XML

XML

- XML (Extensible Markup Language - «расширяемый язык разметки») - это универсальный, независящий от платформы стандарт описания информациии, который можно использовать для представления иерархических данных и унификации передаваемой информации.
- XML стал стандартом передачи данных в сети Интернет.
- Стандарт XML и связанных с ним форматов определяется консорциумом W3C(World Wide Web Consortium).

```
<?xml version="1.0" encoding="UTF-8"?>
<root>
   <!-- Комментарий -->
   <books  count="2">
      <book  pages="246" price="548.00">
        <title>Рефакторинг</title>
        <author>Мартин Фаулер</author>
      </book>
      <book  pages="186" price="382.00">
        <title>Разработка пользовательского интерфейса</title>
        <author>Дженифер Тидвелл</author>
      </book>
      <book></book>
      <book />
   </books>
</root>
```

XML КРИТЕРИИ

- Каждому начальному дескриптору должен соответствовать конечный дескриптор
- Пустой элемент должен завершаться значением />
- Элементы никогда не могут перекрываться
- Элемент не может содержать в себе двух элементов с одним и тем же именем в одном множестве.
- Документ может иметь только один корневой элемент
- Атрибуты должны иметь кавычки вокруг значений
- Комментарии и инструкции обработки не должны помещаться внутри дескрипторов
- XML чувствителен к регистру

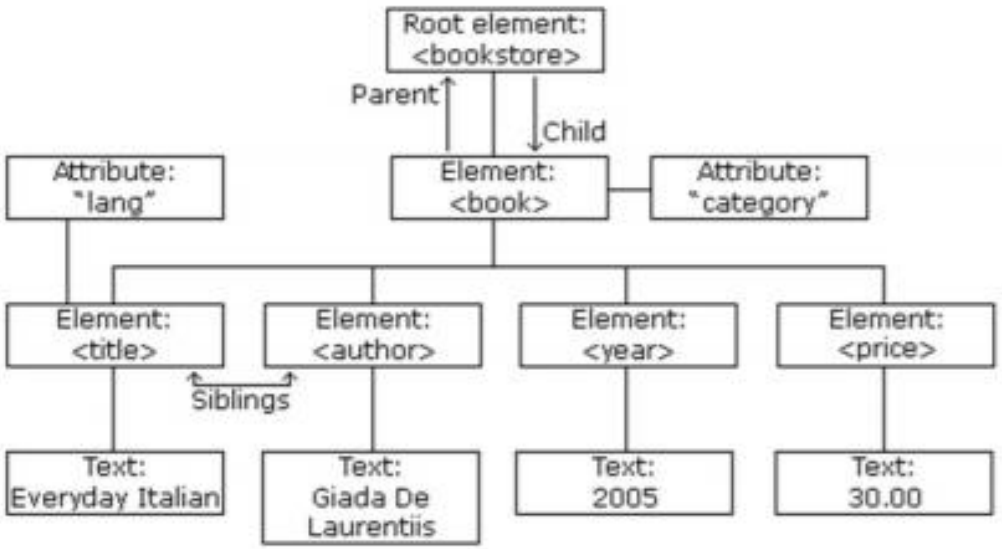
СТРУКТУРА ДОКУМЕНТА

- XML-декларация – эта конструкция находится в самом начале документа и необходима для его правильного разбора. Может отсутствовать.
- Комментарии
- Корневой элемент
- Дочерние элементы. Именно они и определяют структуру документа
- Пустые элементы – элемент может не содержать внутри никакой информации.
- Атрибуты – включаются внутрь открывающегося тэга элемента и могут нести дополнительные данные.

```
<?xml version="1.0" encoding="UTF-8"?>
<root>
   <!-- Комментарий -->
   <books count="2">
      <book pages="246" price="548.00">
        <title>Рефакторинг</title>
        <author>Мартин Фаулер</author>
      </book>
      <book pages="186" price="382.00">
        <title>Разработка пользовательского интерфейса</title>
        <author>Дженифер Тидвелл</author>
      </book>
      <book></book>
      <book />
   </books>
</root>
```

ПРИМЕР

Example:




```
<bookstore>
  <book category="COOKING">
    <title lang="en">Everyday Italian</title>
    <author>Giada De Laurentiis</author>
    <year>2005</year>
    <price>30.00</price>
  </book>
  <book category="CHILDREN">
    <title lang="en">Harry Potter</title>
    <author>J K. Rowling</author>
    <year>2005</year>
    <price>29.99</price>
  </book>
  <book category="WEB">
    <title lang="en">Learning XML</title>
    <author>Erik T. Ray</author>
    <year>2003</year>
    <price>39.95</price>
  </book>
</bookstore>
```

ДОПОЛНИТЕЛЬНАЯ ИНФОРМАЦИЯ

- Схемы данных (Schemas) предназначены для создания правил построения XML-документов.
- XSL – это язык преобразований XSLT (XSL Transformation). Он применяется для преобразования XSL–документов в другие типы документов или другие XSL -документы. Часто XSLT используется для преобразования XSL–документа в формат HTML.
- XQuery — язык запросов, разработанный для обработки данных в формате XML. XQuery использует XML как свою модель данных.
- XPath (XML Path Language) — язык запросов к элементам XML-документа. Разработан для организации доступа к частям документа XML в файлах трансформации XSLT и является стандартом консорциума W3C.

ЗАДАНИЕ

- Создать XML-файл, хранящий языки и денежные знаки для стран, проверить правильность кода при помощи веб браузера
- Создать XML-файл, описывающий структуру IVKHK (отделение, специальность, группа, студент), проверить правильность кода при помощи веб браузера
- Создать XML-файл – рецепты пиццы, проверить правильность кода при помощи веб браузера

При создании файлов, используйте также атрибуты.

XML

Плюсы

- Открытый формат
- Упрощает обмен, передачу данных
- Поддержка множества языков программирования

Минусы

- Не подходит для хранения больших данных
- Не подходит для хранения видео и изображений
- Может содержать сложную структуру

СИМВОЛЬНЫЕ СУЩНОСТИ

Predefined character entities:

- & - символ &
- < - символ <
- > - символ >
- ' - символ ‘
- " - символ “

Numbered character entities

<usr>⅞ Ⅻ </usr>

PCDATA CDATA

- PCDATA
  - PCDATA означает анализируемые символьные данные.
  - Символьные данные это текст, который находится между открывающим и закрывающим тегами XML элемента.
  - PCDATA — это текст, который будет анализироваться парсером. Т.е. парсер будет проверять этот текст на наличие сущностей и другой разметки.
  - Теги внутри такого текста будут восприняты, как вложенная разметка, а сущности будут раскрыты.
  - Тем не менее, анализируемые символьные данные не должны содержать символы **&**, **<** или **>**. Они должны быть представлены соответствующими сущностями **&**, **<** и **>**.
- CDATA
  - CDATA означает неанализируемые символьные данные.
  - CDATA — это текст, который не будет анализироваться парсером. Теги внутри такого текста не будут восприняты, как вложенная разметка, а сущности не будут раскрыты.

```
<?xml version="1.0" encoding="utf-8"?>
<!-- In this collection, we will keep each title "as is" -->
<videos>
  <![CDATA[<VdoColl>Collection of Videos</VdoColl>]]>
  <video>
        <title>The Distinguished Gentleman</title>
        <director>Jonathan Lynn</director>
        <length>112 Minutes</length>
        <format>DVD</format>
        <rating>R</rating>
  </video>
  <video>
        <title>Her Alibi</title>
        <director>Bruce Beresford</director>
        <length>94 Mins</length>
        <format>DVD</format>
        <rating>PG-13</rating>
  </video>
</videos>
```

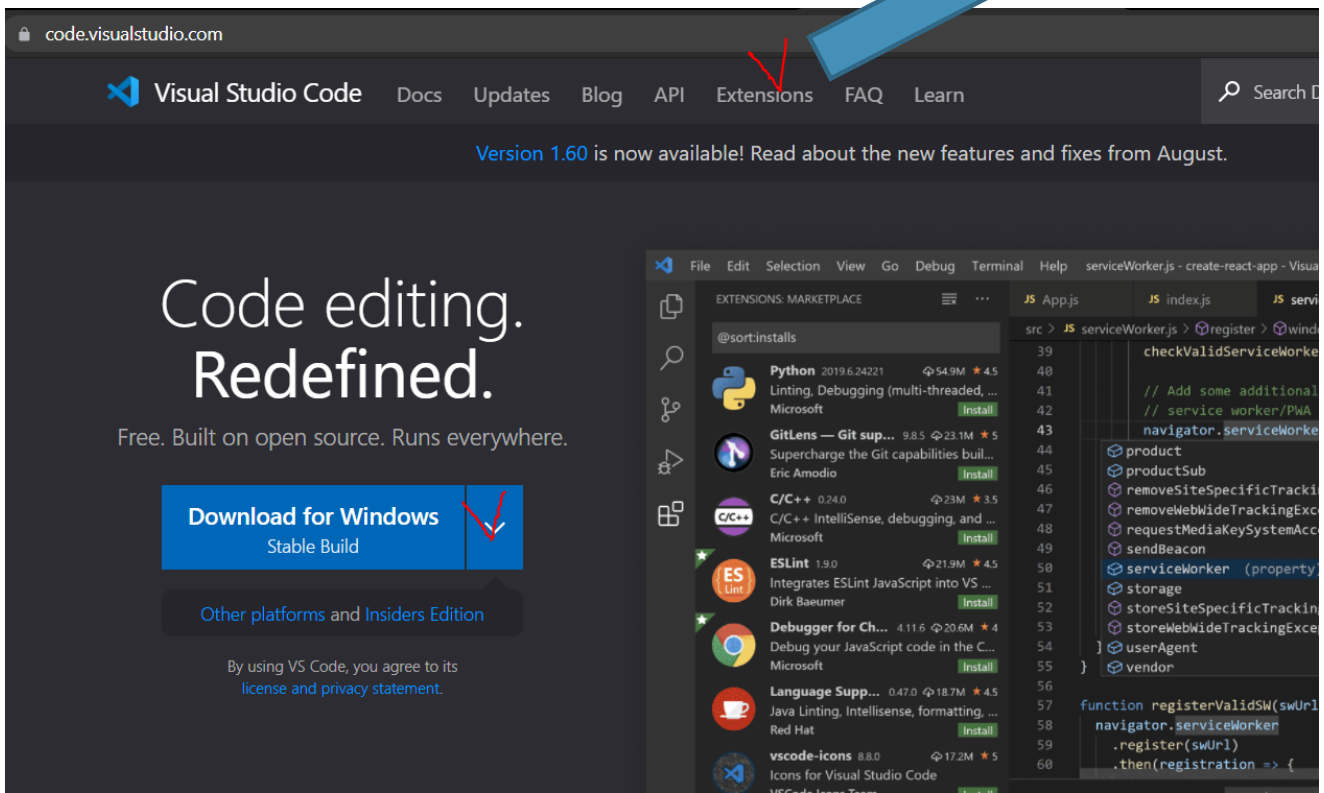
http://www.functionx.com/xml/Lesson05.htm

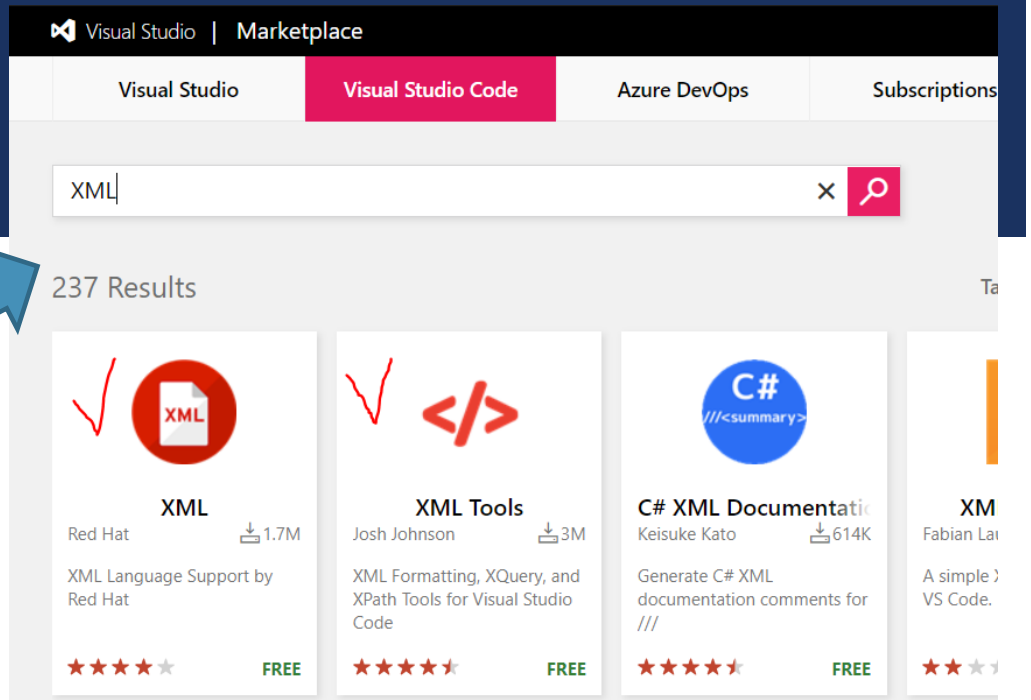
ВОПРОСЫ

- Обязателен ли корневой элемент?
- Обязателен ли пролог документа XML?
- Можно ли в xml документе добавить атрибут в закрывающий тег?
- Как можно записать символ больше в xml-документе?
- XML теги чувствительны к регистру?
- Значения атрибутов при создании XML должны заключаться в кавычки обязательно?
- Может ли в имене тега в xml документе быть пробелы, к примеру вот так <your last name> ?

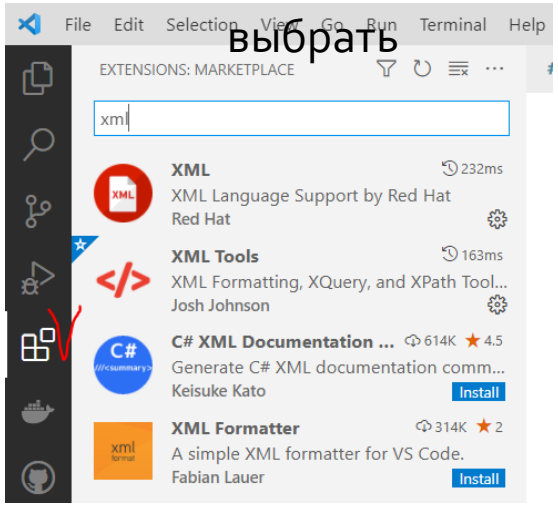
VISUAL STUDIO CODE

- https://code.visualstudio.com/





или в Visual Studio Code выбрать



ПРИМЕР. ИСПОЛЬЗОВАНИЕ CSS

people.xml > ...

```xml
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet href="people.css" type="text/css"?>
<people>
    <person>
        <name>Natalja Ivleva</name>
        <email>natalja.ivleva@test.com</email>
        <phone>1234567</phone>
    </person>
    <person>
        <name>Oleg Makarov</name>
        <email>oleg.makarov@test.com</email>
        <phone>7654321</phone>
    </person>
</people>
```

people.css

```css
people {
    display: block;
    margin: 5em;
}

name {
    display:block;
    font-size: x-large;
    color: blue;
    margin-top: 2em;
}

phone {
    display: block;
    font-size: 0.8em;
    color: black;
}

email {
    display: block;
    font-size: 0.8em;
    color: grey;
}
```

Natalja Ivleva
natalja.ivleva@test.com
1234567

Oleg Makarov
oleg.makarov@test.com
7654321

XML - DOCUMENT TYPE DEFINITION - DTD

- Определение типов документа (DTD) декларирует допустимые строительные блоки XML документа. Оно задает структуру документа со списком допустимых элементов и атрибутов.
- DTD может декларироваться как в коде самого XML документа, так и во внешнем файле с подключением его к XML документу.
- С точки зрения DTD все XML (и HTML) документы состоят из следующих строительных блоком:
  - Элементы
  - Атрибуты
  - Сущности
  - PCDATA
  - CDATA

ОПРЕДЕЛЕНИЕ ЭЛЕМЕНТОВ В XML DTD

- Декларация элементов `<!ELEMENT имя-элемента (содержимое-элемента)>`
- Пустые элементы `<!ELEMENT имя-элемента EMPTY>`
- Элементы с анализируемыми символьными данными `<!ELEMENT имя-элемента (#PCDATA)>`
- Элементы с потомками (последовательности) `<!ELEMENT имя-элемента (потомок1,потомок2,...)>`
  - Когда потомки декларируются в последовательности разделенные запятыми, они должны появляться в том же самом порядке. В полной декларации дочерние элементы также должны быть декларированы.
- Декларация минимум одного элемента **+** `<!ELEMENT имя-элемента (имя-потомка+)>`
- Декларирование от нуля и больше элементов ***** `<!ELEMENT имя-элемента (имя-потомка*)>`
- Декларирование от нуля до одного элемента **?** `<!ELEMENT имя-элемента (имя-потомка?)>`
- Декларирование альтернативных элементов **|** `<!ELEMENT note (to,from,header,(message|body))>`

ОПРЕДЕЛЕНИЕ АТРИБУТОВ В XML DTD

- Декларация атрибутов

```
<!ATTLIST имя-элемента имя-атрибута тип-атрибута значение-атрибута>
```

Параметр "***значение-атрибута***" может принимать следующие значение:

Значение	Описание
value	Значение атрибута по умолчанию
#REQUIRED	Атрибут обязателен
#IMPLIED	Атрибут не обязателен
#FIXED value	Атрибут имеет фиксированное значение

DTD:

```
<!ELEMENT square EMPTY>
<!ATTLIST square width CDATA "0">
```

XML:

```
<square width="100" />
```

DTD:

```
<!ATTLIST payment type (check|cash) "cash">
```

XML:

```
<payment type="check" />
```

или

```
<payment type="cash" />
```

DTD:

```
<!ATTLIST person number CDATA #REQUIRED>
```

Правильный XML:

```
<person number="5677" />
```

СЛЕДУЕТ ЛИ ИЗБЕГАТЬ ИСПОЛЬЗОВАНИЯ АТРИБУТОВ?

- При использовании атрибутов могут возникать некоторые проблемы:
  - атрибуты не могут содержать множественные значения (элементы могут)
  - атрибуты сложно расширять (для последующих изменений)
  - атрибуты не могут описывать структуры (элементы могут)
  - атрибуты более сложны для манипулирования программами
  - значения атрибутов сложно проверять по DTD
- Если использовать атрибуты в качестве контейнеров данных, то конечные документы будут сложны для чтения и манипулирования. **Для описания данных следует использовать элементы**.
- Атрибуты следует использовать для предоставления информации, не относящейся к данным.
- Для негласного правила запрета на использование атрибутов также есть одно исключение. Оно вступает в действие, когда, например, необходимо присвоить элементам идентификатор. <note id="p501">

ВНУТРЕННЯЯ ДЕКЛАРАЦИЯ DTD

- Если DTD декларируется внутри XML файла, то она должны быть заключена в специальный тег декларации **DOCTYPE**, который имеет следующий синтаксис:

```
<?xml version="1.0"?>
<!DOCTYPE note [
<!ELEMENT note (to,from,heading,body)>
<!ELEMENT to (#PCDATA)>
<!ELEMENT from (#PCDATA)>
<!ELEMENT heading (#PCDATA)>
<!ELEMENT body (#PCDATA)>
]>

<note>
   <to>Tove</to>
   <from>Jani</from>
   <heading>Напоминание</heading>
   <body>Не забудь обо мне в эти выходные</body>
</note>
```

- ***!DOCTYPE note*** определяет, что корневым элементом документа является **note**
- ***!ELEMENT note*** определяет, что элемент **note** содержит четыре элемента: **to, from, heading, body**
- ***!ELEMENT to*** определяет, что элемент **to** должен быть типа "#PCDATA"
- ***!ELEMENT from*** определяет, что элемент **from** должен быть типа "#PCDATA"
- ***!ELEMENT heading*** определяет, что элемент **heading** должен быть типа "#PCDATA"
- ***!ELEMENT body*** определяет, что элемент **body** должен быть типа "#PCDATA"

ВНЕШНЯЯ ДЕКЛАРАЦИЯ DTD

- Если DTD декларируется во внешнем файле, то подключение осуществляется следующим образом:

```
<?xml version="1.0"?>
<!DOCTYPE note SYSTEM "note.dtd">
<note>
    <to>Tove</to>
    <from>Jani</from>
    <heading>Напоминание</heading>
    <body>Не забудь обо мне в выходные</body>
</note>
```

А вот что содержится в файле "note.dtd", который декларирует DTD:

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

ПРИМЕР (1)

Пример очень простого XML DTD, описывающего список людей:

```
<!ELEMENT people_list (person*)>
<!ELEMENT person (name, birthdate?, gender?, socialsecuritynumber?)>
<!ELEMENT name (#PCDATA) >
<!ELEMENT birthdate (#PCDATA) >
<!ELEMENT gender (#PCDATA) >
<!ELEMENT socialsecuritynumber (#PCDATA) >
```

Начиная с первой строки:

1. Элемент `<people_list>` содержит любое число элементов `<person>` . Знак `<*>` означает, что возможно 0, 1 или более элементов `<person>` внутри элемента `<people_list>` .
2. Элемент `<person>` содержит элементы `<name>` , `<birthdate>` , `<gender>` и `<socialsecuritynumber>` . Знак `<?>` означает, что элемент необязателен. Элемент `<name>` не содержит `<?>` , что означает, что элемент `<person>` ***обязательно должен*** содержать элемент `<name>` .
3. Элемент `<name>` содержит данные.
4. Элемент `<birthdate>` содержит данные.
5. Элемент `<gender>` содержит данные.
6. Элемент `<socialsecuritynumber>` содержит данные.

ПРИМЕР (2)

Пример XML-документа, использующего этот DTD:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE people_list SYSTEM "example.dtd">
<people_list>
   <person>
      <name>
         Fred Bloggs
      </name>
      <birthdate>
         27/11/2008
      </birthdate>
      <gender>
         Male
      </gender>
      <socialsecuritynumber>
         1234567890
      </socialsecuritynumber>
   </person>
</people_list>
```

ДЛЯ ЧЕГО ИСПОЛЬЗУЮТ DTD?

- С DTD каждый ваш XML файл может нести описание своего собственного формата.
- С DTD различные, не связанные друг с другом, группы людей могут приходить к соглашению о стандартах взаимно обмениваемых данных.
- С DTD вы можете быть уверены, что получаемые из внешних источников данные будут корректными.
- Также, вы можете использовать DTD, чтобы проводит проверки корректности своих собственных данных.

ПРИМЕР

1)SIMPLE-XML-MOVIES.XML:

```
<?xml version="1.0"?>
<!DOCTYPE MOVIES SYSTEM "SIMPLE-XML-MOVIES.DTD">
<MOVIES>
        <MOVIE TYPE="COMEDY" RATING="G" REVIEW="4" YEAR="1995">
                <TITLE>GOLMAAL</TITLE>
                <WRITER>NITIN </WRITER>
                <PRODUCER>AASHISH</PRODUCER>
                <DIRECTOR>KRISHNA</DIRECTOR>
                <ACTOR>GAURAV</ACTOR>
                <COMMENTS>OSCAR</COMMENTS>
        </MOVIE>
        <MOVIE TYPE="THRILLER" RATING="G" REVIEW="4" YEAR="1998">
                <TITLE>BAZZIGAR</TITLE>
                <WRITER>AASHISH</WRITER>
                <PRODUCER>NITIN</PRODUCER>
                <DIRECTOR>VIPUL</DIRECTOR>
                <ACTOR> KRISHNA</ACTOR>
                <COMMENTS>AWESOME</COMMENTS>
        </MOVIE>
        <MOVIE TYPE="ROMANCE" RATING="G" REVIEW="4" YEAR="1998">
                <TITLE>MURDER</TITLE>
                <WRITER>VIPUL</WRITER>
                <PRODUCER>KRISHNA</PRODUCER>
                <DIRECTOR>NITIN</DIRECTOR>
                <ACTOR>AASHISH</ACTOR>
                <COMMENTS>AWESOME</COMMENTS>
        </MOVIE>
</MOVIES>
```

2)SIMPLE-XML-MOVIES.DTD:

```
<!ELEMENT MOVIES (MOVIE)+>
<!ELEMENT MOVIE (TITLE,WRITER+,PRODUCER,DIRECTOR,ACTOR*,COMMENT?)>
<!ATTLIST MOVIE
        TYPE (DRAMA|ROMANCE|TRADEGY|COMEDY|HORROR|THRILLER) "DRAMA"
        RATING (G|O|E|K|B) "G"
        REVIEW (1|2|3|4|5) "3"
        YEAR CDATA #IMPLIED >
<!ELEMENT TITLE (#PCDATA)>
<!ELEMENT WRITER (#PCDATA)>
<!ELEMENT PRODUCER (#PCDATA)>
<!ELEMENT DIRECTOR (#PCDATA)>
<!ELEMENT ACTOR (#PCDATA)>
<!ELEMENT COMMENT (#PCDATA)>
```

ЗАДАНИЕ

- Создайте XML документ, используя DTD

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE contacts [
   <!ELEMENT contacts (contact*)>
   <!ELEMENT contact (name?, email+)>
   <!ELEMENT name (#PCDATA)>
   <!ELEMENT email (#PCDATA)>
<!ATTLIST name type (firstname | lastname) "firstname">
]>
```

- Создайте DTD и примените к ранее созданным XML файлам (см. слайд 7)

XML SCHEMA

- XML схема описывает структуру XML документа.
- Язык XML схем также называют *Определение схемы XML* или *XSD* (от анг. XML Schema Definition).
- XML схема — это основанная на XML альтернатива DTD.
- XML схема:
  - определяет элементы, которые могут появляться в XML документе
  - определяет атрибуты, которые могут появляться в XML документе
  - определяет, какие элементы являются дочерними
  - определяет порядок дочерних элементов
  - определяет количество дочерних элементов
  - определяет, пустой ли элемент или может содержать текст
  - **определяет типы данных элементов и атрибутов**
  - определяет фиксированные значения и значения по умолчанию элементов и атрибутов

```
<?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>
```

ОПРЕДЕЛЕНИЕ ПРОСТЫХ ЭЛЕМЕНТОВ В XML СХЕМЕ

- Простой элемент — это элемент XML, который содержит только текст или число или … Простой элемент не может содержать другие элементы или атрибуты.
- <xs:element name="xxx" type="yyy"/>

  где **xxx** — имя элемента, а **yyy** — тип данных элемента.
- XML схемы имеют множество встроенных типов данных. Наиболее часто используемыми являются следующие типы: xs:string, xs:decimal, xs:integer, xs:boolean, xs:date, xs:time

```
<lastname>Refsnes</lastname>
<age>36</age>
<dateborn>1970-03-27</dateborn>
```

```
<xs:element name="lastname" type="xs:string"/>
<xs:element name="age" type="xs:integer"/>
<xs:element name="dateborn" type="xs:date"/>
```

- Значения по умолчанию и фиксированные значения

```
<xs:element name="color" type="xs:string" default="red"/>
```

```
<xs:element name="color" type="xs:string" fixed="red"/>
```

ОПРЕДЕЛЕНИЕ АТРИБУТОВ В XML СХЕМЕ

- Простые элементы не могут иметь атрибуты. Если у элемента есть атрибуты, то он относится к комплексным или составным типам. Но сам по себе атрибут всегда декларируется, как простой тип.

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

```
<xs:attribute name="lang" type="xs:string"/>
```

- Атрибуты могут иметь значения по умолчанию ИЛИ фиксированные значения. Значение по умолчанию присваивается атрибуту автоматически, если не определено никакого другого значения.

```
<xs:attribute name="lang" type="xs:string" default="EN"/>
```

- По умолчанию атрибуты являются необязательными для использования. Чтобы декларировать обязательный атрибут, следует воспользоваться атрибутом "use":

```
<xs:attribute name="lang" type="xs:string" use="required"/>
```

ОГРАНИЧЕНИЯ ПО ЗНАЧЕНИЮ И НАБОРУ ЗНАЧЕНИЙ

```
<xs:element name="age">
   <xs:simpleType>
      <xs:restriction base="xs:integer">
         <xs:minInclusive value="0"/>
         <xs:maxInclusive value="120"/>
      </xs:restriction>
   </xs:simpleType>
</xs:element>
```

```
<xs:element name="car">
   <xs:simpleType>
      <xs:restriction base="xs:string">
         <xs:enumeration value="Audi"/>
         <xs:enumeration value="Golf"/>
         <xs:enumeration value="BMW"/>
      </xs:restriction>
   </xs:simpleType>
</xs:element>
```

или

```
<xs:element name="car" type="carType"/>
<xs:simpleType name="carType">
   <xs:restriction base="xs:string">
      <xs:enumeration value="Audi"/>
      <xs:enumeration value="Golf"/>
      <xs:enumeration value="BMW"/>
   </xs:restriction>
</xs:simpleType>
```

ОГРАНИЧЕНИЯ ПО СЕРИИ ЗНАЧЕНИЙ

Чтобы ограничить содержимое XML элемента серией чисел или букв, следует использовать ограничитель pattern.

```
<xs:element name="letter">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:pattern value="[a-z]"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

Элемент letter - может быть ОДНА буква в нижнем регистре в диапазоне от "a" до "z"

```
<xs:element name="initials">
  <xs:simpleType>
    <xs:restriction base="xs:string">
      <xs:pattern value="[A-Z][A-Z][A-Z]"/>
    </xs:restriction>
  </xs:simpleType>
</xs:element>
```

Элемент "initials" с ограничением. Его единственным допустимым значением может быть ТРИ буквы в ВЕРХНЕМ РЕГИСТРЕ в диапазоне от "A" до "Z"

https://www.w3schools.com/xml/schema_facets.asp

ОГРАНИЧЕНИЯ ДЛЯ ТИПОВ ДАННЫХ

Ограничитель	Описание
enumeration	Определяет список приемлемых значений
fractionDigits	Определяет максимальное число знаков после десятичной запятой. Должно быть равно или больше нуля
length	Определяет точное число символов или объектов списка. Должно быть равно или больше нуля
maxExclusive	Определяет верхнюю границу для числовых значений (значение должно быть меньше указанного здесь)
maxInclusive	Определяет верхнюю границу для числовых значений (значение должно быть меньше или равно указанному здесь)
maxLength	Определяет максимальное число символов или объектов списка. Должно быть равно или больше нуля
minExclusive	Определяет нижнюю границу для числовых значений (значение должно быть больше указанного здесь)
minInclusive	Определяет нижнюю границу для числовых значений (значение должно быть больше или равно указанному здесь)
minLength	Определяет минимальное число символов или объектов списка. Должно быть равно или больше нуля
pattern	Определяет точную последовательность приемлемых символов
totalDigits	Определяет точное количество допустимых цифр. Должно быть больше нуля
whiteSpace	Определяет способ обработки пробельных символов

ОПРЕДЕЛЕНИЕ СОСТАВНЫХ ЭЛЕМЕНТОВ В XML СХЕМЕ

- Составной элемент содержит другие элементы и/или атрибуты.
- Существует четыре вида составных элементов:
  - пустые элементы
  - элементы, которые содержат только другие элементы
  - элементы, которые содержат только текст
  - элементы, которые содержат как другие элементы, так и текст

```
<product pid="1345"/>
```

```
<employee>
   <firstname>John</firstname>
   <lastname>Smith</lastname>
</employee>
```

ПРИМЕР

```
<product pid="1345"/>
```

```
<employee>
   <firstname>John</firstname>
   <lastname>Smith</lastname>
</employee>
```

```
<xs:element name="product">
   <xs:complexType>
      <xs:attribute name="prodid" type="xs:positiveInteger"/>
   </xs:complexType>
</xs:element>
```

```
<xs:element name="employee" type="personinfo"/>
<xs:complexType name="personinfo">
   <xs:sequence>
      <xs:element name="firstname" type="xs:string"/>
      <xs:element name="lastname" type="xs:string"/>
   </xs:sequence>
</xs:complexType>
```

```
<xs:element name="employee">
   <xs:complexType>
      <xs:sequence>
         <xs:element name="firstname" type="xs:string"/>
         <xs:element name="lastname" type="xs:string"/>
      </xs:sequence>
   </xs:complexType>
</xs:element>
```

```
<xs:element name="employee" type="personinfo"/>
<xs:element name="student" type="personinfo"/>
<xs:element name="member" type="personinfo"/>
<xs:complexType name="personinfo">
   <xs:sequence>
      <xs:element name="firstname" type="xs:string"/>
      <xs:element name="lastname" type="xs:string"/>
   </xs:sequence>
</xs:complexType>
```

ПРИМЕР - ЗАДАНИЕ

1. Посмотрите пример создания схемы и ответьте на вопросы.
Ссылка на пример:
https://msiter.ru/tutorials/uchebnik-po-xml-shemam/primer-xml-shemy
Нарисуйте DOM схему для документа.
Вопросы:

- Почему для описания элемента shipto используются complexType и sequence?
- Для чего нужны атрибуты maxOccurs и minOccurs?
- Как определить, что атрибут обязателен в элементе?
- Какие типы существуют для описания числовых данных?

2. Создайте XML схему в отдельном файле для описания книг (books.xml)

```
<Books>
    <Book>
        <BookID>1</BookID>
        <ISBN>978-161729126</ISBN>
        <Title>Pro ASP.NET MVC 4</Title>
        <Description>The ASP.NET MVC 4 Framework is the latest evolution of Microsoft's A
        <Price>38.8500</Price>
        <Pages>500</Pages>
        <Year>2013-01-16</Year>
        <Language>ENG</Language>
        <Category>Web</Category>
        <Authors>
            <Author AuthorID="2" Firstname="Adam" Lastname="Freeman" />
        </Authors>
    </Book>
    <Book>
        <BookID>2</BookID>
        <ISBN>978-161729125</ISBN>
        <Title>Professional ASP.NET MVC 4</Title>
        <Description>An outstanding author team presents the ultimate Wrox guide to ASP.N
        <Price>29.4700</Price>
        <Pages>400</Pages>
```

ЗАДАНИЕ

- Посмотрите RSS новостей https://www.postimees.ee/rss создайте XML файл, который содержит 10 новостей (item – элементы)
- Создайте DTD и XML schema для новостей

XPATH

- XPath - язык запросов к элементам XML-документа.
- Разработан для организации доступа к частям документа XML в файлах трансформации XSLT и является стандартом консорциума W3C.
- XPath призван реализовать навигацию по DOM в XML.
- В XPath используется компактный синтаксис, отличный от принятого в XML.

ЗАДАНИЕ

- Изучите учебный материал по Xpath
  - https://www.w3schools.com/xml/xpath_intro.asp
  - https://msiter.ru/tutorials/uchebnik-xml-dlya-nachinayushchih/xml-i-xpath
- Добавьте 5 фильмов в файл movies.xml
- С помощью XPath выведите:
  - Фильмы определенного жанра
  - Названия фильмов 21 века
  - Фильмы определенного режиссера
  - Фильмы, где играет определенный актер
  - Фильмы, где играет больше 3 актеров

- Сохраните запросы XPath и результаты выполнения запросов в текстовый файл

ИСТОЧНИКИ

- https://www.w3schools.com/xml/default.asp
- Учебник XML DTD. https://msiter.ru/tutorials/uchebnik-xml-dtd/
- https://ru.wikipedia.org/wiki/DTD
- Учебник по XML схемам. https://msiter.ru/tutorials/uchebnik-po-xml-shemam