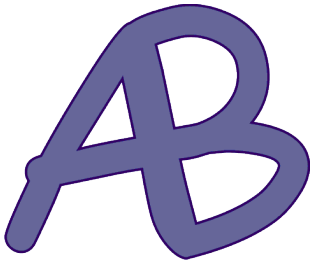


Wykład 10
Kolekcje, pliki tekstowe, c.d.

Przykład: Notatnik

Przykład:

Wykonamy prosty notatnik obsługujący pliki tekstowe.
Notatnik posiadał będzie następujące funkcje:

- odczyt z pliku,
- zapis do pliku,
- zamian koloru fontu i tła
- zmiana rozmiaru fontu,
- obsługa schowka

Notatnik

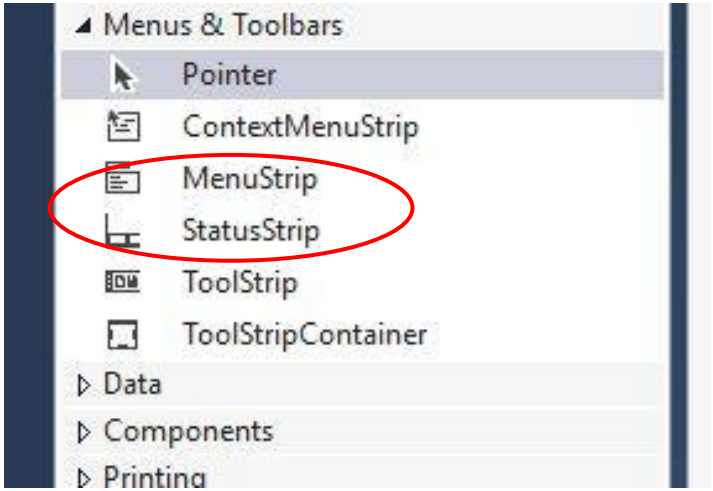
Notatnik zbudujemy w oparciu o zwykły komponent textBox.

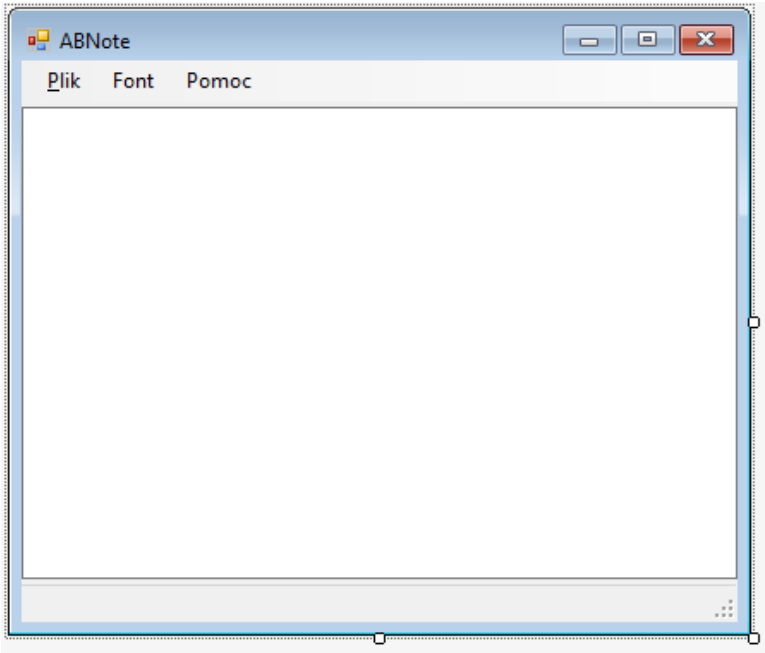
Po ustawieniu pole .Multilane na true, może on wyświetlać teksty wielolinijkowe.

texBox, oprócz pole .Text w którym zapisać można pojedynczą zmienną string, posiada strukturę .Lines, która jest tablicą string-ów (jedno pole jedna linia wyświetlona w textBox-ie.

Notatnik - leyaut

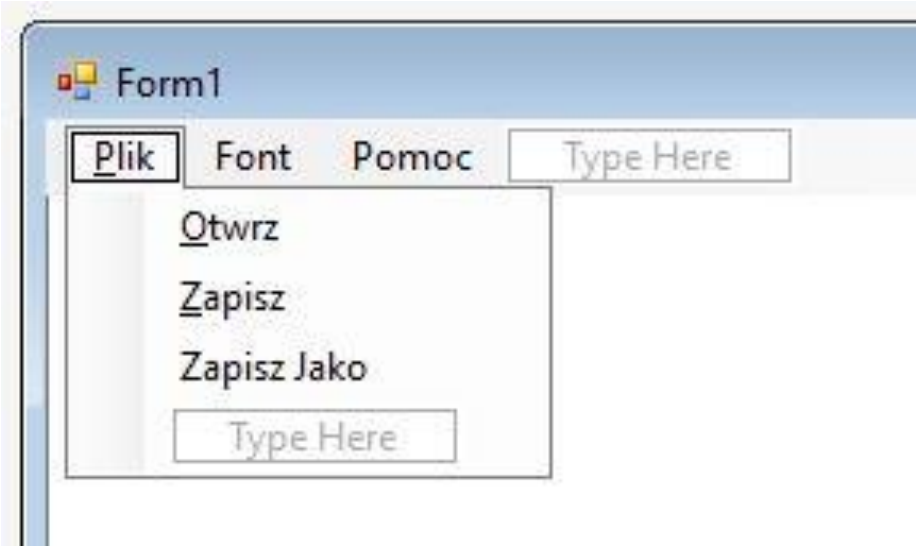
Oprócz texBox wstawiamy komponenty menuStrip i StatusStrip





Notatnik - leyaut

Kolejnym krokiem jest wypełnienie pozycji menu

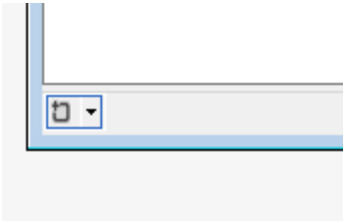


Poprzedzając nazwę pozycji menu znakiem & (np. &Plik) wybieramy aktywny klawisz.

Można też dodać skróty klawiaturowe w oknie właściwości pozycja ShortcutKeys.

Notatnik - leyaut

Przygotowujemy pasek statusu



Kontrolka statusStrip poosiada strukturę .Items, do której dodać możemy kilka typów obiektów. W naszym przypadku dodajemy Label, który wyświetli nazwę pliku.

Tekst wstawimy w sposób następujący:

```
statusStrip1.Items[0].Text = "tekst na pasku statusu";
```

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.IO;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
```

[0] oznacza pierwszy element na liści Items

Notatnik - leyaut

Teraz dodamy i oprogramujemy obiekt klasy openFileMenu – aby wczytać nazwę pliku.

Obsługę okna openDialgFile umieszczamy w zdarzeniu klikinięcia na pozycję menu (wystarczy kliknąć dwukrotnie na pozycję „Plik->Otwórz”

```
private void otwrzToolStripMenuItem_Click(object sender, EventArgs e)
{
    if (openFileDialog1.ShowDialog()==DialogResult.OK)
    {
        statusStrip1.Items[0].Text = openFileDialog1.FileName;
        textBox1.Lines = CzytajPlik(openFileDialog1.FileName);
    }
}
```

Metodę CzytajPlik() musimy jeszcze napisać. Powinna ona zwrócić tablicę łańcuchów, którą wstawiamy do pola Lines w texBox1

Notatnik - leyaut

```
public static string[] CzytajPlik(string nazwaPliku)
{
    List<string> tekst = new List<string>();
    try
    {
        using (StreamReader sr = new StreamReader(nazwaPliku))
        {
            string wiersz;
            while (!sr.EndOfStream)
            {
                wiersz = sr.ReadLine();
                tekst.Add(wiersz);
            }
        }
        return tekst.ToArray();
    }
    catch (Exception e)
    {
        MessageBox.Show("Błąd odczytu pliku " + nazwaPliku + " (" + e.Message + ")");
        return null;
    }
}
```

Odczyt linii z pliku i zapis dodawanie ich do listy.
Patrz poprzedni wykład

Zapis do pliku

Dodamy i oprogramujemy obiekt klasy SaveFileMenu – aby wybrać nazwę pliku.

Obsługę okna saveDialgFile umieszczamy w zdarzeniu klikinięcia na pozycję menu (wystarczy kliknąć dwukrotnie na pozycję „Plik->Zapisz jako”

```
private void zapiszJakoToolStripMenuItem_Click(object sender, EventArgs e)
{
    if (saveFileDialog1.ShowDialog() == DialogResult.OK)
    {
        statusStrip1.Items[0].Text = saveFileDialog1.FileName;
        ZapiszDoPliku(saveFileDialog1.FileName, textBox1.Lines);

    }
}
```

Metodę ZapiszDopliku( )() musimy jeszcze napisać. Nie zawraca ona żadnej wartości , ale przyjmuje dwa parametry – nazwę pliku i tablicę łańcuchów do zapisania w pliku. Tablica bierzemy ze struktury lines pola texBox.

Zapis do pliku

```
public static void ZapiszDoPliku(string nazwaPliku, string[] tekst)
{
    using (StreamWriter sw = new StreamWriter(nazwaPliku))
    {
        foreach (string wiersz in tekst)
            sw.WriteLine(wiersz);
    }
}
```

Można też tak:

```
public static void ZapiszDoPliku(string nazwaPliku, string[] tekst)
{
    using (StreamWriter sw = new StreamWriter(nazwaPliku))
    {

        for (int i = 0; i < tekst.Length; i++)
            sw.WriteLine(tekst[i]);
    }
}
```

Obsługa schowka

Obsługa schowka systemowego dla komponentu TexBox jest prosta.

Posiada on gotowe metody wymiany danych ze schowkiem.

```
textBox1.Copy( );
textBox1.Cut( ) ;
textBox1.Paste( ) ;
textBox1.SelectAll( );
textBox1.Undo( );
```

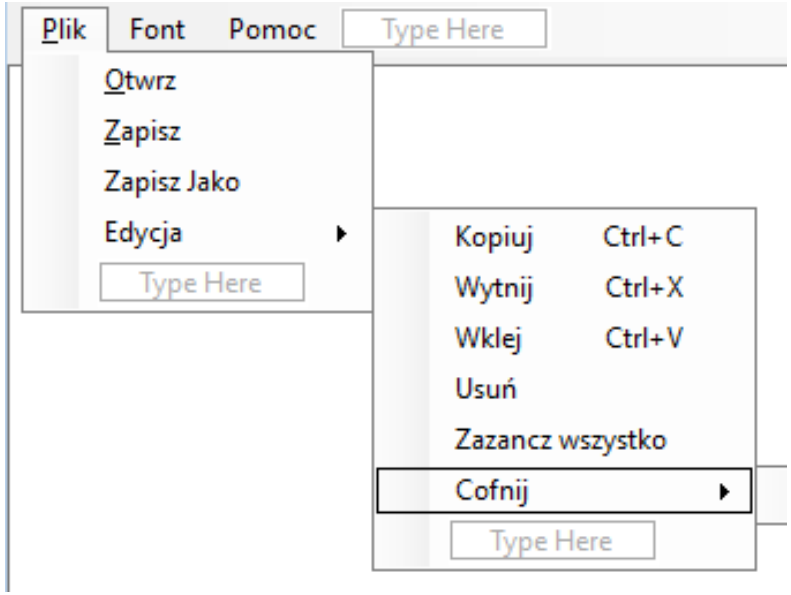
Mamy do dyspozycji także pole „SelectedText” zawierające tekst zaznaczony. Możemy na przykład wykasować zaznaczenie:

```
textBox1.SelectedText = "";
```

Obsługa schowka

Do menu dodajemy submenu Edycja, np. tak:

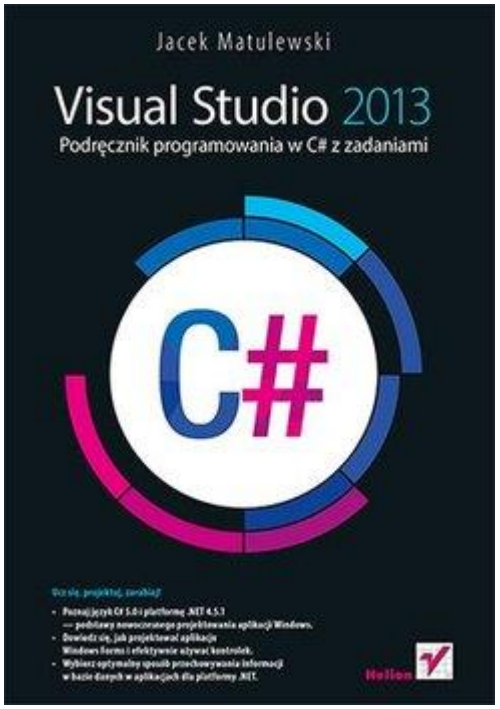
Następnie oprogramowujemy zdarzenia kliknięci na menu (zdarzenie Click).

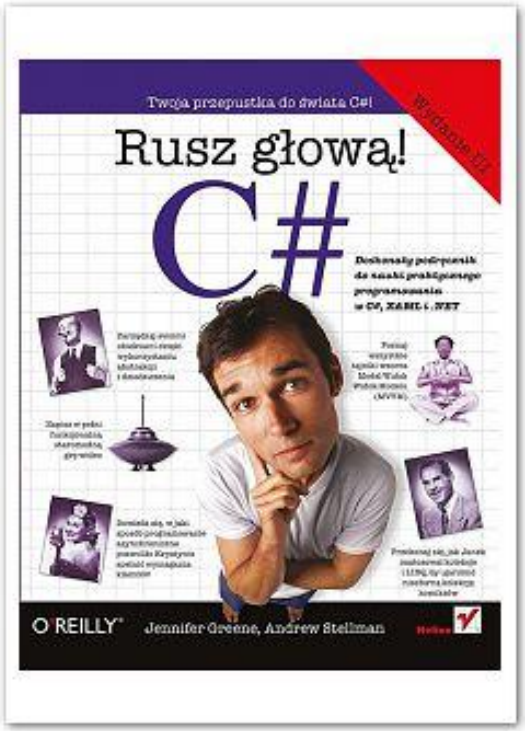


Obsługa schowka

```
private void kopiujToolStripMenuItem_Click(object sender, EventArgs e)
{
    textBox1.Copy();
}
private void wytnijToolStripMenuItem_Click(object sender, EventArgs e)
{
    textBox1.Cut();
}
private void wklejToolStripMenuItem_Click(object sender, EventArgs e)
{
    textBox1.Paste();
}
private void usuńToolStripMenuItem_Click(object sender, EventArgs e)
{
    textBox1.SelectedText = "";
}
private void zazanczWszystkoToolStripMenuItem_Click(object sender, EventArgs e)
{
    textBox1.SelectAll();
}
private void cofnijToolStripMenuItem_Click(object sender, EventArgs e)
{
    textBox1.Undo();
}
```

Literatura:





Użyte w tej prezentacji tabelki pochodzą z książki:Visual Studio 2013. Podręcznik programowania w C# z zadaniami
Autor: Matulewski Jacek, Helion