вместо DTD.

XML-схемы: как?[5] (open link)

.

Элементы XML-схем[6] (open link)

В этом разделе описываются элементы XML-схем и раскрывается разница между простыми и сложными элементами.

Простые типы элементов XML-схем[7] (open link)

В этом разделе объясняются простые типы элементов XML-схем.

Грани XML-схем[8] (open link)

В этом разделе рассказывается, как добавлять ограничения в XML-элементы.

Сложные типы элементов XML-схем[9] (open link)

В этом разделе объясняются сложные типы элементов XML-схем.

1: http://xml.nsu.ru/schema/schema_elements_ref.xml
2: http://xml.nsu.ru/schema/schema_resources.xml
3: http://xml.nsu.ru/schema/schema_intro.xml
4: http://xml.nsu.ru/schema/schema_why.xml
5: http://xml.nsu.ru/schema/schema_howto.xml
6: http://xml.nsu.ru/schema/schema_elements.xml
7: http://xml.nsu.ru/schema/schema_simple.xml
8: http://xml.nsu.ru/schema/schema_facets.xml
9: http://xml.nsu.ru/schema/schema_complex.xml

Пример XML-схемы[10] (open link)

Простой пример фактуры отгрузки и соответствующей ей XML-схемы.

Описание элементов схем XML

Элементы XML-схем[11] (open link)

Список допустимых элементов XML-схем.

Типы данных[12] (open link)

Список примитивных типов данных XML-схем.

Типы данных: даты[13] (open link)

Список примитивных типов данных XML-схем для представления дат.

Примеры XML-схем[14] (open link)

Изучайте XML-схемы на примерах.

Ресурсы по схемам XML

Ресурсы по схемам XML в Интернете[15] (open link)

Несколько ссылок на другие ценные ресурсы Интернет по XML-схемам.

Введение в схемы XML

XML-схема - это основанная на XML альтернатива DTD
XML-схема описывает структуру XML-документа

Что вы должны уже знать

Прежде, чем вы начнете изучать язык XML-схем, вы должны иметь общее представление о XML и о пространствах имен XML. Если вы хотите сначала изучить эти предметы, посетите нашу школу XML:[16] (open link)

Кроме того, неплохо иметь некоторое знание DTD:[17] (open link)

Что такое XML-схема?

XML-схема:

10: http://xml.nsu.ru/schema/schema_example.xml
11: http://xml.nsu.ru/schema/schema_elements_ref.xml
12: http://xml.nsu.ru/schema/schema_datatypes.xml
13: http://xml.nsu.ru/schema/schema_datetypes.xml
14: http://xml.nsu.ru/schema/schema_examples.xml
15: http://www.w3schools.com/schema/schema_resources.asp
16: http://xml.nsu.ru/xml/xml_home.xml
17: http://xml.nsu.ru/dtd/dtd_home.xml

- Определяет элементы, которые могут появляться в документе
- Определяет атрибуты, которые могут появляться в документе
- Определяет, какие элементы являются дочерними
- Определяет последовательность, в которой появляются дочерние элементы
- Определяет число дочерних элементов
- Определяет пустой ли элемент или он может включать в себя текст
- Определяет типы данных элементов и атрибутов
- Определяет значения атрибутов по умолчанию

Схемы XML идут на смену определениям DTD

Мы считаем, что очень скоро XML-схемы будут применяться в Web-приложениях вместо DTD-таблиц, и вот почему:

- XML-схемам легче научиться, чем DTD
- XML-схемы можно в будущем расширить, если понадобится какое-либо добавление
- XML-схемы богаче и полезнее, чем DTD
- XML-схемы написаны на XML
- XML-схемы поддерживают типизацию данных
- XML-схемы поддерживают пространства имен

XML-схемы теперь являются рекомендацией W3C

Вначале XML-схемы были предложены компанией Microsoft, но 2 мая 2001 года консорциум W3C выдвинул XML-схемы в качестве своей официальной рекомендации

Спецификация была пересмотрена членами W3C и теперь закреплена.

Более полную информацию о деятельности и статусе W3C вы можете получить в нашей школе W3C School[18] (open link)

XML-схемы: почему?

Есть несколько причин, по которым XML-схемы предпочтительней определений DTD

XML-схемы поддерживают типизацию данных

Одно из самых серьезных преимуществ XML-схем состоит в том, что они поддерживают типизацию данных.

Благодаря этому:

- Легче описывать разрешенное содержание документа
- Легче проверять правильность данных
- Легче работать с данными из баз данных
- Легче задавать фасеты данных (ограничения на данные)
- Легче задавать паттерны данных (форматы данных)
- Легче преобразовывать данные различных типов

XML-схемы используют синтаксис XML

18: http://www.w3schools.com/w3c/

Еще один существенный плюс XML-схем состоит в том, что они пишутся на XML.

Благодаря этому:

- Вам не нужно изучать какой-то еще язык
- Для редактирования схем можно использовать XML-редактор
- Для анализирования схем можно использовать XML-парсер
- Можно работать с XML-схемами посредством XML DOM
- Можно преобразовывать схемы посредством XSLT

XML-схемы увеличивают достоверность обмена данными

При отправке данных от отправителя получателю, очень важно, чтобы оба имели одинаковые "ожидания" относительно содержания.

С помощью XML-схем отправитель может описать то, каким образом получатель должен эти данные понимать.

Например, дата 1999-03-11 в некоторых странах может быть интерпретирована как 3 ноября, а в других - как 11 марта. XML-элемент, содержащий описание типа данных, например:

```
<date type="date">1999-03-11</date>
```

обеспечит верную трактовку содержания, поскольку тип данных date требует использования формата CCYY-MM-DD.

XML-схемы расширяемы

XML-схемы расширяемы, также, как и XML, поскольку они пишутся на XML.

Благодаря расширяемости каждой конкретной схемы:

- Встраивать одни схемы в другие
- Создавать свои собственные типы данных, производя их из стандартных типов
- Ссылаться из документа на несколько схем

Правильности не достаточно

Правильный XML-документ - это документ, который удовлетворяет синтаксическим правилам XML:

- Он должен начинаться с XML-декларации
- Он должен иметь один-единственный уникальный корневой элемент
- Каждому начальному тэгу должен соответствовать конечный тэг
- Тэги XML зависят от регистра
- Все элементы должны быть закрыты
- Все элементы должны бытьправильно вложены друг в друга
- Все значения атрибутов должны быть заключены в кавычки
- Вместо специальных символов должны применяться сущности XML

Даже если XML-документ не содержит синтаксических ошибок, он может содержать ошибки и эти ошибки могут привести к серьезным последствиям. Представьте себе ситуацию: вы сделали заказ на пять дюжин лазерных принтеров, а не на просто пять лазерных принтеров. При использовании XML-схем большая часть подобных ошибок может быть отслежена вашим валидационным программным обеспечением.

XML-схемы: как?

XML-документы могут содержать ссылку на определение DTD или на XML-схему

Простой XML-документ

Этот простой XML-документ (note.xml) позаимствован из нашей школы XML.

```
<?xml version="1.0"?>
<note>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend!</body>
</note>
```

В этом примере элемент note является сложным элементом, поскольку он содержит другие элементы. Все другие элементы - простые, поскольку не содержат внутри других элементов.

Простое определение DTD

А вот простой файл DTD (note.dtd из нашей школы DTD), он задает элементы приведенного выше XML-документа:

```
<!ELEMENT note (to, from, heading, body)>
<!ELEMENT to (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT heading (#PCDATA)>
<!ELEMENT body (#PCDATA)>
```

Первая строка вэтом определении DTD задает элемент note как содержащий в себе четыре других элемента: "to, from, heading, body".

Строки со 2 по 5 задают: элемент to имеет тип "#PCDATA", элемент from имеет тип "#PCDATA" и т.д.

Простая XML-схема

Эта простая XML-схема (note.xsd) тоже задает элементы приведенного выше XML-файла:

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name="note">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="to" type="xs:string"/>
        <xs:element name="from" type="xs:string"/>
        <xs:element name="heading" type="xs:string"/>
        <xs:element name="body" type="xs:string"/>
      </xs:sequence>
    </xs:complexType>
</xs:element>
```

```
</xs:schema>
```

Схема задает: элемент note является сложным элементом, содержащим последовательность других простых элементов.

Ссылка на определение DTD

В этом XML-документе имеется ссылка на определение DTD:

```
<?xml version="1.0"?>
<!DOCTYPE note SYSTEM
"http://www.w3schools.com/dtd/note.dtd">
<note>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend!</body>
</note>
```

Ссылка на XML-схему

В этом XML-документе имеется ссылка на XML-схему:

```
<?xml version="1.0"?>

<note xmlns:xsi=
"http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation=
"http://www.w3schools.com/schema/note.xsd">

<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend!</body>
</note>
```

Элементы XML-схем

XML-схемы задают элементы ваших XML-файлов

Простые и сложные элементы

```
<?xml version="1.0"?>
<note>
<to>Tove</to>
<from>Jani</from>
<heading>Reminder</heading>
<body>Don't forget me this weekend!</body>
</note>
```

В этом примере элемент note является сложным элементом, поскольку он содержит другие элементы. Все другие элементы - простые, поскольку не содержат внутри других элементов.

Определение простых элементов

Простые элементы определяются вот так:

```
Синтаксис:
<xs:element name="name" type="type"/>

Примеры:
<xs:element name="to" type="xs:string"/>
<xs:element name="from" type="xs:string"/>
<xs:element name="heading" type="xs:string"/>
<xs:element name="body" type="xs:string"/>
```

Определение сложных элементов

Сложные элементы определяются вот так:

```
Синтаксис:
<xs:element name="name">
    <xs:complexType>
        .
        . содержимое элемента
        .
    </xs:complexType>
</xs:element>

Пример:
<xs:element name="note">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="to" type="xs:string"/>
        <xs:element name="from" type="xs:string"/>
        <xs:element name="heading" type="xs:string"/>
        <xs:element name="body" type="xs:string"/>
      </xs:sequence>
    </xs:complexType>
</xs:element>
```

Описание документа Note

Вот XML-схема, задающая документ Note:

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name="note">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="to" type="xs:string"/>
        <xs:element name="from" type="xs:string"/>
        <xs:element name="heading" type="xs:string"/>
        <xs:element name="body" type="xs:string"/>
```

```
      </xs:sequence>
    </xs:complexType>
</xs:element>

</xs:schema>
```

В этой схеме определяется, что элемент note является сложным элементом, состоящим из последовательности других простых элементов.

Ссылки на другие элементы

Эта XML-схема задает совершенно то же самое, что и приведенная выше:

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name="note">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="to"/>
      <xs:element ref="from"/>
      <xs:element ref="heading"/>
      <xs:element ref="body"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>

<xs:element name="to" type="xs:string"/>
<xs:element name="from" type="xs:string"/>
<xs:element name="heading" type="xs:string"/>
<xs:element name="body" type="xs:string"/>

</xs:schema>
```

Обратите внимание, что элемент note ссылается на свои под-элементы с помощью атрибута ref.

Простые типы элементов

Элементы простого типа могут содержать только текст

Что представляет собой элемент простого типа?

Элемент простого типа - это XML-элемент, который может содержать только текст, он не может содержать в себе другие элементы или атрибуты.

Однако, выражение "только текст" довольно неопределенно. Текст может быть различных типов. Он может быть одного из типов, включенных в определение XML-схем, а может принадлежать типу, который вы создали сами.

Также вы можете добавить к типу данных дополнительные ограничения, грани (facets), чтобы данные находились в определенных пределах, вы можете потребовать, чтобы формат данных соответствовал определенному паттерну.

Как определить элемент простого типа

Синтаксис определения элемента простого типа таков:

```
<xs:element name="xxx" type="yyy"/>
```

тут xxx - имя элемента, а yyy - его тип.

Вот некоторые XML-элементы:

```
<lastname>Refsnes</lastname>

<age>34</age>

<dateborn>1988-03-27</dateborn>
```

А вот соответствующие им определения элементов простого типа:

```
<xs:element name="lastname" type="xs:string"/>

<xs:element name="age" type="xs:number"/>

<xs:element name="dateborn" type="xs:date"/>
```

Общие типы данных XML-схем

XML-схемы имеют множество встроенных типов данных. Вот список наиболее употребительных из них:

- xs:string
- xs:decimal
- xs:integer
- xs:boolean
- xs:date
- xs:time

С полным списком встроенных в XML-схемы типов данных вы можете познакомиться в разделах "Элементы XML-схем".

Ограничения на тип содержимого

Когда в XML-документе заданы типы, это накладывает ограничение на содержимое элементов.

Если XML-элемент типа xs:date содержит строку "Hello Mother", этот элемент вызовет ошибку при валидации.

С помощью определений в XML-схемах вы можете добавлять свои собственные ограничения на содержимое XML-элементов. Эти ограничения называются гранями (facets). Вы больше узнаете о гранях в следующем разделе.

Грани XML-схем

Ограничения на содержимое XML-элементов называются гранями

Ограничения на значение элемента

В этом примере определяется элемент age, на его значение накладывается ограничение:

```
<xs:element name="age">

<xs:simpleType>
  <xs:restriction base="xs:integer">
    <xs:minInclusive value="16"/>
    <xs:maxInclusive value="34"/>
  </xs:restriction>
</xs:simpleType>

</xs:element>
```

Ограничения на значения из списка

Иногда необходимо ограничить содержание XML-элементов несколькими возможными фиксированными значениями.

В этом примере определяется элемент car:

```
<xs:element name="car">

<xs:simpleType>
  <xs:restriction base="xs:string">
    <xs:enumeration value="Audi"/>
    <xs:enumeration value="Mercedes"/>
    <xs:enumeration value="Volvo"/>
  </xs:restriction>
</xs:simpleType>

</xs:element>
```

Элемент car определяется как принадлежащий простому типу. На содержание этого элемента наложено ограничение, основывающееся на встроенном в XML-схемы типе данных string, значение элемента может иметь только несколько значений из приведенного списка: Audi, Mercedes, Volvo.

Этот пример можно записать и так:

```
<xs:element name="car" type="carType"/>

<xs:simpleType name="carType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="Audi"/>
    <xs:enumeration value="Mercedes"/>
    <xs:enumeration value="Volvo"/>
  </xs:restriction>
</xs:simpleType>
```

В этом случае тип "carType" может использоваться и другими элементами, потому что он не является частью элемента "car".

Сложные типы элементов

Сложные элементы могут содержать другие элементы и атрибуты

На этой неделе страница на реставрации

Пример XML-схемы

Пример фактуры отгрузки в формате XML

Помните каталог CD из школы XML? А вот фактура на компакт-диски:

```
<?xml version="1.0" encoding="ISO-8859-1" ?>

<shipOrder>
  <shipTo>
    <name>Tove Svendson</name>
    <street>Ragnhildvei 2</street>
    <address>4000 Stavanger</address>
    <country>Norway</country>
  </shipTo>
  <items>
    <item>
      <title>Empire Burlesque</title>
      <quantity>1</quantity>
      <price>10.90</price>
    </item>
    <item>
      <title>Hide your heart</title>
      <quantity>1</quantity>
      <price>9.90</price>
    </item>
  </items>
</shipOrder>
```

Эта фактура состоит из корневого элемента <shipOrder>, который имеет дочерние элементы <shipTo> и <items>. Элемент <items> содержит элементы <item>. Элемент <item> содержит элементы <title>,<quantity>, и <price>.

Пример XML-схемы

Вот XML-схема, которая определяет приведенную выше фактуру отгрузки:

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name="shipOrder" type="order"/>

<xs:complexType name="order">
  <xs:sequence>
```

```
    <xs:element name="shipTo" type="shipAddress"/>
    <xs:element name="items" type="cdItems"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="shipAddress">
  <xs:sequence>
    <xs:element name="name" type="xs:string"/>
    <xs:element name="street" type="xs:string"/>
    <xs:element name="address" type="xs:string"/>
    <xs:element name="country" type="xs:string"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="cdItems">
  <xs:sequence>
    <xs:element name="item" type="cdItem"
                maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="cdItem">
  <xs:sequence>
    <xs:element name="title" type="xs:string"/>
    <xs:element name="quantity" type="xs:positiveInteger"/>
    <xs:element name="price" type="xs:decimal"/>
  </xs:sequence>
</xs:complexType>

</xs:schema>
```

В это схеме определяется: элемент <shipOrder> имеет тип order. Order - это элемент сложного типа, состоящий из элементов <shipTo> и <items>. Элемент <shipTo> имеет тип shipAddress - сложный тип, состоящий из элементов <name>, <street>, <address>, и <country>. Элемент <items> имеет тип cdItems - сложный тип, состоящий из элементов <item>. Элемент <item> имеет тип cdItem - сложный тип элементов, состоящий из элементов <title>, <quantity>, и <price>. Элемент <title> - нормальный элемент, имеющий тип string.

Сложно? Почитайте следующие разделы и вы все поймете.

Если вы используете Internet Explorer 5.0 и выше, вы можете посмотреть на фактуру:[19] (open xml)

и на соответствующую XML-схему:[20] (open schema)

Описание элементов XML-схем

Элементы XML-схем

19: http://xml.nsu.ru/schema/order.xml
20: http://xml.nsu.ru/schema/order_schema.xml

Элемент	Описание
all	Определяет под-элементы в произвольном порядке. Дочерние элементы не обязательны, но могут появляться только по одному разу
annotation	Родительский элемент элементов-комментариев <appInfo> и <documentation>
any	Определяет любые под-элементы
anyAttribute	Определяет любые атрибуты
appInfo	Элемент-комментарий. Задает титул схемы
attribute	Определяет атрибут
attributeGroup	Определяет группу атрибутов
choice	Определяет выбор других элементов. Является аналогом оператора "|" в DTD
complexContent	Определяет ограничения или расширения контентной модели мложного типа
complexType	Определяет элемент сложного типа complexType
documentation	Элемент-комментарий. Предоставляет полезную информацию о схеме
element	Определяет элемент
extension	Определяет расширения элемента
field	Определяет поле. Использует XPath. Может применяться внутри элемента <unique> для определения полей
group	Определяет группу элементов
import	Задает импорт декларации типов из другой схемы
include	Задает включение другой схемы в существующее пространство имен
key	Задает элементы или атрибуты с ключом, указывающим на другой элемент
keyref	Задает элементы или атрибуты, на которые указывает ключ
list	Определяет элементы, которые могут содержать список значений
redefine	Задает переопределение уже определенных элементов
restriction	Задает ограничения элемента
schema	Задает корневой элемент схемы
selector	Задает селектор для отбора XML-элементов
sequence	Задает последовательность других элементов. Является аналогом оператора "," в DTD
simpleContent	Определяет контентную модель типа, который может содержать только символьные данные
simpleType	Определяет элемент простого типа simpleType
union	Определяет элементы или атрибуты, которые могут иметь множественные значения
unique	Определяет элементы или атрибуты, которые должны иметь уникальные значения

Грани XML-схем

Грани задают ограничения на содержание элементов.

Элемент	Описание
enumeration	Задает список значений
length	Задает длину
maxLength	Задает максимальную длину
minLength	Задает минимальную длину
maxExclusive	Задает максимальное значение
maxInclusive	Задает максимальное значение включительно
minExclusive	Задает минимальное значение
minInclusive	Задает минимальное значение включительно
fractionDigits	Задает число цифр в дроби
totalDigits	Задает число цифр
pattern	Задает паттерн содержимого элементов
whiteSpace	Задает значение пробелов в содержимом элементов

Атрибуты XML-схем

Атрибуты предоставляют дополнительную информацию об элементах

Атрибут	Описание
abstract	Задает элемент как имеющий абстрактный тип
attributeFormDefault	Задает квалификацию локальных атрибутов как глобально заданных
base	Задает базовый тип элемента
block	Задает запрещенное выведение ограничением (derivations-by-restriction)
blockDefault	Задает изначальное ограничение block на все определения типов.
default	Задает значение элемента или атрибута по умолчанию
elementFormDefault	Задает квалификацию локального элемента как глобально определенного
final	Задает запрещенное выведение ограничением (derivations-by-restriction)
finalDefault	Задает изначальное ограничение final на все определения типов
fixed	Задает фиксированное значение элемента или атрибута
form	Задает, что локально объявленные элементы определяются в конкретных экземплярах документов
itemType	Задает тип пунктов списка
memberTypes	Задает тип членов, использованных в союзе (union)
maxOccurs	Задает максимальное количество вхождений элемента

minOccurs	Задает минимальное количество вхождений элемента
mixed	Задает элемент как имеющий смешанный тип
name	Задает имя элемента или атрибута
namespace	Задает пространство имен элемента или атрибута
noNamespaceSchemaLocation	Задает местоположение документа-схемы, который не имеет результирующих пространств имен
nillable	Определяет, что элемент может иметь пустое значение NULL (nil)
processContents	Определяет, как валидатор схемы должен обрабатывать элемент
ref	Задает ссылку на глобально определенный элемент
schemaLocation	Определяет местоположение схемы
substitutionGroup	Определяет, что элементы заменяются другими элементами
targetNamespace	Задает результирующее пространство имен схемы
type	Задает тип элемента
use	Задает использование элемента (обязательный или нет)
value	Задает значение элемента схемы
xsi:nil	Задает реальное содержание пустого (NULL) элемента XML-документа
xsi:schemaLocation	Задает реальное местоположение элемента в XML-документе
xsi:type	Задает реальный тип элемента в XML-документе

XML-схемы: типы данных

В XML-схемах имеется несколько встроенных типов данных

Примитивные типы данных

Имя	Описание	Пример	Грани
string	Строка символов как последовательность 10646 символов Unicode или ISO/IEC, включая пробел, символ табуляции, возврат каретки и перевод строки	John Lennon	enumeration length maxLength minLength pattern whiteSpace
boolean	бинарные логические значения: true или false,1 или 0.	false	pattern whiteSpace
decimal	Десятичное число как последовательность десятичных цифр, разделенных периодом как десятичным разделителем	3145.56	enumeration fractionDigits maxExclusive maxInclusive minExclusive minInclusive pattern

			totalDigits whiteSpace
float	32-битное число с плавающей запятой, за мантиссой идет (не обязательно) экспонента	4.6E4	enumeration maxExclusive maxInclusive minExclusive minInclusive pattern whiteSpace
double	64-битное число с плавающей запятой, за мантиссой идет (не обязательно) экспонента	4.6E4	enumeration maxExclusive maxInclusive minExclusive minInclusive pattern whiteSpace
hexBinary	шестнадцатиричные данные в виде последовательности бинарных октетов	0FFF	enumeration length maxLength minLength pattern whiteSpace
base64Binary	Бинарные данные в кодировке base64 в виде последовательности бинарных октетов	GpM7	enumeration length maxLength minLength pattern whiteSpace
anyURI	Универсальный идентификатор ресурса (Uniform Resource Identifier) определенный в стандартах RFC 2396 и RFC 2732	http://cnet.com	enumeration length maxLength minLength pattern whiteSpace
QName	Пригодное XML-имя как определяется в пространствах имен XML	xs:element	enumeration length maxLength minLength pattern whiteSpace
NOTATION	Атрибут NOTATION как определяется в XML		enumeration length maxLength minLength pattern whiteSpace

XML-схемы: типы дат

В XML-схемах имеется несколько встроенных типов дат

Примитивные типы дат

Имя	Описание	Формат/Пример	Грани

duration	Продолжительность периода времени в виде значений лет, месяцев, дней, часов, минут и секунд (по § 5.5.3.2 стандарта ISO 8601)	**PnYnMnDTnHnMnS** P1M3DT4H (один месяц, 3 дня и 4 часа)	enumeration maxExclusive maxInclusive minExclusive minInclusive pattern whiteSpace
dateTime	Момент времени как комбинация календарной даты и времени (по § 5.4 стандарта ISO 8601)	**CCYY-MM-DD** **THH:MM:SS** 1950-03-26T15:30:01	enumeration maxExclusive maxInclusive minExclusive minInclusive pattern whiteSpace
time	Ежедневный момент времени как время суток (по § 5.3 стандарта ISO 8601)	**HH:MM:SS** 15:30:01	enumeration maxExclusive maxInclusive minExclusive minInclusive pattern whiteSpace
date	Календарная дата (по § 5.2.1 стандарта ISO 8601)	**CCYY-MM-DD** 1950-03-26	enumeration maxExclusive maxInclusive minExclusive minInclusive pattern whiteSpace
gYearMonth	Месяц года (по § 5.2.1 стандарта ISO 8601)	**CCYY-MM** 1950-03	enumeration maxExclusive maxInclusive minExclusive minInclusive pattern whiteSpace
gYear	Год (по § 5.2.1 стандарта ISO 8601)	**CCYY** 1950	enumeration maxExclusive maxInclusive minExclusive minInclusive pattern whiteSpace
gMonthDay	День и месяц в виде календарной даты (по § 3 стандарта ISO 8601)	**-MM-DD** -03-26	enumeration maxExclusive maxInclusive minExclusive minInclusive pattern whiteSpace
gDay	День месяца (по § 3 стандарта ISO 8601)	**-DD** -26	enumeration maxExclusive maxInclusive minExclusive minInclusive pattern whiteSpace
gMonth	Месяц (по § 3 стандарта ISO 8601)	**-MM** -03	enumeration maxExclusive

maxInclusive
minExclusive
minInclusive
pattern
whiteSpace

Примеры XML-схем

Изучайте XML-схемы на этих примерах

Элементы XML-схем

<xs:schema> <xs:element>

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name="note">
.
(здесь располагаются под-элементы)
.
</xs:element>

</xs:schema>
```

<xs:annotation> <xs:appInfo> <xs:documentation>

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:annotation>
  <xs:appInfo>W3Schools Note</xs:appInfo>
  <xs:documentation xml:lang="en">
      This Schema defines a W3Schools note
  </xs:documentation>
</xs:annotation>

<xs:element name="note">
.
(здесь располагаются под-элементы)
.
```

<xs:complexType> <xs:sequence>

```
<xs:element name="note">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="to" type="xs:string"/>
        <xs:element name="from" type="xs:string"/>
        <xs:element name="heading" type="xs:string"/>
        <xs:element name="body" type="xs:string"/>
      </xs:sequence>
    </xs:complexType>
</xs:element>
```

<xs:simpleType> <xs:restriction>

```
<xs:element name="age">
```

```
<xs:simpleType>
  <xs:restriction base="xs:integer">
  .
(здесь располагаются ограничения)
  .
  </xs:restriction>
</xs:simpleType>

</xs:element>
```

<xs:minInclusive> <xs:maxInclusive>

```
<xs:element name="age">

<xs:simpleType>
  <xs:restriction base="xs:integer">
    <xs:minInclusive value="16"/>
    <xs:maxInclusive value="34"/>
  </xs:restriction>
</xs:simpleType>

</xs:element>
```

<xs:minExclusive> <xs:maxExclusive>

```
<xs:element name="age">

<xs:simpleType>
  <xs:restriction base="xs:integer">
    <xs:minExclusive value="15"/>
    <xs:maxExclusive value="35"/>
  </xs:restriction>
</xs:simpleType>

</xs:element>
```

<xs:enumeration>

```
<xs:element name="car">

<xs:simpleType>
  <xs:restriction base="xs:string">
    <xs:enumeration value="Audi"/>
    <xs:enumeration value="Mercedes"/>
    <xs:enumeration value="Volvo"/>
  </xs:restriction>
</xs:simpleType>

</xs:element>
```

<xs:length> <xs:totalDigits> <xs:fractionDigits>

```
<xs:element name="salary">

<xs:simpleType>
  <xs:restriction base="xs:decimal">
    <xs:length value="6"/>
    <xs:totalDigits value="5"/>
```

```
      <xs:fractionDigits value="2"/>
    </xs:restriction>
  </xs:simpleType>

  </xs:element>
```

Атрибуты XML-схем

name, type, use

```
<xs:element name="lastname" type="xs:string" use="required"/>
<xs:element name="firstname" type="xs:string" use""required"/>
<xs:element name="dateborn" type="xs:date" use="required"/>
<xs:element name="salary" type="xs:decimal" use="optional"/>
```

Developed by Metaphor (c) 2002