**Работа С# с xml документом:**

**Работа с XML в C# XML-документы:**

На сегодняшний день XML является одним из распространенных стандартов документов, который позволяет в удобной форме сохранять сложные по структуре данные. Поэтому разработчики платформы .NET включили в фреймворк широкие возможности для работы с XML.

Прежде чем перейти непосредственно к работе с XML-файлами, сначала рассмотрим, что представляет собой xml-документ и как он может хранить объекты, используемые в программе на c#.

Например, у нас есть следующий класс:

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | class User  {      public string Name { get; set; }      public int Age { get; set; }      public string Company { get; set; }  } |

В программе на C# мы можем создать список объектов класса User:

|  |  |
| --- | --- |
| 1  2  3 | User user1 = new User { Name = "Bill Gates", Age = 48, Company = "Microsoft" };  User user2 = new User { Name = "Larry Page", Age = 42, Company = "Google" };  List<User> users = new List<User> { user1, user2 }; |

Чтобы сохранить список в формате xml мы могли бы использовать следующий xml-файл:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11 | <?xml version="1.0" encoding="utf-8" ?>  <users>    <user name="Bill Gates">      <company>Microsoft</company>      <age>48</age>    </user>    <user name="Larry Page">      <company>Google</company>      <age>48</age>    </user>  </users> |

XML-документ объявляет строка <?xml version="1.0" encoding="utf-8" ?>. Она задает версию (1.0) и кодировку (utf-8) xml. Далее идет собственно содержимое документа.

XML-документ должен иметь один единственный корневой элемент, внутрь которого помещаются все остальные элементы. В данном случае таким элементом является элемент <users>. Внутри корневого элемента <users> задан набор элементов <user>. Вне корневого элемента мы не можем разместить элементы user.

Каждый элемент определяется с помощью открывающего и закрывающего тегов, например, <user> и </user>, внутри которых помещается значение или содержимое элементов. Также элемент может иметь сокращенное объявление: <user /> - в конце элемента помещается слеш.

Элемент может иметь вложенные элементы и атрибуты. В данном случае каждый элемент user имеет два вложенных элемента company и age и атрибут name.

Атрибуты определяются в теле элемента и имеют следующую форму: название="значение". Например, <user name="Bill Gates">, в данном случае атрибут называется name и имеет значение Bill Gates

Внутри простых элементов помещается их значение. Например, <company>Google</company> - элемент company имеет значение Google.

Названия элементов являются регистрозависимыми, поэтому <company> и <COMPANY> будут представлять разные элементы.

Таким образом, весь список Users из кода C# сопоставляется с корневым элементом <users>, каждый объект User - с элементом <user>, а каждое свойство объекта User - с атрибутом или вложенным элементом элемента <user>

Что использовать для свойств - вложенные элементы или атрибуты? Это вопрос предпочтений - мы можем использовать как атрибуты, так и вложенные элементы. Так, в предыдущем примере вполне можно использовать вместо атрибута вложенный элемент:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | <?xml version="1.0" encoding="utf-8" ?>  <users>    <user>      <name>Bill Gates</name>      <company>Microsoft</company>      <age>48</age>    </user>    <user>      <name>Larry Page</name>      <company>Google</company>      <age>48</age>    </user>  </users> |

**Пример инициализации xml –файла.Данные в xml файл добавлены вручную.**

private void LoadEmployees()

{

XmlDocument doc = new XmlDocument();

doc.Load("xmltext.xml");

foreach(XmlNode node in doc.DocumentElement)

{

string name = node.Attributes[0].Value;

int age = int.Parse(node["Age"].InnerText);

bool programmer = bool.Parse(node["Programmer"].InnerText);

listBox1.Items.Add(new Employee(name, age, programmer));

}

}

Пример части кода добавления поездов и пассажиров.

**Добавление поездов:**

//ячейка пассажира

public struct Passenger

{

public string Name;

public string Surname;

public string Otchestvo;

public string DepartureCity;

public string ArrivalCity;

}

public List<Passenger> MainPassList = new List<Passenger>();// лист всех пассажиров

// маршрут поезда

public struct Route

{

public string City;//город

public string Departure;//прибытие

public string Arrival;//отправление

}

//ячейка поезда

public struct Train

{

public string numb;// номер поезда

public string DepartureCity\_ArrivalCity;//начальный город - конечный город поезда

public string DepartureTime;//отправление поезда

public string ArrivalTime;//прибытие поезда

public IList<Route> RouteTrain;// маршрут поезда

}

//ячейка пассажира

public struct Passenger

{

public string Name;

public string Surname;

public string Otchestvo;

public string DepartureCity;

public string ArrivalCity;

}

namespace Laba4

{

public partial class Form1 : Form

{

public List<Train> MainTrainList = new List<Train>(); //лист всех поездов

public List<Passenger> MainPassList = new List<Passenger>();// лист всех пассажиров

public Form2 f2 { get; set; }

public Form3 f3 { get; set; }

public Form1()

{

InitializeComponent();

Trains.Visible = false;

People.Visible = false;

Route\_data\_grid.Visible = false;

f3 = new Form3(this);

информацияToolStripMenuItem.Visible = false;

поискToolStripMenuItem.Visible = false;

}

private void Form1\_Load(object sender, EventArgs e)

{

Form2 form2 = new Form2();

form2.f1 = this;

label4.Visible = false;

label5.Visible = false;

}

//начальная кнопка выбора поездов

private void button1\_Click(object sender, EventArgs e)

{

// выход, если была нажата кнопка Отмена и прочие (кроме ОК)

if (openFileDialog1.ShowDialog() != DialogResult.OK)

{

return;

}

//имя файла теперь хранится в openFileDialog1.FileName

MessageBox.Show("Файл выбран: " + openFileDialog1.FileName);

if (openFileDialog1.FileName != "Файл не найден!")

{

try

{

//добавление из xml файла в датагрид(на форму)

TrainsDataSet.ReadXml(openFileDialog1.FileName);

Trains.DataSource = TrainsDataSet;

Trains.DataMember = "Train";

Route\_data\_set.ReadXml(openFileDialog1.FileName);

Route\_data\_grid.DataSource = Route\_data\_set;

Route\_data\_grid.DataMember = "Item";

button1.Visible = false;

Trains.Visible = true;

поискToolStripMenuItem.Visible = true;

информацияToolStripMenuItem.Visible = true;

label3.Visible = false;

label5.Visible = true;

}

catch (ArgumentException)

{

MessageBox.Show("Неверный XML файл.");

TrainsDataSet.Clear();

TrainsDataSet.Reset();

Route\_data\_set.Reset();

Route\_data\_set.Clear();

Trains.DataSource = null;

Route\_data\_grid.DataSource = null;

button1\_Click(sender, e);

}

catch (XmlException)

{

MessageBox.Show("Неверный XML файл.");

TrainsDataSet.Clear();

TrainsDataSet.Reset();

Route\_data\_set.Reset();

Route\_data\_set.Clear();

Trains.DataSource = null;

Route\_data\_grid.DataSource = null;

button1\_Click(sender, e);

}

}

//функция добавления в гланый лист

AddTrainXML();

}

//функция начального добавления поездов из xml

public void AddTrainXML()

{

MainTrainList.Clear();

int gcount = 0;

Train Item = new Train();

for (int i = 0; i < Trains.Rows.Count; i++)

{

//добавляем все критерии в айтем для листа

Item.RouteTrain = new List<Route>();

Item.numb = Convert.ToString(Trains.Rows[i].Cells[4].Value);

Item.DepartureTime = Convert.ToString(Trains.Rows[i].Cells[0].Value);

Item.ArrivalTime = Convert.ToString(Trains.Rows[i].Cells[1].Value);

Item.DepartureCity\_ArrivalCity = Convert.ToString(Trains.Rows[i].Cells[2].Value) + " - ";

Item.DepartureCity\_ArrivalCity += Convert.ToString(Trains.Rows[i].Cells[3].Value);

//добавляем маршрут

Route ItemRoute = new Route();

for (int j = gcount; j < Route\_data\_grid.Rows.Count; j++)

{

if (Convert.ToString(Route\_data\_grid.Rows[j].Cells[1].Value) != "")

{

ItemRoute.City = Convert.ToString(Route\_data\_grid.Rows[j].Cells[0].Value);

ItemRoute.Departure = Convert.ToString(Route\_data\_grid.Rows[j].Cells[1].Value);

ItemRoute.Arrival = Convert.ToString(Route\_data\_grid.Rows[j].Cells[2].Value);

Item.RouteTrain.Add(ItemRoute);

}

else

{

ItemRoute.City = Convert.ToString(Route\_data\_grid.Rows[j].Cells[0].Value);

ItemRoute.Arrival = Convert.ToString(Route\_data\_grid.Rows[j].Cells[2].Value);

ItemRoute.Departure = Convert.ToString(Route\_data\_grid.Rows[j].Cells[1].Value);

gcount = j + 1;

Item.RouteTrain.Add(ItemRoute);

break;

}

}

//всовываем ячейку в главный лист

MainTrainList.Add(Item);

}

}

//добавить поезд через загрузку файла

public void поездToolStripMenuItem\_Click(object sender, EventArgs e)

{

OpenFileDialog openFileDialog2 = new OpenFileDialog();

// выход, если была нажата кнопка Отмена и прочие (кроме ОК)

if (openFileDialog2.ShowDialog() != DialogResult.OK)

return;

//имя файла теперь хранится в openFileDialog1.FileName

MessageBox.Show("Файл выбран: " + openFileDialog2.FileName);

if (openFileDialog2.FileName != "Файл не найден!")

{

try

{

//добавление из xml файла в датагрид(на форму)

XmlDocument xdok = new XmlDocument();

xdok.Load(openFileDialog2.FileName);

if (xdok.SelectSingleNode("//Train") == null)

{

MessageBox.Show("Неверный XML файл.");

поездToolStripMenuItem\_Click(sender, e);

return;

}

TrainsDataSet.ReadXml(openFileDialog2.FileName);

Trains.DataSource = TrainsDataSet;

Trains.DataMember = "Train";

Route\_data\_set.ReadXml(openFileDialog2.FileName);

Route\_data\_grid.DataSource = Route\_data\_set;

Route\_data\_grid.DataMember = "Item";

button1.Visible = false;

Trains.Visible = true;

}

catch (XmlException)

{

MessageBox.Show("Неверный XML файл.");

поездToolStripMenuItem\_Click(sender, e);

}

}

AddTrainXML();

}

public void AddPassXML()

{

MainPassList.Clear();

for (int i = 0; i < People.Rows.Count; i++)

{

Passenger Item = new Passenger();

Item.Surname = Convert.ToString(People.Rows[i].Cells[0].Value);

Item.Name = Convert.ToString(People.Rows[i].Cells[1].Value);

Item.Otchestvo = Convert.ToString(People.Rows[i].Cells[2].Value);

Item.DepartureCity = Convert.ToString(People.Rows[i].Cells[3].Value);

Item.ArrivalCity = Convert.ToString(People.Rows[i].Cells[4].Value);

MainPassList.Add(Item);

}

}

//добавить пассажира через файл

private void указатьФайлToolStripMenuItem\_Click(object sender, EventArgs e)

{

// выход, если была нажата кнопка Отмена и прочие (кроме ОК)

if (openFileDialog1.ShowDialog() != DialogResult.OK)

return;

//имя файла теперь хранится в openFileDialog1.FileName

MessageBox.Show("Файл выбран: " + openFileDialog1.FileName);

if (openFileDialog1.FileName != "Файл не найден!")

{

try

{

DataSet dset = new DataSet(); // создаем новый пустой кэш данных

dset.ReadXml(openFileDialog1.FileName); // записываем в него XML-данные из файла

foreach (DataRow Item in dset.Tables["Person"].Rows)

{

int n = People.Rows.Add(); // добавляем новую сроку в dataGridView1

People.Rows[n].Cells[0].Value = Item["Surname"]; // заносим в первый столбец созданной строки данные из первого столбца таблицы ds.

People.Rows[n].Cells[1].Value = Item["Name"]; // то же самое со вторым столбцом

People.Rows[n].Cells[2].Value = Item["Otchestvo"]; // то же самое с третьим столбцом

People.Rows[n].Cells[3].Value = Item["Departure"];

People.Rows[n].Cells[4].Value = Item["Arrival"];

}

button1.Visible = false;

People.Visible = true;

label4.Visible = true;

}

catch (XmlException)

{

MessageBox.Show("Выбран неверный файл.");

указатьФайлToolStripMenuItem\_Click(sender, e);

}

catch (NullReferenceException)

{

MessageBox.Show("Неверный XML файл.");

указатьФайлToolStripMenuItem\_Click(sender, e);

}

}

AddPassXML();

}

public void добавитьВручнуюToolStripMenuItem\_Click(object sender, EventArgs e)

{

Form2 form2 = new Form2();

form2.ShowDialog();

if (form2.check\_add)

{

int n = People.Rows.Add();

People.Rows[n].Cells[0].Value = form2.textBox2.Text;

People.Rows[n].Cells[1].Value = form2.textBox1.Text;

People.Rows[n].Cells[2].Value = form2.textBox3.Text;

People.Rows[n].Cells[3].Value = form2.textBox4.Text;

People.Rows[n].Cells[4].Value = form2.textBox5.Text;

People.Visible = true;

label4.Visible = true;

AddPassXML();

}

}

}

}