

# **XML - описание технологии**

---

- **Основные сведения**
  - **Стандарты XML**
  - **Работа с XML в .NET. Чтение и запись в XML-файлы.**
- Редактирование XML-файлов**

# Основные сведения об

---

## XML

**XML** (Extensible Markup Language) – это язык разметки документов, позволяющий структурировать информацию разного типа, используя для этого произвольный набор инструкций.

**XML-файл** – это текстовый файл.

# Основные сведения об

## XML

Пример XML-файла:

```
<?xml version="1.0" encoding="UTF-8" ?>
<Издания>
  <Газеты>
    <Кыым Код=«23» Экземпляров=«10000»/>
    <АиФ Код=«90» Экземпляров=«50000»/>
  </Газеты>
  <Журналы>
    <Колокольчик Код=«56» Экземпляров=«5000»/>
  </Журналы>
</Издания>
```

**узлы** (арrows point to <Издания>, <Газеты>, <Журналы>)

**ЗНАЧЕНИЯ** (arrows point to «23», «10000», «90», «50000», «56», «5000»)

**АТТРИБУТЫ** (arrows point to Код, Экземпляров)

# Основные сведения об

---

## XML

### Использование XML

- Роль универсального формата для обмена информацией;
- Базовый стандарт для нового языка описания ресурсов RDF;
- Дополнение к HTML для распространения в Web “нестандартной” информации (XHTML);
- Промежуточный формат данных в трехзвенных системах (приложение -> СУБД).
- Стилиевые таблицы (XSL) обеспечивают независимость отображения XML- документов от устройства вывода.

# Основные сведения об

---

## XML

**DTD** – стандартный способ описания грамматики XML (список допустимых элементов, их возможное содержимое и атрибуты ит.д.).

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

Верифицирующий XML-анализатор при обработке документа будет сверять порядок определения элементов и их атрибутов с тем, что указано в DTD-нотациях.

# Основные сведения об

## XML

```
<?xml version="1.0" encoding="koi-8"?>
```

```
<log>
```

```
<event date=" 27/May/1999:02:32:46 " 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/1999:02:41:47 " result="success">
```

```
<ip-from> 195.209.248.12 </ip-from>
```

```
<method>GET</method>
```

```
<url-to> /soft.htm</url-to>
```

```
<response>200</response>
```

```
</event>
```

```
</log>
```

```
<?xml encoding="koi8-r"?>
```

```
<!ELEMENT log (event)+>
```

```
<!ELEMENT event (ip-from,method,uri-to,result)>
```

```
<!ELEMENT method (#PCDATA)>
```

```
<!ELEMENT ip-from (#PCDATA)>
```

```
<!ELEMENT url-to (#PCDATA)>
```

```
<!ELEMENT response (#PCDATA)>
```

```
<!ATTLIST event
```

```
  result CDATA #IMPLIED
```

```
  date CDATA #IMPLIED>
```

# Стандарты

---

**XML**

**XSLT**

XSLT предназначен для преобразования XML-документов в форму представления традиционного браузера или для обработки XML-файлов с помощью скриптов.

# Стандарты

---

## XML

Для работы с XML применяются **XML-парсеры**:

- **Simple API for XML (SAX)**
- **Document Object Model (DOM).**

**SAX** – основан на курсорах и событиях, возникающих при проходе по узлам XML документа. Нетребователен к ресурсам.

**DOM** – полностью загружает документ в память и представляет его в виде дерева, поэтому можно произвольно перемещаться по XML документу. Требует много памяти.



# Стандарты

---

- XML
- XAPI

В спецификации «Интерфейс прикладного программирования баз данных XML» ([XML Database API \(XAPI\)](#)) описывается нейтральный по отношению к поставщику и языку интерфейс прикладного программирования для баз данных XML.

# Стандарты

---

- **XML**  
SQL/XML

SQL/XML охватывает следующие документы:

- Спецификации для представления данных SQL (в особенности строк и таблиц строк, а также выборок и результатов выполнения запросов) в виде XML и, наоборот.
- Спецификации, связанные с преобразованием схем SQL в схемы XML и, наоборот.
- Спецификации для представления операций SQL (вставить, обновить, удалить).
- Спецификации для передачи сообщений для XML при использовании с SQL.

# Стандарты

---

## XML

### Другие стандарты XML

- CSS – каскадные таблицы стилей.
- XForms – Web-формы для обработки данных XML.
- SOAP – протокол, предназначенный для использования XML для передачи сообщений.
- WSDL – язык описания Web-сервисов.
- XQuery – язык запросов XML.
- XUpdate – предназначен для работы с регулярными XML-документами, а также с XML-документами в совокупностях баз данных и даже с виртуальными моделями данных XML.

# Работа с XML в

---

**.NET**

Сборка **System.Xml.dll**

**XmlTextReader** – класс для чтения из файла XML-элементов.

Перегрузки:

```
XmlTextReader reader = new XmlTextReader();
```

```
XmlTextReader reader = new XmlTextReader(fileName);
```

# Работа с XML в

---

**.NET**

**Чтение XML из**

**файла**

Пример:

```
XmlTextReader reader = new XmlTextReader («fl.xml»);
```

```
while (reader.Read()) //пока считывает элементы
```

```
{
```

```
    // выводим название элемента
```

```
    MessageBox.Show(reader.Name);
```

```
}
```

# Работа с XML в

---

## .NET

Метод **Read()** последовательно перемещается по XML-файлу. При достижении конца файла возвращает FALSE.

### Свойства класса:

**NodeType** – тип элемента

**Name** – имя элемента

**Value** – значение (текст) элемента

# Работа с XML в

## .NET

```
XmlTextReader reader = new XmlTextReader("ToolbarSettings.xml");
while (reader.Read())
{
    if (reader.NodeType == XmlNodeType.Element) //если найден узел
    {
        //Чтение элемента узла
        richTextBox1.Text += "Элемент: " + reader.Name + "\n";

        while (reader.MoveToNextAttribute())
            //Чтение атрибутов узла
            richTextBox1.Text += "Атрибут: " + reader.Name + "\n";

        richTextBox1.Text += "\n";
    }
}
```

# Работа с XML в

---

## .NET

Метод **MovetoNextAttribute()**

последовательно перемещается по всем атрибутам узла.

Свойство **HasAttributes** – используется для проверки наличия атрибутов. Возвращает **true** либо **false**.



# Работа с XML в

## .NET

```
private void button3_Click(object sender, EventArgs e)
{
    XmlTextReader reader = new XmlTextReader("ToolsMenu.xml");
    while (reader.Read())
    {
        if (reader.NodeType == XmlNodeType.Element) //если найден узел
        {
            if (reader.HasAttributes) //если у узла имеются атрибуты
            {
                //Чтение элемента узла
                richTextBox1.Text += "Элемент: " + reader.Name + "\n";

                while (reader.MoveToNextAttribute())
                    //Чтение атрибутов узла
                    richTextBox1.Text += "Атрибут: " + reader.Name + "\n";

                richTextBox1.Text += "\n";
            }
        }
    }
}
```

# Работа с XML в

NET

## Создание XML-

## ДОКУМЕНТОВ

`XmlTextWriter` – класс для создания XML-документов.

`XmlTextWriter writer = new XmlTextWriter(имя файла, кодировка)`

- **WriteStartDocument()** – записывает строку с объявлением версии XML и указанной кодировкой.
- **WriteEndDocument()** – закрывает все открытые теги и атрибуты
- Функции **WriteStartElement(название узла)** и **WriteEndElement()** записывают начало и конец узла
- Функция **WriteAttributeString(название атрибута, значение атрибута)** – записываем атрибут узла
- **WriteElementString()** – создает элемент, содержащий одно текстовое значение, например `<Дата>01.05.01</Дата>`

# Работа с XML в .NET

---

```
XmlTextWriter writer = new XmlTextWriter("e:\\price.xml", Encoding.GetEncoding(1251));
writer.WriteStartDocument();
writer.WriteStartElement("Заказы");

    // сохраняем заказы
writer.WriteStartElement("Заказ");
    writer.WriteAttributeString("Адрес", "Пр.Ленина");
    writer.WriteAttributeString("Дата", "12.04.2013");

    // сохраняем товар
writer.WriteStartElement("Товар");
    writer.WriteAttributeString("Цена", "150");
    writer.WriteAttributeString("Название", "Карандаши");
writer.WriteEndElement();

writer.WriteEndElement();

writer.WriteEndElement();
writer.WriteEndDocument();

writer.Close();
```

# Работа с XML в

## .NET

### Задача

1  
Имеется общая база данных ИМИ. Требуется сформировать XML-файл из данных по группе для загрузки в эту базу данных.

Файл должен содержать сведения:

- о студентах: код, фамилия, имя, размер стипендии;
- об изучаемых предметах: код, название предмета

```
<?xml version="1.0" encoding="WINDOWS-1251"?>
- <ИТ-11>
  - <Студенты>
    <Студент Код="10" Фамилия="Кузьмин" Имя="Павел" Стипендия="5000"/>
  </Студенты>
  - <Предметы>
    <Предмет Код="100" Название="Базы данных"/>
  </Предметы>
</ИТ-11>
```

# Работа с XML в

## NET

```
XmlTextWriter writer = new XmlTextWriter("e:\\it-11.xml", Encoding.GetEncoding(1251));
writer.WriteStartDocument();
writer.WriteStartElement("ИТ-11");

    writer.WriteStartElement("Студенты");
        // сохраняем сведения о студентах
        writer.WriteStartElement("Студент");
            writer.WriteAttributeString("Стипендия", "5000");
            writer.WriteAttributeString("Имя", "Павел");
            writer.WriteAttributeString("Фамилия", "Кузьмин");
            writer.WriteAttributeString("Код", "10");
        writer.WriteEndElement();
    writer.WriteEndElement();

    writer.WriteStartElement("Предметы");
        // сохраняем сведения о предметах
        writer.WriteStartElement("Предмет");
            writer.WriteAttributeString("Название", "Базы данных");
            writer.WriteAttributeString("Код", "100");
        writer.WriteEndElement();
    writer.WriteEndElement();

writer.WriteEndElement();
writer.WriteEndDocument();

writer.Close();
```

# Работа с XML в

## .NET

---

### Редактирование XML- файлов

**XmlDocument** – класс для загрузки XML-документ в память для того, чтобы изменить атрибуты узлов, добавить или удалить новые элементы.

```
XmlDocument document = new XmlDocument();
```

**Load**(FileName) – загружает XML-документ

**XmlNode** – узел в дереве класса **XmlDocument**

**DocumentElement** – корневой документ

**ChildNodes** – все потомки узла

**Save**(FileName) – сохраняет файл

# Работа с XML в

---

## .NET

### Задача

Добавить к созданному файлу сведения о другом студенте.

```
<?xml version="1.0" encoding="WINDOWS-1251"?>
- <ИТ-11>
  - <Студенты>
    <Студент Код="10" Фамилия="Кузьмин" Имя="Павел" Стипендия="5000"/>
    <Студент Код="20" Фамилия="Иванов" Имя="Петр" Стипендия="200"/>
  </Студенты>
  - <Предметы>
    <Предмет Код="100" Название="Базы данных"/>
  </Предметы>
</ИТ-11>
```

# Работа с XML в

## NET

```
XmlDocument document = new XmlDocument();  
document.Load("e:\\it-11.xml");
```

```
XmlNode root = document.DocumentElement;  
foreach (XmlNode node in root.ChildNodes) //цикл по дочерним узлам  
{  
    if (node.Name == "Студенты") //если нужный узел найден  
    {  
        XmlElement elem = document.CreateElement("Студент"); //создаем элемент  
        node.AppendChild(elem); //добавляем его в узел "Студенты"  
  
        XmlAttribute attribute1 = document.CreateAttribute("Стипендия"); // создаём атрибут  
        attribute1.Value = "200"; // устанавливаем значение атрибута  
        elem.Attributes.Append(attribute1); // добавляем атрибут  
  
        XmlAttribute attribute2 = document.CreateAttribute("Имя"); // создаём атрибут  
        attribute2.Value = "Петр"; // устанавливаем значение атрибута  
        elem.Attributes.Append(attribute2); // добавляем атрибут  
  
        XmlAttribute attribute3 = document.CreateAttribute("Фамилия"); // создаём атрибут  
        attribute3.Value = "Иванов"; // устанавливаем значение атрибута  
        elem.Attributes.Append(attribute3); // добавляем атрибут  
  
        XmlAttribute attribute4 = document.CreateAttribute("Код"); // создаём атрибут  
        attribute4.Value = "20"; // устанавливаем значение атрибута  
        elem.Attributes.Append(attribute4); // добавляем атрибут  
    }  
}  
document.Save("E:\\text2.xml"); //сохраняем изменения
```



# Лабораторная работа 11

---

Создайте приложение для работы с XML-файлами, которое должно содержать три метода:

1. Создание XML-файла.
2. Добавление сведений в уже существующий XML-файл.
3. Вывод данных из выбранного XML-файла.