Tlačítka

Minule jsme si ukázali, jak založit nový projekt WPF aplikace, jak do aplikace vložit popisek, jak popisek nastavit a jak aplikaci spustit. Nyní si ukážeme, jak v aplikaci vytvářet tlačítka a jak je nastavovat.

Ve chvíli, když vytváříme desktopovou aplikaci, musíme uživatelům dát možnost aplikaci ovládat. Pomocí popisků sice můžeme uživatelům data prezentovat, ale pro zpracování jejich vstupu musíme využít prvků jiných. Jedním z prvků, které můžeme k tomuto účelu použít, je tlačítko. Při vytváření tlačítka máme opět stejné možnosti jako při vytváření popisku – použít toolbox, napsat kód v XAMLu anebo vytvořit element pomocí C#. V následující části bude ukázáno, jak tlačítko vytvářet pomocí jazyka XAML.

Při vytváření tlačítka pomoc jazyka XAML je nejprve nutné napsat znak < (menší než).

Následně je potřeba zapsat název vytvářeného elementu – v tomto případě Button.

Déle je nutné uzavřít tag a to buď pomocí znaku >, po jehož zápisu by vývojové prostředí mělo vygenerovat tag uzavírající anebo je možné otevírající tag uzavřít pomocí znaku /, po jehož zápisu by vývojové prostředí mělo vygenerovat znak >. Stejně jako minule popisek, i tlačítko umístíme do automaticky vytvořeného elementu Grid. Definice prvku poté vypadá následovně

1. Způsob:

   ```
   <Button></Button>
   ```

2. Způsob:

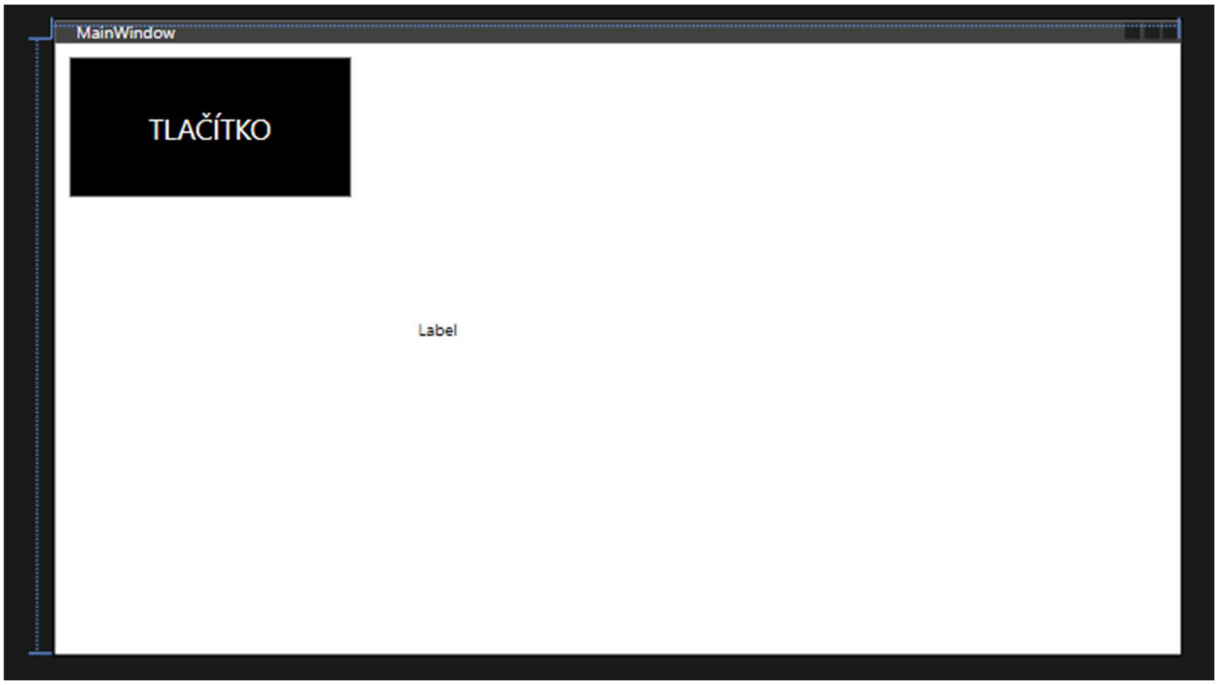
   ```
   <Button/>
   ```

Po zápisu elementu Button do XAMLu zjistíme, že vytvořené tlačítko zabralo celý rozsah okna. Pokud bude potřeba, aby tlačítko zabíralo místa méně je potřeba tlačítko dále nastavit pomocí jeho atributů.

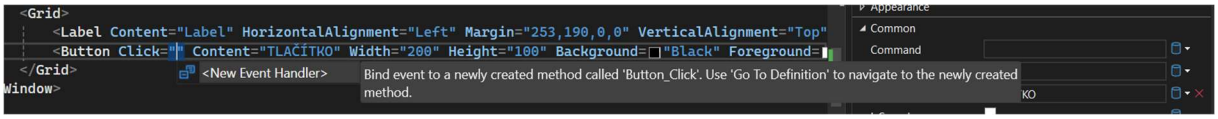
Spousta atributů tlačítka se bude shodovat s atributy popisku, takže pokud budeme potřebovat vytvořit tlačítko s textem TLAČÍTKO, které bude mít velikost 200x100, barvu pozadí černou, barvu textu uvnitř tlačítka bílou, velikost fontu 20px a umístění na pozici 10px zleva a 10px ze shora, pak bude zápis prvku v XAMLu vypadat následovně

```
<Button Content="TLAČÍTKO" Width="200" Height="100" Background="Black"
Foreground="White" FontSize="20" VerticalAlignment="Top"
HorizontalAlignment="Left" Margin="10, 10, 0, 0"/>
```

Tlačítko v aplikaci bude vypadat následovně



Nyní, když aplikaci spustíme, můžeme tlačítko používat tím, že na něj klikneme. Problém je akorát v tom, že tlačítku zatím nebyla přiřazena žádná metoda, které by po jeho stisknutí byla vykonána. Pokud budeme potřebovat tlačítku akci vykonanou při jeho stisknutí přiřadit (což budeme potřebovat vždy – jinak by nám bylo tlačítko asi na dvě věci) budeme muset již psát kód nejenom v XAMLu ale také v jazyce C#. Dejme tomu, že by naším cílem bylo vytvořit jednoduchou aplikaci ve které budeme zjišťovat kolikrát bylo tlačítko stisknuto a tuto hodnotu zobrazovat v popisku vytvořeném minule. První, co bude třeba u tlačítka nastavit, je atribut Click. Při jeho zápisu do XAMLu nám vývojové prostředí nabídne všechny použitelné metody ze C# souboru propojeného s XAMLem a také nám dá možnost vytvořit metodu novou. Jelikož jsme ještě zatím žádnou metodu nevytvářely necháme ji vytvořit automaticky.



Zápis v XAMLu po automatickém vygenerování bude vypadat následovně

```
<Button Click="Button_Click" Content="TLAČÍTKO" Width="200" Height="100"
Background="Black" Foreground="White" FontSize="20" VerticalAlignment="Top"
HorizontalAlignment="Left" Margin="10, 10, 0, 0"/>
```

Nyní se přesuneme do souboru MainWindow.xaml.cs – do souboru je možné přejít pomocí horní lišty, průzkumníku řešení na pravé straně, dvojitým kliknutím na okno aplikace v designu, pravým kliknutím na oknu v designu a následným výběrem možnosti view code (zobrazit kód).

Soubor, který se nám otevře by měl zatím vypadat následovně

```
using System.Text;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;

namespace MyDesktopApp
{
    /// <summary>
    /// Interaction logic for MainWindow.xaml
    /// </summary>
    public partial class MainWindow : Window
    {
        public MainWindow()
        {
            InitializeComponent();
        }

        private void Button_Click(object sender, RoutedEventArgs e)
        {

        }
    }
}
```

Jako první, co v souboru uvidíme jsou příkazy using následované názvem importovaného namespacu. Jedná se o nástroje, které jsou do aplikace importovány a které můžeme při programování používat. Většina z těchto příkazů bude zašedlá, což značí že z daného namespacu nic nepoužíváme a že jeho import není nutně potřeba.

Dále se v souboru nachází klíčové slovo namespace následované názvem naší aplikace. Jedná se o jmenný prostor naší aplikace, ve kterém budou automaticky vytvářeny další námi definované programové části.

Uvnitř jmenného prostoru se nachází třída MainWindow, která dědí ze třídy Window. Jedná se o třídu definující vlastnosti hlavního okna naší aplikace. Označení partial značí, že definice třídy je rozdělena na více částí – automaticky vygenerovaný kód, který zde nevidíme a část kterou si nadefinujeme sami (aktuální soubor).

Uvnitř třídy MainWindow je vytvořen konstruktor této třídy. Uvnitř konstruktoru se nachází volání metody InitializeComponent, která provádí inicializaci všech uživatelských prvků ze XAMLu.

Nakonec se ve třídě nachází metoda Button_Click. Jedná se o metodu, které byla automaticky vytvořena při nastavování atributu Click u tlačítka ve XAMLu. Tato metoda bude zavolána pokaždé když na ve spuštěné aplikaci klikneme na tlačítko.

Naším úkolem bylo vytvořit aplikaci, která bude počítat kolikrát bylo tlačítko stisknuto a tuto hodnotu následně vypisovat do popisku. K tomu, aby aplikaci počítala počet stisknutí, musíme tuto hodnotu někam ukládat.

Z tohoto důvodu vytvoříme v aplikaci atribut, do kterého bude hodnota počtu uložena. Kód upravíme následujícím způsobem

```
using System.Text;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;

namespace MyDesktopApp
{
    public partial class MainWindow : Window
    {
        public MainWindow()
        {
            InitializeComponent();
        }

        int _count = 0;
        private void Button_Click(object sender, RoutedEventArgs e)
        {
            _count++;
        }
    }
}
```

Nyní, když stiskneme tlačítko, dojde ke zvětšení hodnoty v atributu _count, nicméně v aplikaci žádnou změnu popisku neuvidíme. Důvod je ten, že hodnota je sice zvětšována, ale nikde není do popisku přiřazena. K tomu, abychom mohli hodnotu do popisku vložit bude potřeba nejprve popisek načíst a k tomu mu budeme muset přiřadit název.

V XAMLu definujeme vytvořenému elementu Label atribut Name (můžete použít i x:Name) a nastavíme na unikátní textovou hodnotu. Definice popisku v XAMLu po úpravě bude vypadat například takto

```
<Label Name="ClickCountLabel" Content="Label" HorizontalAlignment="Left"
Margin="253,190,0,0" VerticalAlignment="Top"/>
```

Nyní můžeme popisek pomocí jeho názvu v C# kódu načíst a nastavit. V XAMLu bychom nastavovali popisku atribut Content a stejný atribut nastavíme i v C#, akorát ho budeme muset načíst jiným způsobem. Pro načtení atributu z objektu je v C# možné používat znak . (tečka). Kód po úpravě bude vypadat takto

```
using System.Text;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;

namespace MyDesktopApp
{
    public partial class MainWindow : Window
    {
        public MainWindow()
        {
            InitializeComponent();
            ClickCountLabel.Content = _count;
        }

        int _count = 0;
        private void Button_Click(object sender, RoutedEventArgs e)
        {
            _count++;
            ClickCountLabel.Content = _count;
        }
    }
}
```

V konstruktoru třídy (zápis základní hodnoty při spuštění aplikace) a v metodě Button_Click (zápis hodnoty při každém kliknutí na tlačítko) přibyl řádek, který nastaví obsah popisku na hodnotu atributu _count. Nyní když aplikaci spustíme a budeme klikat na tlačítko bude se hodnota v popisku postupně zvedat.

Aplikace funguje, ale co kdybychom potřebovali hodnotu počtu kliknutí zobrazit i v samotném tlačítku. To, co bychom mohli udělat, by bylo nastavení názvu tlačítka a následně jeho načtení v kódu podobně jako u popisku. Další způsob je využití parametrů metody, která je při kliknutí tlačítka zavolána. Metoda má dva parametry (`object sender, RoutedEventArgs e`), které k tomuto účelu použít. Odkaz na tlačítko, na které bylo kliknuto bude uložen jednak v parametru sender a jednak v atributu Source parametru e. Než ale takto načtenému tlačítku nastavíme atribut Content na požadovanou hodnotu, je nutné provést převod mezi datovými typy. Parametr sender i atribut Source parametru e mají datový typ object a my budeme potřebovat pracovat s instancí datového typu Button. Pro načtení tlačítka a převod mezi datovými typy je možné použít následující kód

```
Button button = (Button)sender;
```

anebo

```
Button button = (Button)e.Source;
```

Nyní, pokud bude metoda přiřazena tlačítku a dojde k jeho stisknutí, bude v proměnné button uložen odkaz na tlačítko, které můžeme nastavit. Pokud by ale metoda byla přiřazena něčemu jinému, než tlačítku bude v proměnné uložena hodnota null a kvůli tomu by měla ještě proběhnout kontrola, že v proměnné je skutečně odkaz na tlačítko uložen a až poté tlačítka nastavovat. Metoda po úpravě bude vypadat následovně

```
private void Button_Click(object sender, RoutedEventArgs e)
{
    _count++;
    ClickCountLabel.Content = _count;
    Button button = (Button)e.Source;
    if (button != null)
    {
        button.Content = _count;
    }
}
```

Nyní se bude počet stisknutí zobrazovat nejen v popisku, ale i v tlačítku.

Příště se podíváme, jak můžeme prvky uvnitř naší aplikace nastavit takovým způsobem, aby se automaticky zarovnávali a tím nám usnadnily práci při jejich pozicování.