XML

eXtensible Markup Language

К.т.н., доцент
Феликс Васильевич Филиппов

Что такое XML?

eXtensible Markup Language
Расширяемый Язык Разметки

- XML — язык для структурирования данных.
- Теги XML не предопределены. Можно использовать любые (свои) теги.
- XML рекомендован **W3C**— **W**orld **W**ide **W**eb **C**onsortium полные спецификации http://www.w3c.org/xml

Для чего используется XML

XML является метаязыком для создания различных языков разметки, которые способны определять произвольные структуры данных — смысловые текстовые объекты, двоичные данные, записи в базе данных, сценарии.

XML используется в web-приложениях при работе браузеров, которые отображают информацию, находящуюся на web-серверах. При этом пользователю отдельно передаются данные в виде **XML-документа**, и отдельно **XSD-документа**— с **правилами интерпретации** этих данных.

Чем отличаются XML и HTML?

XML не является заменой HTML - они были разработаны с различными целями:

- **HTML** был разработан *для управления отображением данных*. Основное внимание уделяется тому, как данные отображаются.

- **XML** был создан *для семантического описания данных*. Основное внимание уделяется тому, как данные удобнее структурировать.

Таким образом, HTML — связан с отображением информации, в то время как XML — с семантическим описанием информации (структурированием данных) .

Ошибочные представления о XML

XML – не язык программирования

(но язык программирования можно описать с помощью XML-разметки)

XML – не протокол передачи данных

(но типичная задача XML описывать структуру документов/данных, передаваемых по компьютерным сетям)

XML – не структура базы данных

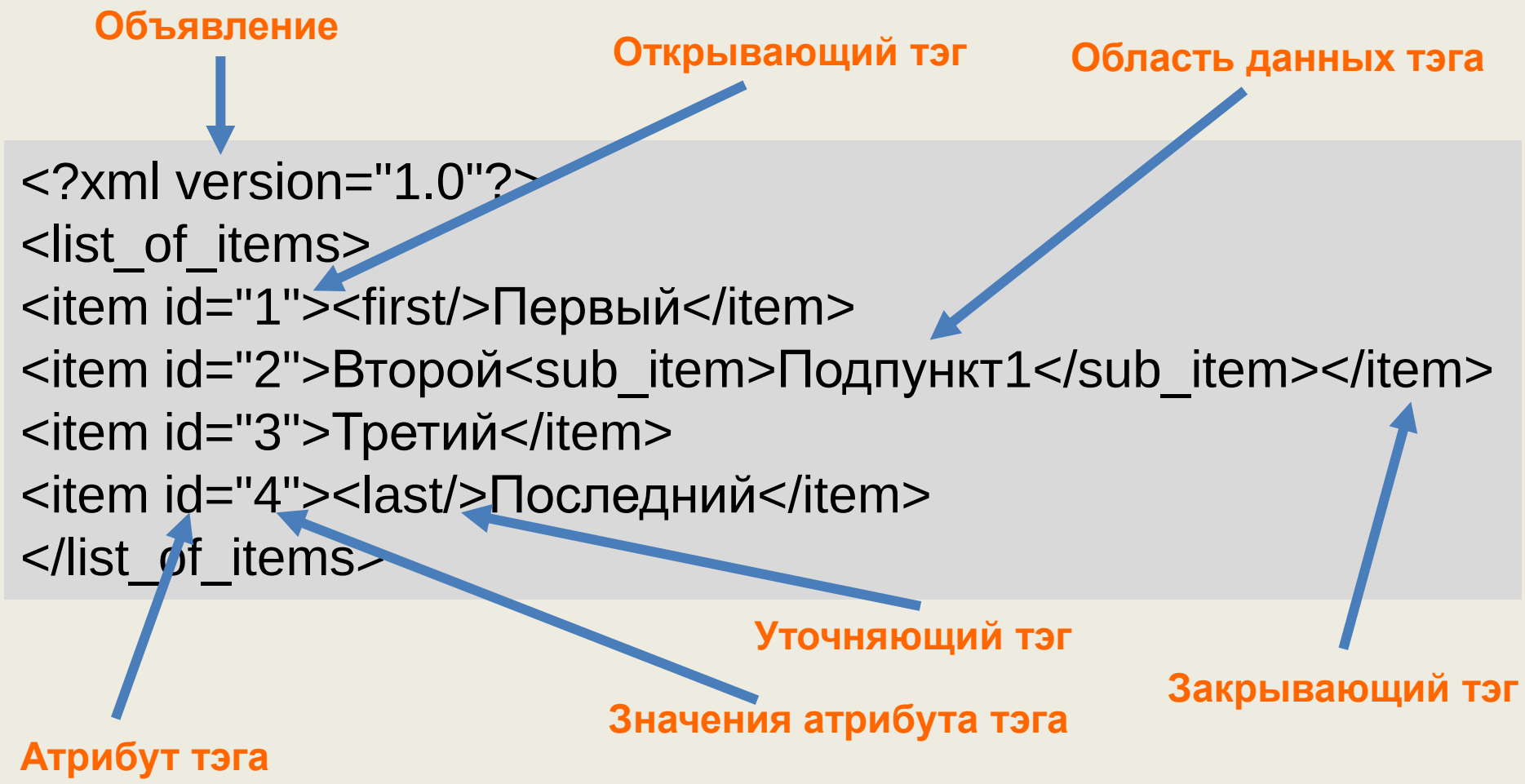
(но XML может храниться в БД и можно выполнять различные запросы к XML-данным)

Логика формирования XML-документа

	<?xml version="1.0"?>
Стакан	<GLASS>
Материал, из которого он сделан (прозрачный?)	<MATERIAL transparent="yes">glass</MATERIAL>
Высота в дюймах	<HEIGHT units="inches">6</HEIGHT>
Количество унций, которое в него помещается	<VOLUME units="ounces">16</VOLUME>
Его содержимое	<CONTENTS>
Описание любых твердых тел и их количества	<SOLID qty="2">ice cube</SOLID>
Описание любых жидкостей и их объема	<LIQUID qty="3" units="ounces">water</LIQUID>
Описание любой другой субстанции и ее количество	<OTHER qty="0"/>
	</CONTENTS>
Имеет или не имеет он крышку	<LID>yes</LID>
	</GLASS>

Автор документа создает его структуру, строит необходимые связи между элементами, используя те тэги, которые удовлетворяют его требованиям и добивается такого типа разметки, которое необходимо ему **для выполнения операций просмотра, поиска, анализа** XML-документа.

Пример XML-разметки

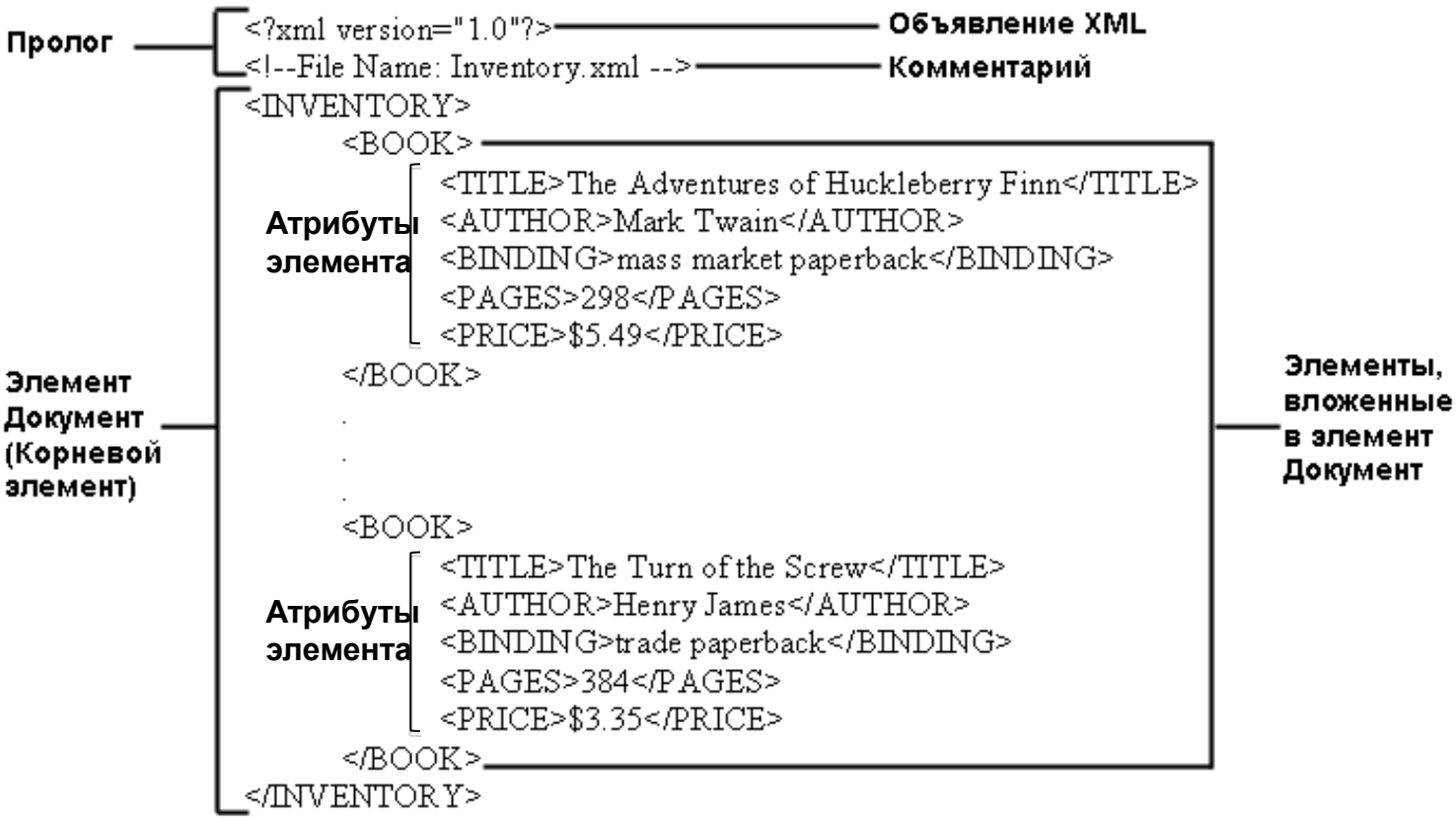


Требования к XML-документам

- В заголовке документа помещается объявление XML, в котором указывается язык разметки документа, номер его версии и дополнительная информация
- Каждый открывающий тэг, определяющий некоторую область данных в документе обязательно должен иметь своего закрывающего "напарника", т.е., в отличие от HTML, нельзя опускать закрывающие тэги
- В XML учитывается регистр символов
- Все значения атрибутов, используемых в определении тэгов, должны быть заключены в кавычки
- Вложенность тэгов в XML строго контролируется, поэтому необходимо следить за порядком следования открывающих и закрывающих тэгов
- Вся информация, располагающаяся между начальным и конечными тэгами, рассматривается в XML как данные и поэтому учитываются все символы форматирования (т.е. пробелы, переводы строк, табуляции не игнорируются, как в HTML)

Если XML- документ не нарушает приведенные правила, то он называется *формально-правильным* и все анализаторы, предназначенные для разбора XML- документов, смогут работать с ним корректно.

Структура XML-документа



Это чистая информация, завернутая в теги. Для того, чтобы отправить, получить и отобразить эту информацию, кто-то должен написать программу.

XML-заголовок
(encoding u standalone)

Примеры правильно построенных объявлений XML:

<?xml version ="1.0"?>
<?xml version = '1.0' encoding='US_ASCII' standalone='yes'?>
<?xml version = '1.0' encoding= 'iso_8859_1' standalone ="no"?>

encoding (кодировка) - задает кодировку символов, использованную в документе;

standalone (автономность) - сообщает процессору XML, есть ли другие файлы, которые нужно загружать.

Пространство имен
(namespace)

Пространство имен **(*namespace*)** – это группа имен элементов и атрибутов.

Добавляя префикс пространства имен к имени элемента или атрибута, мы сообщаем **анализатору** о том, из какого пространства имен оно происходит.

```
<library xmlns:book="http: /www.inquary. com/spec ">
   <book>
         <title>
         Семантический веб.
         </title>
   </book>
</library>
```

Пространство имен *(namespaces)*

- Позволяют избегать многозначных толкований элементов и атрибутов (с одинаковыми именами)
- Группируют понятия, относящиеся к одному и тому же приложению (объекту, понятию и т.д.)
- Используются уникальные идентификаторы – определяющие пространства имен
- Элемент или атрибут однозначно идентифицируется по своему имени плюс по пространству имен к которому элемент/атрибут относится

Сущности
(entity)

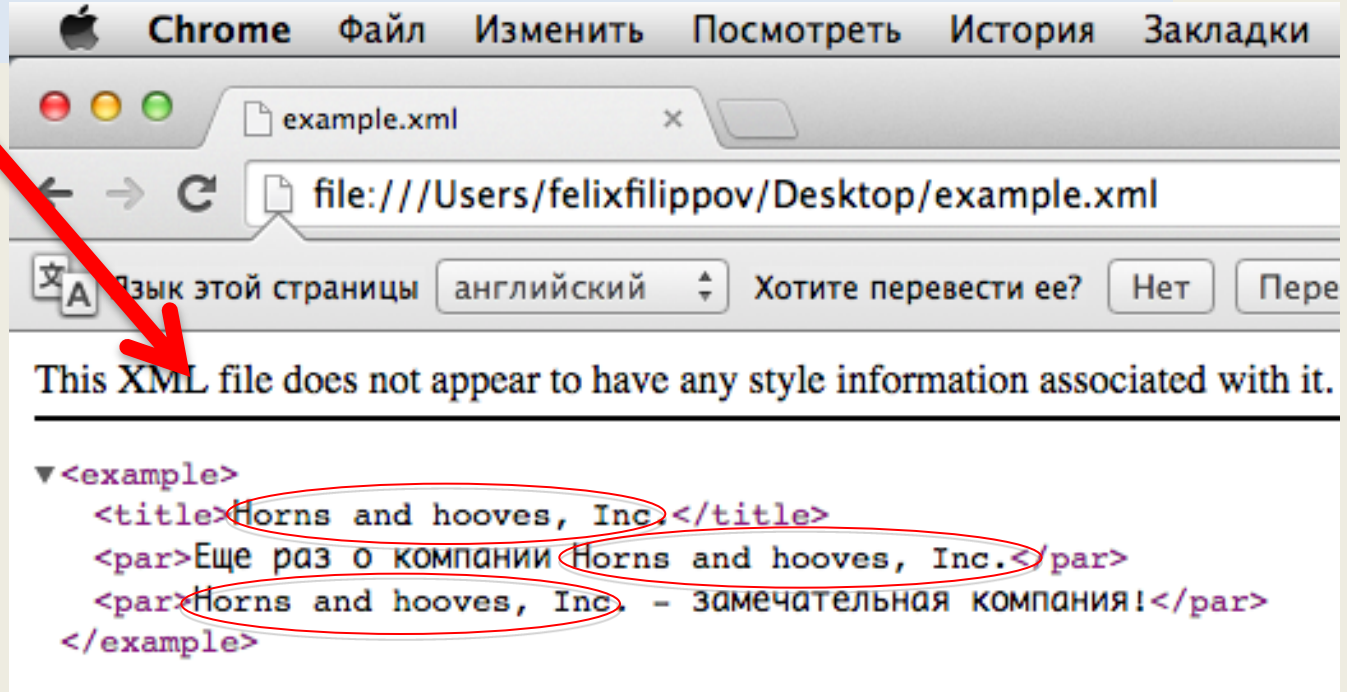
***Сущность* (*entity*)** является заместителем содержания, которую можно однажды объявить и многократно использовать почти в любом месте документа.

Используется два типа сущностей:

- ***internal entity*** (внутренние сущности);
- ***external entity*** (внешние сущности).

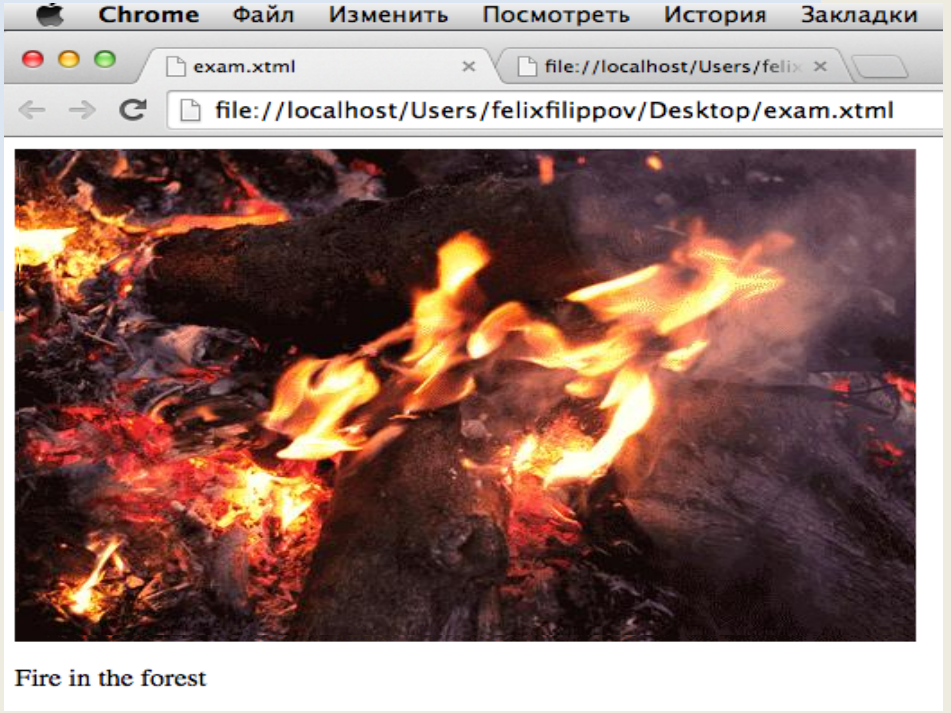
Внутренние сущности
(internal entity)

```
<?xml version="1.0"?>
<!DOCTYPE example
[<!ENTITY abc "Horns and hooves, Inc.”>]
>
<example>
<title>&abc;</title>
<par>Еще раз о компании &abc; </par>
<par>&abc; - замечательная компания!</par>
</example>
```



Внешние сущности
(external entity)

```
<?xml version="1.0"?>
<!DOCTYPE longdoc SYSTEM "/Users/felixfilippov/Desktop"
[
<!ENTITY picture SYSTEM "fire.gif" NDATA PNG>
<!ENTITY part2  "Fire in the forest">
]
>
<longdoc>
&picture;
&part2;
</longdoc>
```



Правила интерпретации XML-документа

XML использует определение типа документа (Document Type Definition — **DTD**) или схему (**XML Schema**) для интерпретации данных

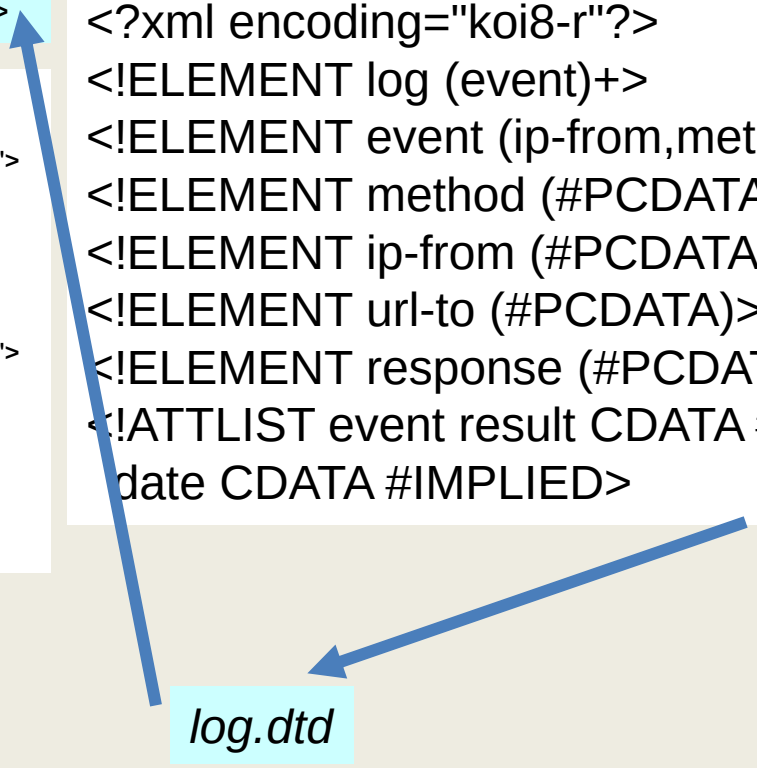
Инструкция	Описание DTD - инструкции
ELEMENT	Объявляет имя типа XML-элемента и его допустимые вложенные (дочерние) элементы.
ATTLIST	Объявляет список XML-атрибутов. Эти атрибуты определяются именем, типом данных, неявными значениями по умолчанию и именами любых элементов, позволяющих их использование.
ENTITY	Объявляет специальные символьные ссылки, текстовые макросы (наподобие инструкции #define языка C/C++) и другое повторяющееся содержимое (наподобие инструкции #include языка C/C++).
NOTATION	Объявляет внешнее содержимое, не относящееся к XML (например, двоичные графические данные), а также внешнее приложение, которое обрабатывает это содержимое.

Document Type Definition DTD

<!--DOCTYPE log SYSTEM "log.dtd"-->

```
<?xml version="1.0" encoding="koi-8"?>
<log>
<event date=" 27/May/2012:02:32:46 " result="success">
 <ip-from> 195.151.62.18 </ip-from>
<method>GET</method>
<url-to> /misc/</url-to>
<response>200</response>
</event>
<event date=" 27/May/2012:02:41:47 " result="success">
<ip-from> 195.209.248.12 </ip-from>
<method>GET</method>
<url-to> /soft.htm</url-to>
<response>400</response>
</event>
</log>
```

```
<?xml encoding="koi8-r"?>
<!ELEMENT log (event)+>
<!ELEMENT event (ip-from,method,uri-to,responce)>
<!ELEMENT method (#PCDATA)>
<!ELEMENT ip-from (#PCDATA)>
<!ELEMENT url-to (#PCDATA)>
<!ELEMENT response (#PCDATA)>
<!ATTLIST event result CDATA #IMPLIED
 date CDATA #IMPLIED>
```



log.dtd

XML Schema

Файл, содержащий XML Schema, обычно имеет расширение «.xsd» (**X**ML **S**chema **d**efinition). Пример файла **country.xsd**:

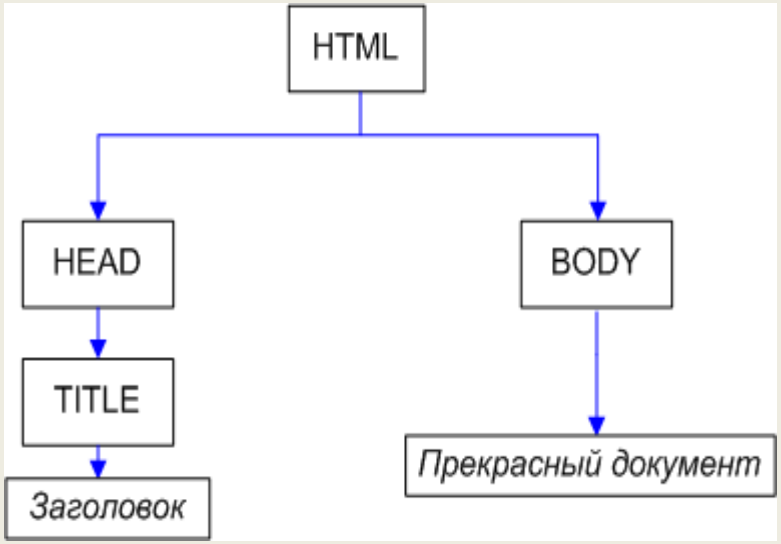
```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="country">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="country_name" type="xs:string"/>
        <xs:element name="population" type="xs:decimal"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

```
<?xml version="1.0" encoding="utf-8"?>
<country>
    <country_name>France</country_name>
    <population>59.7</population>
</country>
```

Объектная модель DOM

Одним из самых мощных интерфейсов доступа к содержимому XML документов является Document Object Model - DOM

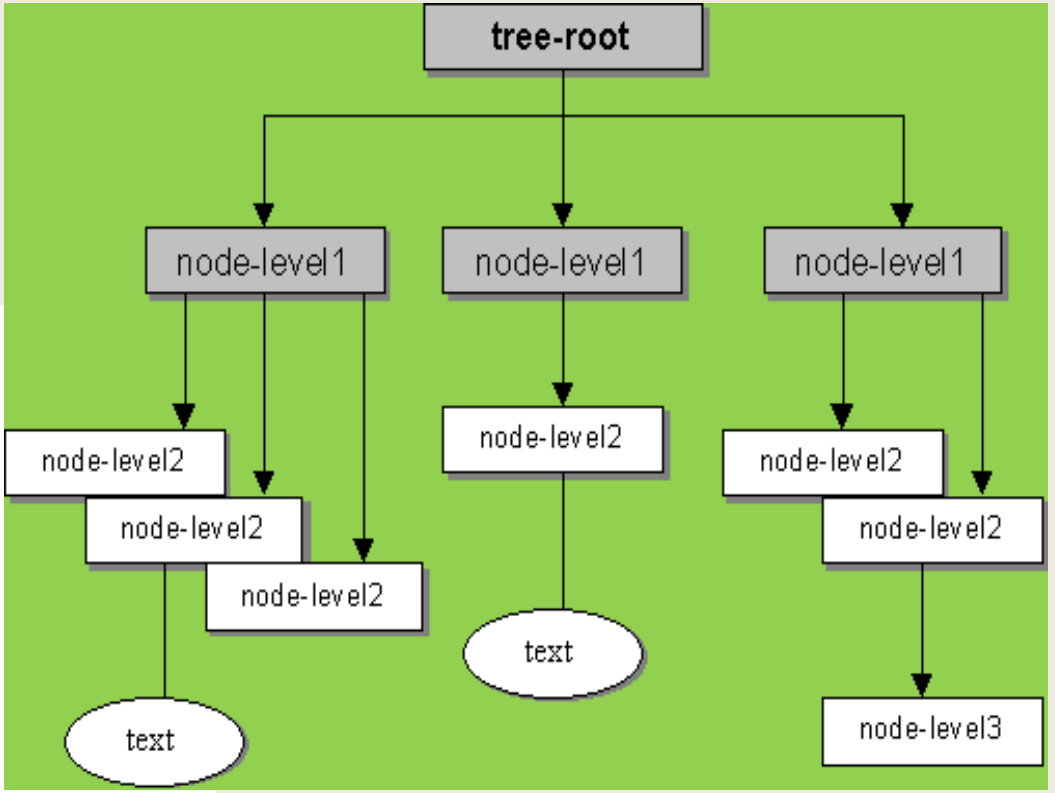
```
<html>
  <head>
      <title>
          Заголовок
       </title>
  </head>
  <body>
      Прекрасный документ
  </body>
</html>
```



DOM — это не зависящий от платформы и языка программный интерфейс, позволяющий программам и скриптам получить доступ к содержимому HTML, XHTML и XML-документов, а также изменять содержимое, структуру и оформление таких документов.

Уровни модели DOM

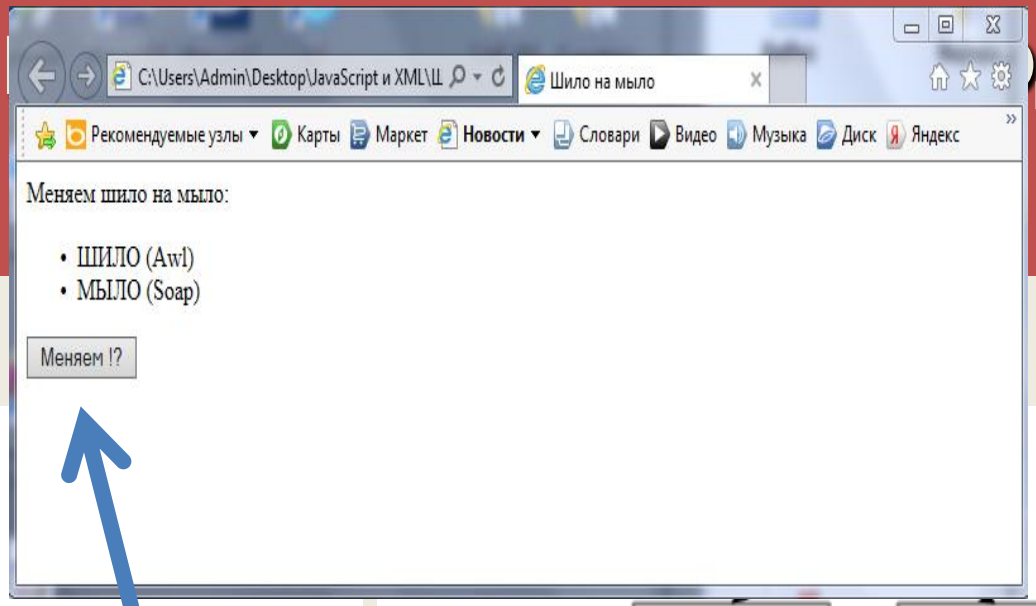
Каждый DOM-элемент является объектом и предоставляет свойства для манипуляции своим содержимым, для доступа к родителям и потомкам.




```
<tree-root>
  <node-level1>
    <node-level2/>
    <node-level2>text</node-level2>
    <node-level2/>
  </node-level1>
<node-level1>
   <node-level2>text</node-level2>
<node-level1>
    <node-level2/>
    <node-level2><node-level3/></node-level2>
</node-level1>
</tree-root>
```

Для манипуляций с DOM используется объект document. Используя document, можно получать нужный элемент дерева и менять его содержание.

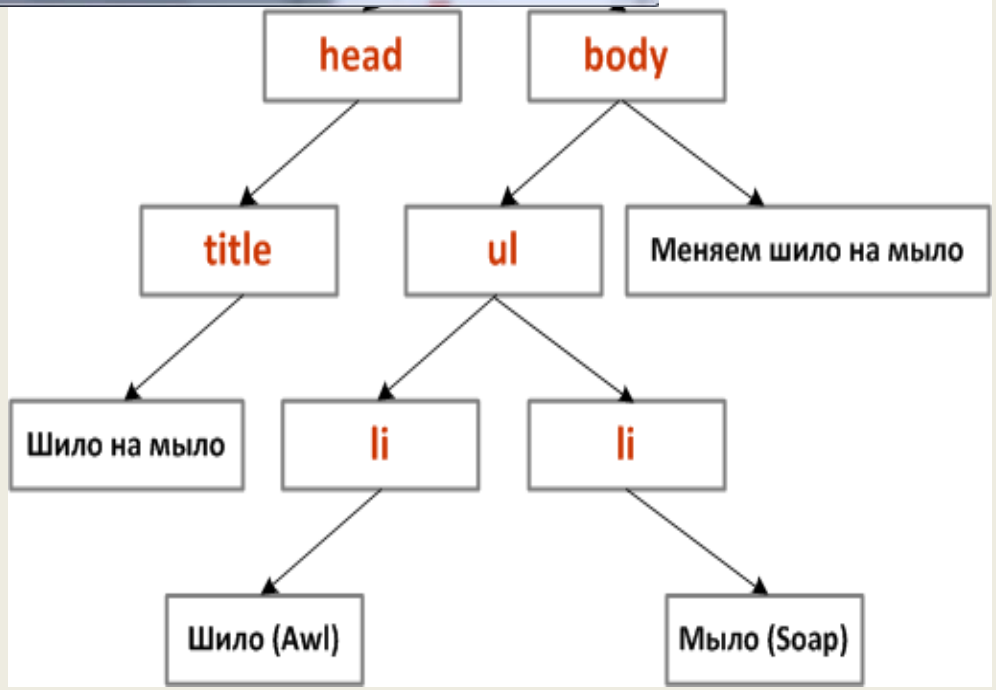
Пример дели




```
<html>
  <head>
    <title>Шило на мыло </title>

      <script type="text/javascript">
        function change()
          {
           var ul = document.getElementsByTagName('ul')[0]
           var a1 = ul.removeChild(ul.firstChild)
           var a2 = ul.removeChild(ul.firstChild)
           ul.appendChild(a2)
           ul.appendChild(a1)
           }
       </script>
  </head>

  <body>Меняем шило на мыло:
  <ul>
       <li>ШИЛО (Awl)</li>
       <li>МЫЛО (Soap)</li>
  </ul>
  <input type="button" value="Меняем !?" onClick=change();>
</body>
</html>
```

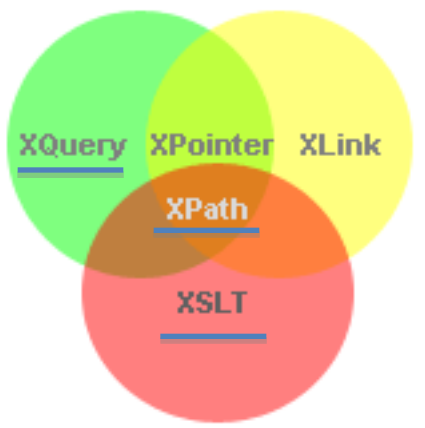


Программные средства для работы с XML

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

- **XML-парсер**:
  - Проверка структуры документов и типов данных, задаваемых DTD/XML Schema
- **XSLT-процессор** (e**X**tensible **S**tylesheet **L**anguage **T**ranformaion - расширяемый язык таблиц стилей):
  - Преобразование XML-документа в другой тип документа (HTML, текстовый и т.д.) https://www.w3schools.com/xml/xml_xslt.asp
- **XML-редактор**:
  - Множество разных
  - Создание, редактирование XML-документов, DTD, XML Schema и т.д.
- **XML-браузер**:
  - Преобразование XML в HTML
  - Реализовано во всех основных - MSIE, Firefox, Opera

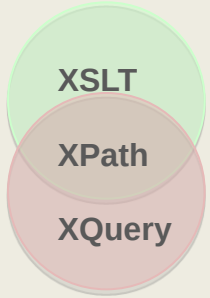
Языки для работы с XML
XSL - языки



XSL

eXtensible Stylesheet Language

Расширяемый Язык Таблицы стилей



XSLT язык для преобразования XML документов
XPath язык для адресации частей XML документа
XQuery язык для запросов к XML документам

XLink язык для гиперссылок в XML документах
XPointer язык для ссылок на отдельные части XML документа

XSLT язык преобразования XML документа

```
<xsl:transform version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">

…   описание шаблона

</xsl:template>
</xsl:transform>
```

transform ≅ stylesheet

Пример: https://www.w3schools.com/xml/xsl_transformation.asp

XPath язык для адресации частей XML документа

XPath язык для адресации отдельных частей XML документа
XPath использует выражения путей для навигации по XML документу
XPath включает библиотеку стандартных функций
XPath важнейшая составляющая часть XSLT и XQuery
XPath является рекомендацией W3C

XPath выражения можно использовать в JavaScript, Java, PHP, Python, C and C++, and lots of other languages.

Пример: https://www.w3schools.com/xml/xml_xpath.asp

XQuery язык для запросов к XML документам

- Язык запросов XML
- Разработан в W3C; первая версия - XQuery 1.0 в 2003г.
- XQuery 3.0 - W3C April 8, 2014
- Надмножество XPath
- Совместим с другими XML-стандартами
- Изначально был предназначен для извлечения информации и не включал средств для модификации существующих документов XML
- XQuery аналог SQL для баз данных
- XQuery поддерживается тремя главными производителями БД (IBM, Oracle, Microsoft), а также многими другими БД

books.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<books>
        <book category="JAVA">
                <title lang="en">Learn Java in 24 Hours</title>
                <author>Robert</author>
                <year>2005</year>
                <price>30.00</price>
        </book>
        <book category="XML">
                <title lang="en">Learn XQuery in 24 hours</title>
                <author>Peter</author>
                <year>2013</year>
                <price>50.00</price>
        </book>
        <book category="XML">
                <title lang="en">Learn XPath in 24 hours</title>
                <author>Jay Ban</author>
                <year>2010</year>
                <price>16.50</price>
        </book>
</books>
```

XQuery

```
let $books := doc("books.xml")
for $x in $books/book
where $x/price < 30
order by $x/price
return $x/title
```

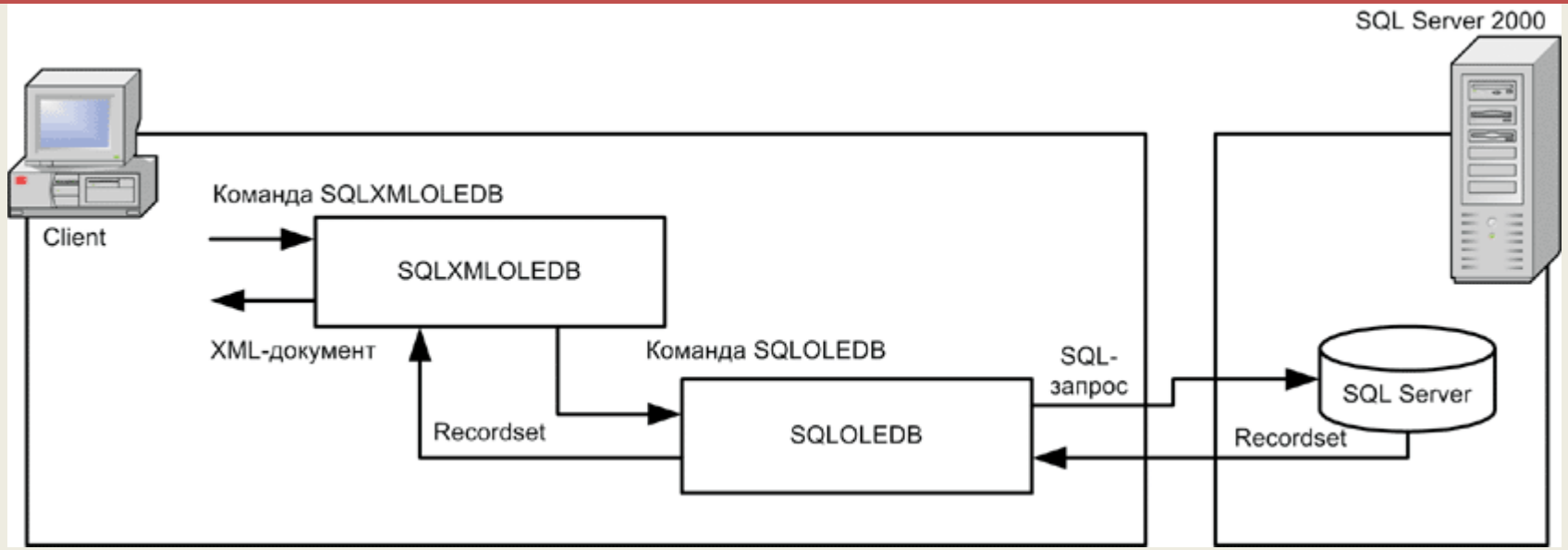
```
<title lang="en">Learn XPath in 24 hours</title>
```

XQuery - FLWOR

Хранение XML в СУБД

- Нативная XML-база данных или традиционная (реляционная) база данных?
  - Открытый вопрос
  - Реляционная модель данных (начало 1970-х годов) – устоявшаяся технология: множество решений, методик, продуктов, множество специалистов, солидный математический и научный базис
  - XML-модель данных (стандарт с 1998г) – первые шаги ...
  - Причины не использовать нативную XML-БД: слабые гарантии производительности при больших и очень больших объемах документов; зачаточные возможности индексирования (над совершенствованием сейчас ведется активная работа)
  - Рекомендация на данный момент: при выборе реляционная БД с (или даже без) поддержкой XML должна иметь приоритет, но**(!)** активно использовать XML как формат описания данных
  - Тем не менее, для ряда приложений нативная XML-БД хороший вариант: в частности, когда требуется интенсивное выполнение запросов к XML-данным

SQL и XML




```
<?xml version="1.0" ?>
<my_root xmlns:sql="urn:schemas-microsoft-com:xml-sql">
   <sql:query client-side-xml="1">
         select count(*) as number_of_authors,city
         from authors
         group by city
         order by number_of_authors
         desc for xml nested
   </sql:query>
</my_root>
```

http://rsdn.ru/article/db/xmlsql.xml

Домашнее задание по XML

По заданной схеме построить XML-документ и проверить его корректность с помощью браузера.