**Урок 58: Файлове**

Едно от важните приложения на компютрите в ежедневието е за бързо обработване на огромно количество информация. Но данните за такава обработка трябва да дойдат отнякъде и да бъдат въведени в компютъра. Ако те са голямо количество, не е подходящо да се използва графично приложение, тъй като трябва да се създадат твърде голям брой текстови полета. От друга страна, ако се използва интерфейса на конзолно приложение, то въвеждането на данните може да се наложи да се направи многократно, тъй като има вероятност да се допусне грешка, която не може да бъде поправена по никакъв начин, освен чрез започване на въвеждането отново. И двата варианта, разбира се, са много неприятни и неприемливи. Затова в езиците за програмиране е предвидена възможността програмата както да чете входните си данни от файл, така и да извежда изходните си данни във файл. Това не е винаги необходимо, но си представете колко много данни има за учениците от едно училище и колко по-лесно би било, ако те са записани във файлове, които компютрите да обработват.

От изучаваните до този момент приложни програми знаем, че те работят с файлове и имат менюта за отваряне, съхраняване и затваряне на файл. В програми на езика C# можем да направим същото, като за целта файловете се разглеждат като **потоци**.

**1. Потоци от данни**

**а) същност** – подредени последователности от байтове, които се изпращат от едно устройство и се получават в друго устройство. Потоците позволяват само последователен достъп до данните.

**б) дефиниране**

Всички потоци в езика C# се намират в пространството от имена **System.IO**, което означава, че ако искаме да използваме файлове, в началото на програмите трябва да добавяме директивата: **using System.IO;** Езикът C# предлага различни потоци, подходящи за различни дейности (BufferedStream, MemoryStream, Stream, FileStream, NetworkStream и др.), но в повечето програми се използват текстови потоци, които позволяват да се четат и извеждат данни в текстови файлове.

**в) физически и логически файл**

* **физически файл** – реалният файл, разположен на носител на данни
* **логически файл** – образ на файла в програмата

**2. Компонен RichTextBox**

Този компонент на графичния интерфейс се нарича **многоредова текстова кутия.** Тя се различава от компонента **TextBox** по това, че в нея може да се извежда текст на повече от един ред. Указание за преминаване на нов ред е низът **„\n**“, поставен в текста ю, който искаме да изведем.

**3. Четене от текстов файл**

* чрез променлива от клас **StreamReader** „запознаваме“ компютъра с файла, от който ще се чете

*StreamReader r = new StreamReader(име на файл, Encoding.GetEncoding("windows-1251"));*

Ако файлът не се намира в текущата папка на изпълнимия файл на програмата, тогава трябва да се зададе и пътят до него, като е препоръчително този път да започва от папката на изпълнимия файл – това прави програмата по-независима при евентуалния ѝ пренос на друг компютър.

* четене на данните от файла – чрез променливата r, която играе ролята на логически файл и може да стане по два начина.
* При ***първия начин*** програмата прочита целия текстов файл чрез метода **ReadToEnd()** на класа StreamReader наведнъж като един низ и след това може да бъде обработван или показан в интерфейсния елемент RichTextBox:

*string textFile = r.ReadToEnd(); richTextBox1.Text = textFile;*

Този вариант се предпочита, когато файлът е под 1 MB и не се налага обработка ред по ред

* При ***втория вариант*** файлът се чете ред по ред чрез метода **ReadLine()** на класа StreamReader, който работи по същия начин като метода ReadLine() на класа Console. За целта трябва да се използва оператор за цикъл. Най-подходящ е оператор while, тъй като обикновено не се знае от колко реда е файлът. Когато се достигне края на файла се прочита неговия знак за край **null** и това може да се използва за условие за край на цикъла. Променливата, която ще се използва в цикъла трябва да бъде от тип низ, тъй като текстовият файл е последователност от редове от тип низ:

*string line = "";*

*while(line!=null)*

*{*

*line = r.ReadLine();*

*richTextBox1.Text = richTextBox1.Text+line + "\n";*

*}*

Този вариант се използва, когато искаме да обработваме входните данни ред по ред.

* затваряне на файла - след като приключим работата с файла трябва да го затворим чрез метода **Close()** на класа StreamReader:

*r.Close();*

Ако искаме да започнем да четем файла отначало, трябва да го отворим отново.

**4. Записване в текстов файл**

Понякога се налага изходните данни от една програма да бъдат записани във файл – за да бъдат разпечатани, препратени до друг компютър, да бъдат използвани като входни данни в друга програма и т.н. Записването в текстов файл също е последователно.

* свързваме променлива от клас **StreamWriter** с физическия файла, в който се записваме:

*StreamWriter w = new StreamWriter(,, Encoding.GetEncoding("windows-1251"));*

* записване на поредния ред във файла – извикване се метода WriteLine, като за аргумент задаваме низа, който трябва да се запише във файла:

*w.WriteLine("Николай Петков");*

* затваряне на файла - след приключване на записването във файл, задължително трябва да затворим файла чрез метода **Close()** на класа StreamWriter:

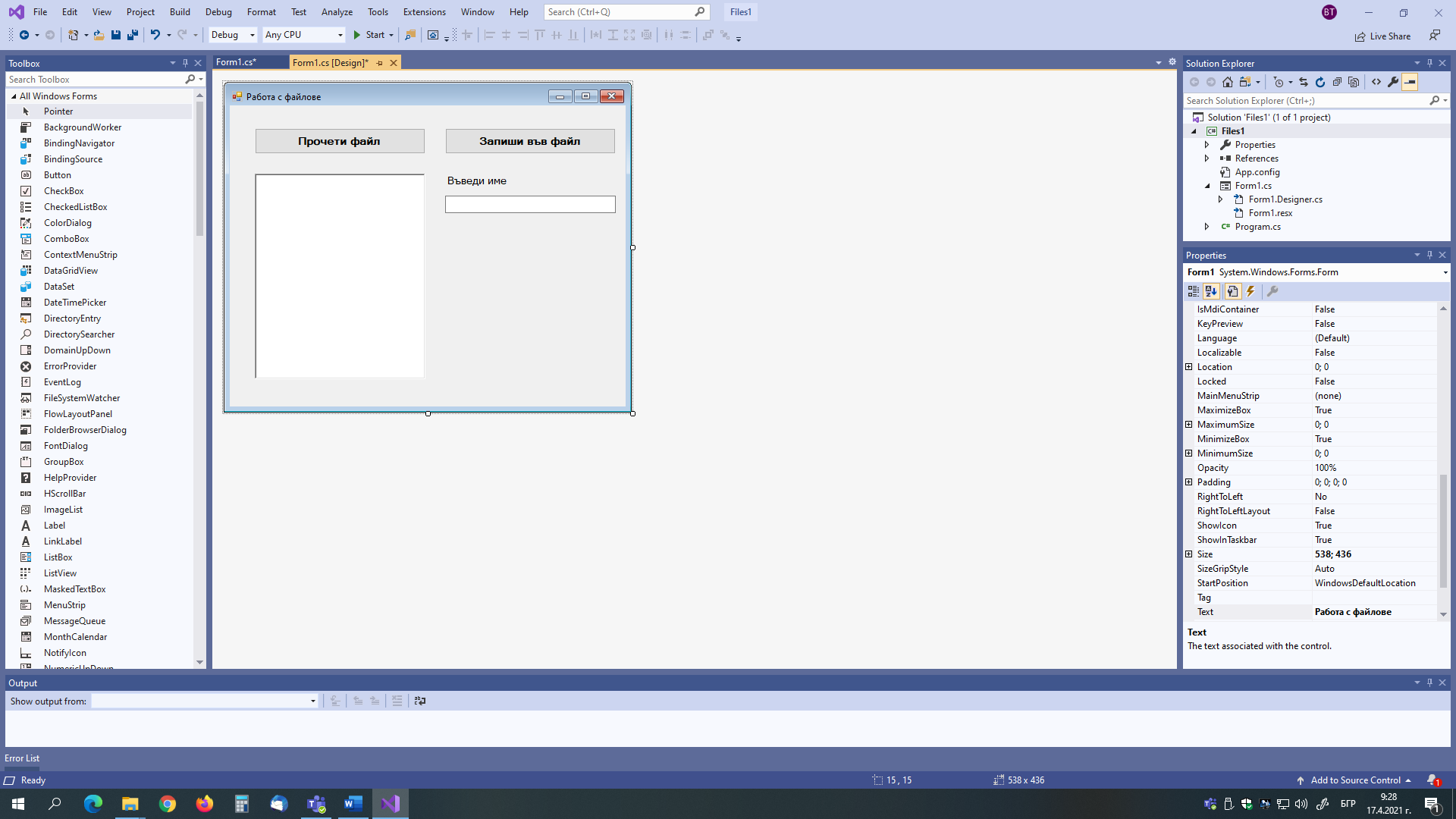
*r.Close();*

По този начин във файла се добавя знак за край на файла и се извършва съхраняването му. Ако не затворим файла, тогава няма гаранция, че изпратеното действително е записано в него.

**5. Работа с компютър** – Създайте графично приложение с име **Files**, в което:

* да прочетете имената от файл с име ***Name.txt*** и да ги изведете в многоредова текстова кутия RichTextBox
* да въвеждате имена чрез текстова кутия TextBox и да ги записвате във файла с име ***Name.txt***

Примерен дизайн:



//четене от файл

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

{

StreamReader r = new StreamReader("Name.txt",Encoding.GetEncoding("UTF-8"));

string line = "";

while (line != null)

{

line = r.ReadLine();

richTextBox1.Text = richTextBox1.Text + line + "\n";

}

r.Close();

}

//записване във файл

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

{

StreamWriter w = new StreamWriter("Name.txt", true, Encoding.GetEncoding("UTF-8"));

w.WriteLine( textBox1.Text);

w.Close();

}

**Дом.: Задача 1:** от урока на стр.103

**Задача 2:** стр. 105