

## Lab. 1.4. Aplikacja desktopowa (WinForm)

Tworzenie aplikacji – formularza w trybie Designer; toolbox; elementy formularza: label, TextBox, Button; atrybuty (Name) oraz Text; obsługa zdarzenia Click dla przycisku, podział kodu na pliki – partial class.

C#

15.05.2018

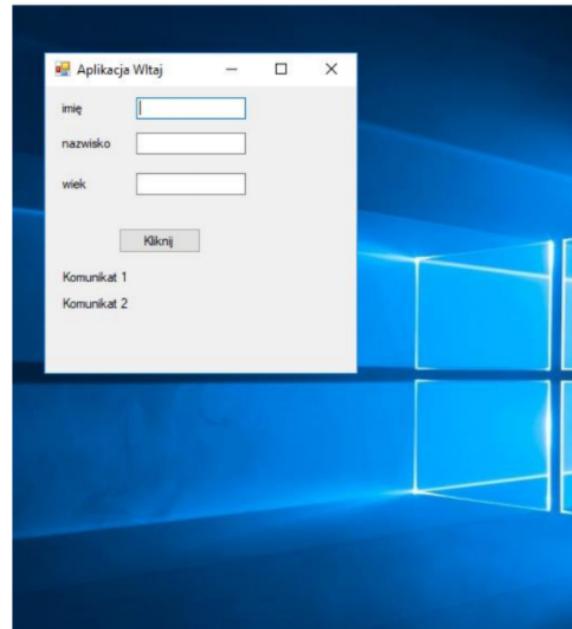
KRZYSZTOF MOLENDY - MICROSOFT IMAGINE ACADEMY@WSEI

1

C#

### Cel

- › Celem tego ćwiczenia będzie przekształcenie opracowanej w Lab. 1.1 aplikacji konsolowej na aplikację desktopową, jednookienkową.
- › Wykorzystana zostanie technologia i biblioteka Windows Forms.
- › W tym ćwiczeniu zademonstrowana zostanie technika wizualnego tworzenia interfejsu użytkownika.



Lab. 1. „Hello World!”

Aplikacja desktopowa (WinForm)

C#

### Tworzenie nowego projektu

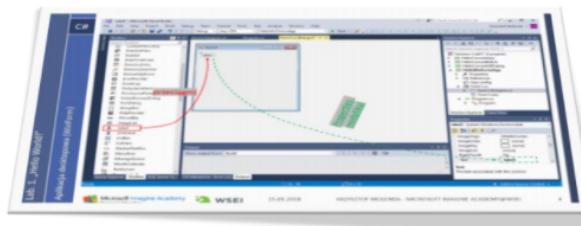
- › Do istniejącej solucji dodaj nowy projekt:
  - W oknie Solution Explorer prawokliknij myszką na *Solution 'Lab01'* i z menu kontekstowego wybierz Add→New Project ...

screencast



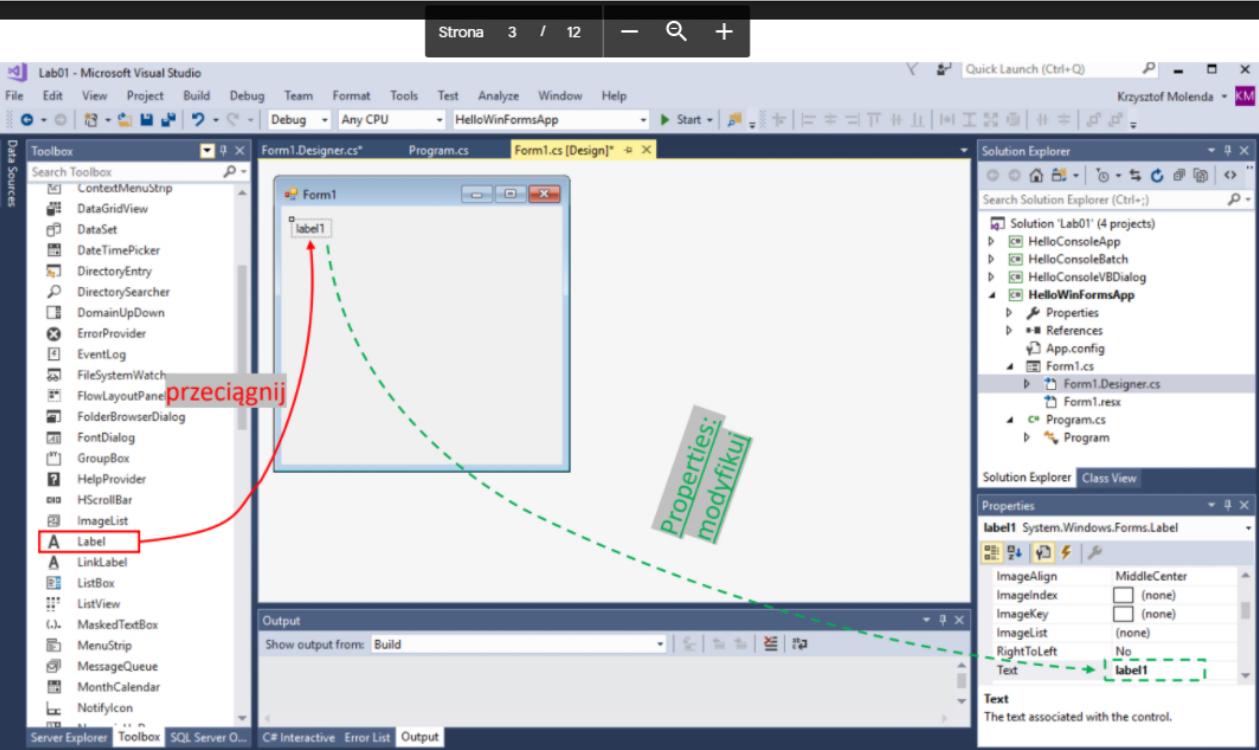
- Utwórz projekt według szablonu **Visual C# → Windows Forms App (.NET Framework)** nadając mu nazwę **HelloWinFormsApp**
- Uczynь ten projekt aktywnym (**StartUp**)

- › Kreator wygeneruje stosowny kod (**Form1.cs**, **Program.cs**) oraz udostępnii Ci interaktywne narzędzia do projektowania formularza (wyglądu okienka) – **Form1.cs [Design]**.
- › Projektując wizualnie interfejs użytkownika (okno aplikacji) posługiwać się będziesz **Toolbox-em** kontrolek formularza oraz oknem **Properties**.
- › Projektowanie interfejsu – ogólnie rzecz biorąc – polega na przeciąganiu kontrolek w obszar formularza, umiejscawianiu ich i określaniu ich parametrów w oknie **Properties**.



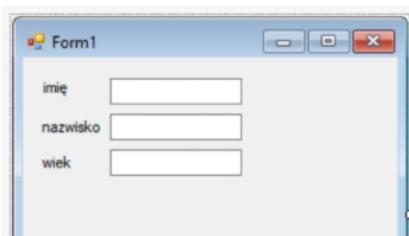
podgląd

przeciagnij

Properties:  
modyfikuj

## Edycja okna formularza (1)

- › Kontrolka typu **Label** – służy do umieszczania tekstu na formularzu.
  - Atrybut **Text** – treść wypisana w formularzu
  - Atrybut **(Name)** – nazwa obiektu kontrolki – z punktu widzenia C# (nazwa zmiennej).
  - Polecenie:
    - > Umieść na formularzu kontrolkę typu **Label**,
    - > zmień jej nazwę **(Name)** z **label1** na **labelImie**
    - > zmień jej treść (atrybut **Text**) na **imię:**
- › Kontrolka typu **TextBox** – pole tekstowe – służy do wprowadzania tekstów
  - Polecenie:



screencast

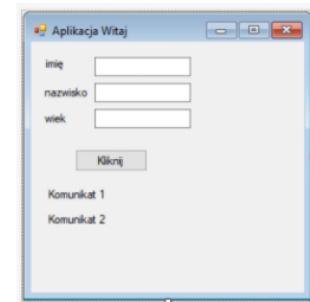


- › Umieść, obok kontrolki typu **Label** imię – kontrolkę typu **TextBox**. W tym polu użytkownik wprowadzał będzie imię
- › Zmień jej nazwę (**Name**) z **textBox1** na **textBoxImie**
- › Przetestuj swój formularz – uruchamiając program (**F5** lub **Ctrl-F5**)
- › Powtórz czynności dla: **nazwisko** oraz **wiek**. Nie zapomnij o nadawaniu nazw (**Name**) odpowiednim kontrolkom.



## Edycja okna formularza (2)

- › Aby zmienić tytuł okna, kliknij na pasku tytułu okienka, następnie w **Properties** wyszukaj atrybut **Text** i zmień jego wartość na **Aplikacja Witaj**.
- › Pod polami tekstowymi umieść przycisk – kontrolkę typu **Button**. Zmień jej nazwę (**Name**) na **buttonOK** oraz tekst na **Kliknij**. Przetestuj klikalność przycisku (uruchom, **F5**). Oczywiście nic nie będzie się dziać – nie przypisano żadnego kodu do przycisku.
- › Pod przyciskiem umieść dwie etykiety typu **Label** o nazwach **labelKomunikat1** oraz **labelKomunikat2** i treści odpowiednio **Komunikat 1** oraz **Komunikat 2**.



screencast



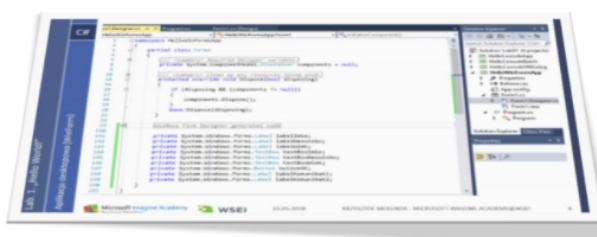
## Edycja okna formularza (2)

- › W trakcie wykonywania przez Ciebie czynności manualnych Visual Studio wygenerowało odpowiedni kod. Kod ten jest wstępnie ukryty – zwinięty.
- › Nieumiejęte zmiany tego kodu mogą spowodować uszkodzenie projektu (części wizualnego projektowania).
  - Zalecenie dla początkujących: nie dotykaj kodu, zmiany rób za pomocą **Properties** w trybie **Design**. Często rób kopię zapasową solucji.
  - Zalecenie dla zaawansowanych: podłącz projekt do systemu kontroli wersji (np. GitHub).
- › Program główny i funkcja **Main()** składa się tylko z 3 linijek kodu – ostatnia linijka tworzy i uruchamia okno aplikacji.
- › Kod obsługi formularza podzielony jest na części – w różnych plikach (**partial class**). W pliku **Form1.Designer.cs** przechowywany jest kod wygenerowany automatycznie. W pliku **Form1.cs** tworzysz własny kod.
- › Zwróć uwagę, że w kodzie pojawił się zapis:
 

```
using System.Windows.Forms;
```



podgląd kodu



Strona 7 / 12 | - Q +

m1.Designer.cs X Program.cs Form1.cs [Design]

HelloWinFormsApp HelloWinFormsApp.Form1 InitializeComponent()

```

1  namespace HelloWinFormsApp
2  {
3      partial class Form1
4      {
5          /// <summary> Required designer variable.
6          private System.ComponentModel.IContainer components = null;
7
8          /// <summary> Clean up any resources being used.
9          protected override void Dispose(bool disposing)
10         {
11             if (disposing && (components != null))
12             {
13                 components.Dispose();
14             }
15             base.Dispose(disposing);
16         }
17
18         Windows Form Designer generated code
19
20         private System.Windows.Forms.Label labelImie;
21         private System.Windows.Forms.Label labelNazwisko;
22         private System.Windows.Forms.Label labelWiek;
23         private System.Windows.Forms.TextBox textBoxImie;
24         private System.Windows.Forms.TextBox textBoxNazwisko;
25         private System.Windows.Forms.TextBox textBoxWiek;
26         private System.Windows.Forms.Button buttonOK;
27         private System.Windows.Forms.Label labelKomunikat1;
28         private System.Windows.Forms.Label labelKomunikat2;
29     }
30
31     
```

Solution Explorer

Search Solution Explorer (Ctrl+Shift+F)

- Solution 'Lab01' (4 projects)
  - HelloConsoleApp
  - HelloConsoleBatch
  - HelloConsoleVBDialog
  - HelloWinFormsApp**
    - Properties
    - References
    - App.config
    - Form1.cs
    - Form1.Designer.cs
    - Form1.resx
    - Program.cs
    - Program

Solution Explorer Class View

Properties

Strona 8 / 12 | - Q +

m1.Designer.cs Program.cs Form1.cs [Design]

HelloWinFormsApp HelloWinFormsApp.Program Main()

```

1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Threading.Tasks;
5  using System.Windows.Forms;
6
7  namespace HelloWinFormsApp
8  {
9      static class Program
10     {
11         /// <summary>
12         /// The main entry point for the application.
13         /// </summary>
14         [STAThread]
15         static void Main()
16         {
17             Application.EnableVisualStyles();
18             Application.SetCompatibleTextRenderingDefault(false);
19             Application.Run(new Form1());
20         }
21     }
22
23     
```

Solution Explorer

Search Solution Explorer (Ctrl+Shift+F)

- Solution 'Lab01' (4 projects)
  - HelloConsoleApp
  - HelloConsoleBatch
  - HelloConsoleVBDialog
  - HelloWinFormsApp**
    - Properties
    - References
    - App.config
    - Form1.cs
    - Form1.Designer.cs
    - Form1.resx
    - Program.cs
    - Program

Solution Explorer Class View

Properties

## Programowanie funkcjonalności formularza

- Do aplikacji dodamy funkcjonalność:  
po kliknięciu w przycisk,  
– zamiast Komunikat 1 pojawi się



powitanie: „**witaj, imię nazwisko!**”

- zamiast Komunikat 2 pojawi się tekst dotyczący emerytury.

> Odpowiedni kod realizujący tę funkcjonalność skojarzony będzie z przyciskiem buttonOK (formalnie – obsługa zdarzenia Click dla tego obiektu).

- Dwukrotnie kliknij na kontrolce przycisku na formularzu – wygenerowany zostanie szkielet kodu procedury obsługi zdarzenia.
- Wprowadź logikę obsługi zdarzenia.

> Przetestuj działanie programu.



screencast



kod

Strona 10 / 12 | - Q +

Form1.cs [Design] Form1.cs Form1.Designer.cs Program.cs

HelloWinFormsApp HelloWinFormsApp.Form1 buttonOK\_Click(object sender, EventArgs e)

```

1  using System;
2  using System.Windows.Forms;
3
4  namespace HelloWinFormsApp
5  {
6      public partial class Form1 : Form
7      {
8          public Form1()
9          {
10             InitializeComponent();
11         }
12
13         private void buttonOK_Click(object sender, EventArgs e)
14         {
15             //tu kod obsługi zdarzenia
16             labelKomunikat1.Text = "Witaj, " + textBoxImie.Text + " " +
17                 textBoxNazwisko.Text;
18
19             int wiek = Convert.ToInt32(textBoxWiek.Text);
20             if( wiek < 67 )
21             {
22                 labelKomunikat2.Text = string.Format(" do emerytury zostało Ci {0} lat", 67 -
23                     wiek);
24             }
25             else
26             {
27                 labelKomunikat2.Text = " jesteś emerytem";
28             }
29         }
30     }

```

Solution Explorer

Search Solution Explorer (Ctrl+.)

Solution 'Lab01' (4 projects)

- >HelloConsoleApp
- >HelloConsoleBatch
- >HelloConsoleVBDialog
- HelloWinFormsApp
  - Properties
  - References
  - App.config
  - Form1.cs
    - Form1.Designer.cs
    - Form1.resx
  - Program.cs

Solution Explorer Class View

Properties

## Testowanie aplikacji

> Aplikacja (podobnie jak te z poprzednich lab.) jest jednoplikowa. Może być uruchomiona w systemie Windows z zainstalowanym środowiskiem .NET Framework 4.5.2.

> Błędy:

- W przypadku pytania o wiek, podanie wartości niekonwertowalnej na typ całkowity (`int`) spowoduje zgłoszenie wyjątku `FormatException`.
- Wprowadzenie zbyt dużej wartości całkowitej (np. `123456778987654321`), przekraczającej zakres typu `int` spowoduje zgłoszenie wyjątku `OverflowException`.



- Nieestetyczny wygląd – po uruchomieniu aplikacji wyświetlane są nic nie mówiące opisy Komunikat 1 oraz Komunikat 2. Z takimi domyślnymi wartościami zainicjowane zostały zmienne labelKomunikat1 oraz labelKomunikat2. W prosty sposób można to poprawić – należy usunąć w odpowiednich atrybutach Text te domyślne wartości.

**Wyzwanie (dla bardziej zaawansowanych):** spróbuj samodzielnie wykonać aplikacje WinForm opisane w:

- > [Tutorial1 MSDN](#) (Picture Viewer)
- > [Tutorial2 MSDN](#) (Math Quiz)
- > [Tutorial3 MSDN](#) (Matching Game)