**Лаб. работа \_\_\_\_\_**

**Чтение, запись текстовых и бинарных файлов, текстовый редактор**

**Часть 1. Чтение/запись текстового файла в кодировке *Unicode.* Обработка исключений *try...catch***

Очень распространенной задачей является сохранение данных на диске **в текстовом** (не двоичном) формате.

Понятно, что такое деление условно, поскольку и текстовые, и не­текстовые файлы на самом деле являются двоичными файлами. Но если нетекстовые файлы открыть, например, Блокнотом, мы увидим то, что называют ’’нечитаемым ме­сивом” - отсюда такая классификация.

Часто данные сохраняют на диск именно в тек­стовом формате, поскольку в этом случае сохраненные данные можно читать, редакти­ровать *любым текстовым редактором*, например, Блокнотом, или TextEdit, или другими.

Следует также уметь и читать текстовые данные *в пользовательской программе*.

Казалось бы, это очень простая задача.

Например, **чтение** текстового файла сводится буквально к нескольким строчкам:

// Создание экземпляра StreamReader для чтения из файла:

**var** R = **new** System.10.StreamReader( @"D:\Textl.txt" );

// Непосредственное чтение всего файла в текстовое поле:

textBoxl.Text = R.ReadToEnd();

Читатель.Close(); // закрытие файла

Однако есть некоторые серьезные нюансы. Создадим проект C#, содержащий текстовое поле и две командные кнопки (рис. 1). Назовем проект TxtUnicode.

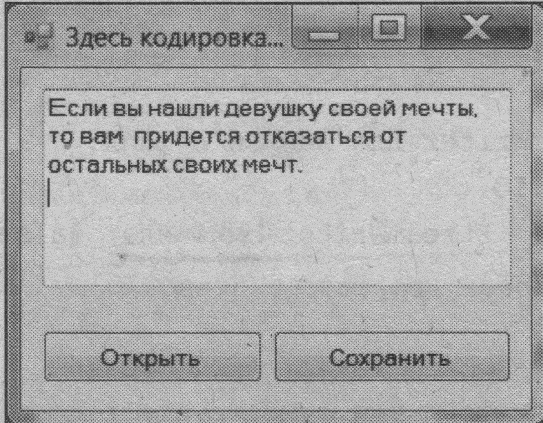


Рис. 1. Форма проекта

В форму поместим текстовое поле и две командные кнопки. Для текстового поля в окне **Properties** сразу укажем для свойства Multiline значение True. Одна кнопка предназначена для открытия файла, а другая - для его со­хранения.

Ниже приведены коды этой программы.

**Листинг 1. Чтение/запись текстового файла в кодировке Unicode**

**// Программа для чтения/записи текстового файла в кодировке Unicode**

**using** System;

**using** System.Drawing;

**using** System.Windows.Forms;

**// Другие директивы using удалены, поскольку они не используются в данной программе**

**namespace** TxtUnicode

{

**public partial class** Forml : Form

{

**// Объявляем ИмяФайла на уровне класса**

**String** ИмяФайла;

**public Forml()**

{

nitializeComponent();

}

**private void** Forml\_Load(**object** sender, EventArgs e)

{

**// Установка начальных значений:**

textBoxl.Multiline = true; textBoxl.Clear();

textBoxl.Size = new Size (268, 112);

buttonl.Text = "Открыть"; buttonl.Tablndex = 0;

button2.Text = "Сохранить";

this.Text = "Здесь кодировка Unicode";

ИмяФайла = 0"D:\Textl.txt";

}

**private void** buttonl\_Click(object sender, EventArgs e)

{

**// Щелчок на кнопке Открыть.**

**// Если открыть файл в кодировке UNICODE,**

**// русские буквы будут читаться корректно**

**try**

{ **// Создание объекта StreamReader для чтения из файла:**

**var** Читатель = **new** System.10.StreamReader(ИмяФайла);

**// Чтение всего файла в текстовое поле:**

textBoxl.Text = Читатель.ReadToEnd();

Читатель.Close(); **// закрытие файла**

**// Можно читать текстовый файл в кодировке UNICODE в массив**

**// строк также таким образом (без Open и Close). Например, так:**

**// var МассивСтрок = System.10.File.ReadAllLines(@"D:\Textl.txt");**

}

**catch** (System.I0.FileNotFoundException **Исключение**)

{

// Обработка исключительной ситуации:

MessageBox.Show(Ситуация.Message + "\n" +

"Такого файла не существует ", "Ошибка",

MessageBoxButtons.OK,

**MessageBoxIcon.Exclamation);**

}

**catch** (Exception **Исключение**)

**// Отчет о других ошибках:**

MessageBox.Show(Ситуация.Message, "Ошибка",

MessageBoxButtons.OK, MessageBoxIcon.Exclamation);

}

}

**private void** button2\_Click(object sender, EventArgs e)

{

**// Click по кнопке Сохранить:**

**try**

**{// Создание объекта StreamWriter для записи в файл:**

**var** Писатель = **new** System.I0.

StreamWriter(ИмяФайла, **false)**;

Писатель.Write(textBoxl.Text);

Писатель.Close();

**// Сохранить текстовый файл можно также таким образом**

**// (без Close), причем если файл уже существует, то он будет заменен:**

**/ / System.10.File.WriteAllText( @"D:\tmp.tmp", textBoxl.Text);**

}

**catch** (Exception Исключение)

{

**// Отчет обо всех возможных ошибках:**

MessageBox.Show(Исключение.Message, "Ошибка",

MessageBoxButtons.OK, MessageBoxIcon.Exclamation);

}

}// Конец **private void** button2

}// Конец класса

}// Конец пространства имен

Итак, обработка ис­ключительной ситуации свелась к информированию пользователя о недоразу­мении.

При обработке события “Click по кнопке **Открыть"** организовано открытие файла D:\Textl.txt.

Обычно в такой ситуации для выбора файла применяют элемент управле­ния **OpenFileDialog**. Пока мы не стали использовать этот элемент чтобы свести к минимуму программный код.

Далее для чтения из файла создали объект (поток) **Читатель**.

За­тем следует чтение файла ИмяФайла методом ReadToEnd() втекстовое поле textBoxl.Text и закрытие файла методом Close ().

При обработке события "Click по кнопке **Сохранить"** аналогично организована запись файла на диск через объект Писатель. При создании объекта писатель первым аргументом является ИмяФайла, а второй аргумент (**false)** указывает, что данные следует *не добавить* (**append**) к содержимому файла (если он уже существует), а *перезаписать* (overwrite).

Запись на диск производится с помощьюметода Write () из свойства Text элемента управления TextBoxl.

**Внимание!**

Запись и чтение текстового файла с помощью данной программы будет осуществляться в *формате (кодировке)* ***Unicode***.

То есть, вы сможете читать эти файлы при помощи Блокно­та, редактировать их, но каждый раз при сохранении из Блокнота вам потребуется следить, чтобы кодировка была (оставалась) **Unicode**.

В редакторах используется как кодировка ANSI, так и Unicode. Кодировку ANSI, включающую русские буквы, называют еще Windows 1251. Некоторые редакторы, на­пример RPad ("русский" Блокнот), вообще не работают с **Unicode**.

**Об отображении кириллицы**

Если в Блокноте подготовить текстовый файл в обычной кодировке ANSI, то прочитать рус­ские буквы приведенной выше программой не получится, хотя английские буквы отобразятся в тексто­вом поле *без проблем.*

Почему?

Дело в том, что выше примененная **техноло­гия** описана в учебных пособиях по С#, в MSDN, на сайтах, подобных <http://msdn.microsoft.com/library/rus/>.

Чаще всего - это англоязычные источники, а для английских текстов переход от одной кодировки к другой оказывается практически неза­метным. Например, английские буквы кодировок Windows 1251, ASCII и Unicode совпадают. Но с русскими же буквами всегда возникают недоразумения.

Программистам такие недоразумения следует уметь учитывать.