WSEI

Microsoft Imagine Academy
Program Member

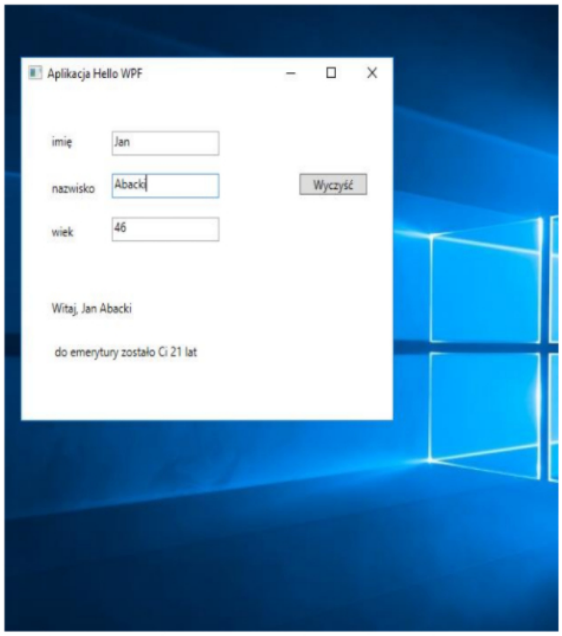
Lab. 1.5. Aplikacja desktopowa (WPF)

Tworzenie aplikacji – formularza w trybie Design; toolbox; elementy formularza: label, TextBox, Button; atrybuty (Name), Text, Content; obsługa zdarzenia `LostFocus` dla pola tekstowego, `Click` dla przycisku, XAML

C#

Cel

- Celem tego ćwiczenia będzie przekształcenie opracowanej w Lab. 1.1 aplikacji konsolowej na aplikację desktopową, jednookienkową, w technologii WPF.
- *Windows Presentation Foundation* (WPF) to GUI (*Graphical User Interface*) – graficzne środowisko użytkownika dla nowszych wersji Windows. W Lab. 1.4. zrealizowaliśmy aplikację z wykorzystaniem biblioteki Windows Forms – trochę starszego, ale nadal kompatybilnego z Windows środowiska graficznego. WPF jest nowocześniejszym i bardziej elastycznym dla programisty środowiskiem.
- Proces tworzenia aplikacji WPF w zasadzie nie odbiega znacząco od procesu tworzenia aplikacji typu WinForm. Również korzystamy z Toolbox'a oraz Properties. Różnicą widoczną na pierwszy rzut oka jest zapis wyglądu formularza w języku znacznikowym XAML (wariant XML). Zastosowanie tej technologii pozwala na wyraźne odseparowanie wyglądu aplikacji od jej logiki oraz usunięcia problemów z uszkodzeniami kodu przy pracy wizualnej.



Tworzenie nowego projektu

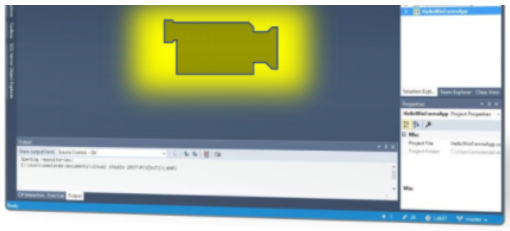
screencast

- Do istniejącej solucji dodaj nowy projekt:
  - W oknie Solution Explorer prawokliknij myszką na *Solution 'Lab01'* i z menu kontekstowego wybierz Add→New Project ...

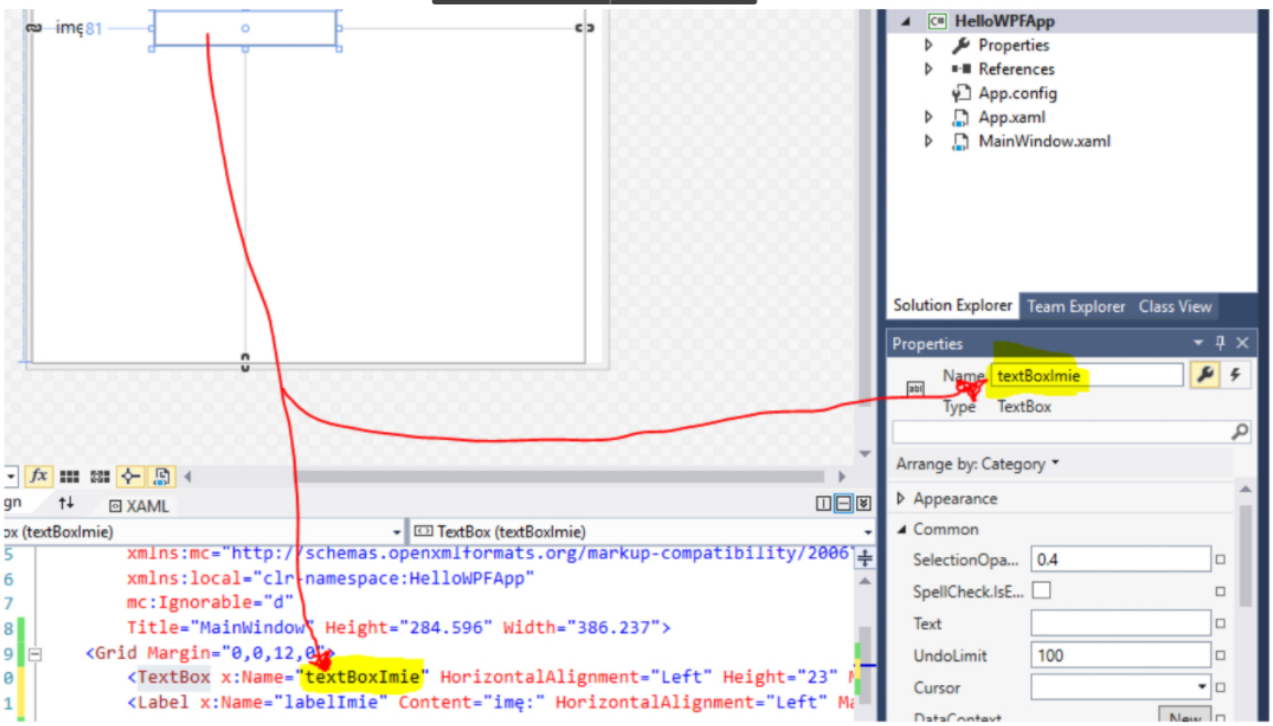
- Utwórz projekt według szablonu Visual C# → WPF App (.NET Framework) nadając mu nazwę HelloWPFApp
- Uczyń ten projekt aktywnym (StartUp)

› Kreator wygeneruje stosowny kod (MainWindow.xlam, MainWindow.xlam.cs, App.xlam, App.xlam.cs) oraz udostępni Ci interaktywne narzędzia do projektowania formularza (wyglądu okienka) – Designer. Modyfikację wyglądu formularza możesz przeprowadzać zarówno wizualnie jak i w kodzie XAML.

› Podobnie jak w Lab. 1.4 utworzymy aplikację jednookienkową, z drobną zmianą – komunikaty zmieniać się będą z chwilą zmiany zawartości w polach tekstowych.



podgląd



Edycja okna formularza

› Pracując w trybie Design w pliku MainWindow.xaml zaprojektuj formularz podobny do tego z poprzedniego Lab 1.4, ale z przyciskiem Wyczyść. Pamiętaj o zmianie programowych nazw kontrolek.

› Kontrolka typu Label – służy do umieszczania tekstu na formularzu.
- Atrybut Content – treść wypisana w formularzu (inaczej niż w WinForm)
- Atrybut x:Name – nazwa obiektu kontrolki – z punktu widzenia C# (nazwa zmiennej).

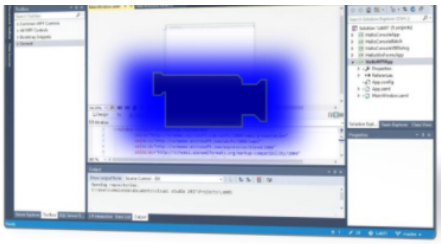
› Kontrolka typu TextBox – pole tekstowe – służy do wprowadzania krótkich tekstów.

screencast

- Atrybut Text – treść pojawiająca się w polu tekstowym
- Atrybut x:Name – nazwa obiektu kontrolki – z punktu widzenia C# (nazwa zmiennej).

› Zmień tytuł okna z „MainWindow” na „Aplikacja Hello WPF”.

› Przetestuj swój formularz – uruchamiając program (zdecydowanie szybsze uruchomienie bez debugera Ctrl-F5)

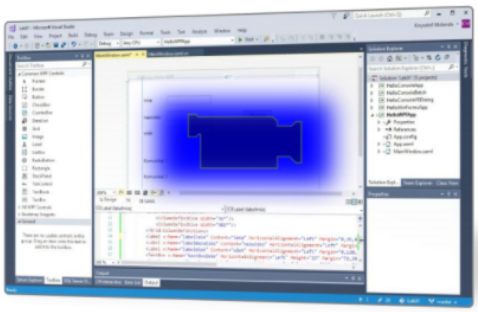


Programowanie funkcjonalności formularza (1)

› Do aplikacji dodamy funkcjonalność: *Komunikat 1 oraz Komunikat 2 zmieniają się po wprowadzeniu tekstów do odpowiednich pól tekstowych.*

› Tym razem sygnałem zmiany będzie „utrata focusu”, czyli opuszczenie pola tekstowego.

- Zaznacz pole tekstowe `textBoxImie`, a następnie okno Properties przełącz na procedury obsługi zdarzeń (ikona błyskawicy)
- Wyszukaj zdarzenie o nazwie LostFocus i dwukrotnie kliknij na pustym polu tekstowym skojarzonym z tym zdarzeniem. Visual Studio przeniesie Cię do wygenerowanego automatycznie szkieletu kodu procedury obsługi zdarzenia LostFocus.
- Wpisz kod opisujący logikę obsługi zdarzenia.
- Podobnie postąpisz dla pól tekstowych textBoxNazwisko oraz textBoxWiek.



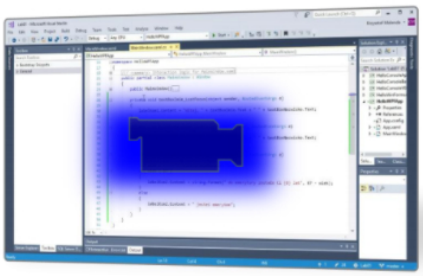
screencast

kod


```
namespace HelloWPFApp
{
    /// <summary> Interaction logic for MainWindow.xaml
    public partial class MainWindow : Window
    {
        public MainWindow()...

        private void textBoxImie_LostFocus(object sender, RoutedEventArgs e)
        {
            labelKom1.Content = "Witaj, " + textBoxImie.Text + " " + textBoxNazwisko.Text;
        }

        private void textBoxNazwisko_LostFocus(object sender, RoutedEventArgs e)
        {
            labelKom1.Content = "Witaj, " + textBoxImie.Text + " " + textBoxNazwisko.Text;
        }

        private void textBoxWiek_LostFocus(object sender, RoutedEventArgs e)
        {
            int wiek = Convert.ToInt32(textBoxWiek.Text);
            if( wiek < 67 )
            {
                labelKom2.Content = string.Format(" do emerytury zostało Ci {0} lat", 67 - wiek);
            }
            else
            {
                labelKom2.Content = " jesteś emerytem";
            }
        }
    }
}
```

Programowanie funkcjonalności formularza (2)

- Pozostaje oprogramować działanie przycisku `buttonWyczysc`. Wygeneruj szkielet kodu procedury obsługującej zdarzenie Click. Wpisz kod „zerujący” pola tekstowe i etykiety.
- Nieestetyczny wygląd – po uruchomieniu aplikacji wyświetlane są nic nie mówiące opisy Komunikat 1 oraz Komunikat 2. Z takimi domyślnymi wartościami zainicjowane zostały zmienne `labelKom1` oraz `labelKom2`. W prosty sposób można to poprawić – należy usunąć w odpowiednich atrybutach Text te domyślne wartości.
- W przypadku podania nienumerycznej (niecałkowitej) wartości w polu wiek zgłaszany jest wyjątek i aplikacja przerywa działanie.
- Aplikacja (podobnie jak te z poprzednich lab.) jest jednoplikowa. Może być uruchomiona w systemie Windows z zainstalowanym środowiskiem .NET Framework 4.5.2.



screencast

```
private void buttonWyczysc_Click(object sender, RoutedEventArgs e)
{
    labelKom1.Content = "";
    labelKom2.Content = "";
    textBoxImie.Text = "";
    textBoxNazwisko.Text = "";
    textBoxWiek.Text = "";
}
```

kod

Wyzwanie (dla bardziej zaawansowanych): spróbuj samodzielnie wykonać aplikację WPF opisaną w:
- Tutorial MSDN (My First WPF Desktop Application)