INN570 2013 Week 5 Prac Exercises

Encoding and Decoding

In this exercise, we will continue with the work from week 3, observe the effect of different encodings and will try to use different methods to decoding.

1 Encoding

From week 3 examples, we have noticed that different encoding (e.g., UTF-8, ANSI) can influence the output of text files. In the following exercise, we will observe the encoding influence to the text files in .NET.

- Open the Birthday Book application *BBookWk2*, run the application, use Chinese Input method to enter '*shan*(山)*mu*(姆)', and '*zhan*(詹)*mu*(姆)*shi*(士)' in the column Name, and enter dates in the column Birthday, similar to what you have done last week.
  Then save the file. You will find that the content of the file is correct even you didn't choose encoding. This is because by default, .NET encoding is UTF-8.

- Double click on the Birthday Book form to open its corresponding code in the code editor. The following line of code in the *saveToolStripMenuItem_Click* ( ) method creates an object of *StreamWriter* (which is in the *if* statement) without explicitly specifying its encoding. Its default encoding is UTF-8.

  ```
  StreamWriter MyStream = new StreamWriter(SaveDialog.FileName);
  ```

  Of course, you can explicitly specify UTF-8 encoding something like:

```
StreamWriter MyStream = new StreamWriter(SaveDialog.FileName, false,
Encoding.UTF8);
```

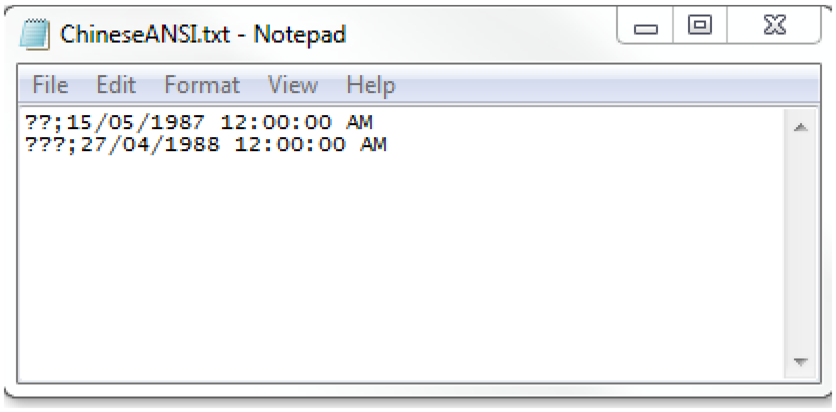
- However, if you do want to save the file in ASCII code. You can explicitly specify the encoding by replacing `StreamWriter MyStream = new StreamWriter(SaveDialog.FileName)` with the following code:

```
StreamWriter MyStream = new StreamWriter(SaveDialog.FileName, false,
Encoding.ASCII);
```

  This code sets encoding as ASCII.

Now you run the program again, enter ‘*shan*(山)*mu*(姆)’, and ‘*zhan*(詹)*mu*(姆)*shi*(士)’ and some dates, then save the file with a file “ChineseANSI.txt”.

Open the file “ChineseANSI.txt” using a text editor such as Notepad, you can find a similar result which you have seen last week, something like:



```
??;15/05/1987 12:00:00 AM
???;27/04/1988 12:00:00 AM
```

Here we just want to show the effect of different encodings. In most cases, you may just like to use the default encoding setting.

2 Decoding

Now we want to add some names in different languages onto the Birthday book. These names are provided in a text file with encoding UFT-8. The text file *inputuft8.txt* is supplied with this week prac material. Add the text file in folder BBookWk2->bin->Debug .

We need to modify the program to read the sentences from the text file and add them onto the Birthday table. We will use three different ways to read the file.

- Use *FileStream*
  Double click on the form Birthday Book to open the corresponding code. At the end of the method *BBook_load( ),* add the following code. This code uses a *FileStream* object to read the sentences in byes from the input file. The result is a block of bytes rather than strings of characters. In this case we need decoding to get the strings of the sentences.

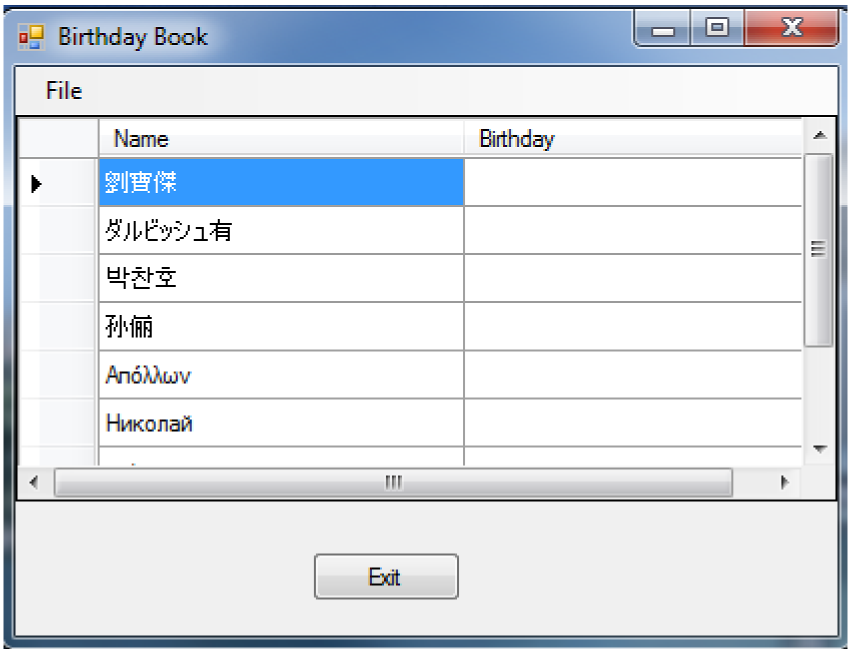
```
//read bytes, decoding, and then add to the table
FileStream fsSource = new FileStream("inpututf8.txt",
                                     FileMode.Open, FileAccess.Read);
byte[] bytes = new byte[fsSource.Length];
fsSource.Read(bytes, 0, bytes.Length);

string contents;
```

```
//decoding to get a string of character
contents = Encoding.UTF8.GetString(bytes);
//split the string into an array of strings
//each string in the array is a sentence from the input file
string [] sentences=contents.Split(new Char[]{'\n'});

//get each sentence and add it onto the birthday table
for (int j = 0; j < sentences.Length; j++)
{
    DataRow new_row = m_BBookTable.NewRow();
    new_row["Name"] = sentences[j];
    m_BBookTable.Rows.Add(new_row);
}
fsSource.Close();
```

Run the program, the following table will be produced.



	Name	Birthday
▶	劉寶傑	
	ダルビッシュ有	
	박찬호	
	孙俪	
	Απόλλων	
	Николай	

- Use *Decoder*

Similar to *FileStream*, the *Decoder* class in .NET Framework provides facilities to convert a sequence of encoded bytes into a set of characters. The method GetChars() is used to decode and convert the bytes to characters. Add the following code into method *BBook_load( )* after the code above, run the program, the names will be displayed twice.

```
//create an object of Decoder
Decoder d = Encoding.UTF8.GetDecoder();
int charsDecodedCount = d.GetCharCount(bytes, 0, bytes.Length);
```

```
char[] chars = new char[charsDecodedCount];

//using decoder class method GetChars to decode and convert
//the bytes in byte array ‘bytes’ which has been declared above
int numChar = d.GetChars(bytes, 0, bytes.Length, chars, 0);

//split the string into an array of strings
//each string in the array is a sentence from the input file
contents = new String(chars);
sentences = contents.Split(new Char[] { '\n' });

//get each sentence and add it onto the birthday table
for (int j = 0; j < sentences.Length; j++)
{
    DataRow new_row = m_BBookTable.NewRow();
    new_row["Name"] = sentences[j];
    m_BBookTable.Rows.Add(new_row);
}
```

Run the program again, this time the list of names will be displayed twice in the birthday book form.

- Use *StreamReader*

We also can use a *StreamReader* object to read the sentences in the input file. In this case, we don’t need decoding since *StreamReader* reads the input file in characters. Add the following code to the method *BBook_load( )*, run the program, the names will be displayed three times.

```
 //create a streamReader object
  StreamReader streamIn;
  //read strings from an input file
  streamIn = new StreamReader("inpututf8.txt");


  string line;
//read each line in the input file and add into the birthday table
  while ((line = streamIn.ReadLine()) != null)
  {
      DataRow dr = m_BBookTable.NewRow();
      dr["Name"] = line;
      m_BBookTable.Rows.Add(dr);
  }
  streamIn.Close();
```

Here we just want to show different ways of decoding. Obviously, you can directly use `StreamReader` to read text in different languages.