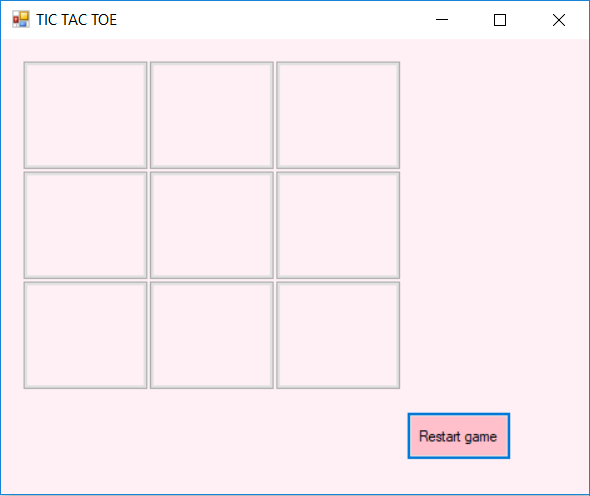
PROJEKT Z PROGRAMOWANIA OBIEKTOWEGO

*Temat: Gra „Kółko i Krzyżyk”*

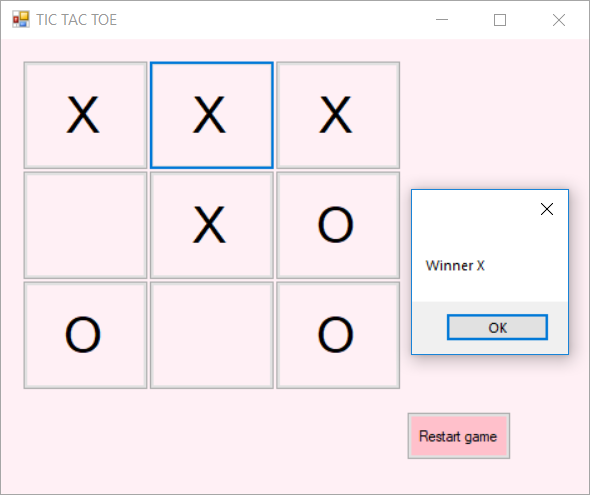
*Autor: Oleksandra Osadcha*

# Opis i założenia projektu

* Głównym zadaniem jest stworzenie programu do gry z komputerem w „Kółko i krzyżyk”.



Rys.1. Początek gry



Rys.2. Koniec gry

* Użytkownik rozpoczyna grę po kliknięciu jednego(dowolnego) przycisku.
* Użytkownik gra krzyżykiem, a komputer gra kółkiem.
* Na zaznaczone przyciski komputer nie reaguje.
* W przypadku, gdy ktoś wygrał:
  + blokują się niewykorzystane przyciski, czyli nie reagują na kliknięcie;
  + po kliknięciu na przycisk „Reset”, gra się resetuje i możemy zagrać na nowo.

# *Opis techniczny projektu*

* Klasy:
  1. ***class GameButton*** – klasa Przycisk.

Klasa, która tworzy przycisk, na którym będzie zaznaczone „X” lub „O”. Klasa ma trzy pola IsClear, IsCross, IsZero, które odpowiadają za czyszczenie przycisków, zaznaczenia krzyżyka lub kółka. Klasa dziedziczy po klasie Button.

Tworzymy następujący konstruktor, żeby zmienić niektóre właściowości(np. rozmiar) naszego przyciska:

public **GameButton**()  //Konstruktor  
{  
    Text=””;  
    Font = new Font(Font.Name, 30, Font.Style);  
    IsClear=true;  
    IsCross=IsZero=false;  
}

Metoda *SetCross* odpowiada za zanaczenie krzyżyka. Odpowiednio zmieniają się następujące pola *IsClear*, *IsCross*, *IsZero*:

        public void **SetCross**()//zaznacz krzyżyk  
        {  
            Text=”X”;   
            IsCross=true; IsClear=IsZero=false;  
        }

Analogicznie jest napisana metoda *SetNull* dla zaznaczenia kółka: public void **SetNull**()//zaznacz kółko  
        {  
            Text=”O”;  
            IsClear=IsCross=false; IsZero=true;  
        }

Metoda *SetClear* do wyczyśczenia przycisków:

        public void **SetClear**()//Wyczyść  
        {  
            Text=””;  
            IsClear=true; IsCross=IsZero=false;  
        }  
    }

* 1. ***class Pole -*** pole, które się składa z 9 przycisków do gry.

Class Pole dziedziczy po UserControl, dlatego ma własności UserControl, z których będziemy korzystać.

public partial class UserControl1 : UserControl  
{  
        **GameButton**[,] P = new GameButton [3,3];

**pole do gry, czyli 9 przycisków**

       public event **FinishEventHandlerOnFinish**; //event, czy ktoś wygrał  
        bool active=true;  //aktywny przycisk lub nieaktywny

int Moves=0;  //liczymy,kroki na przypadek remisu  
         
        public UserControl1 ()  
        {  
        InitializeComponent();  
        this.ClientSizeChanged +=new EventHandler(PoleClientSizeChanged);  
  
            //tworzymy tu przyciski  
            for (int r=0; r<3; r++)  
                for (int c=0; c<3; c++)  
            {  
                P[r,c]=new GameButton ();  
                P[r,c].Parent = this;  
                P[r,c].Click += new EventHandler(GameButtonClick);  
            }  
        }  
  
        void **Control**() //sprawdzamy, czy ktoś wygrał               {  
if ( P[0,0].IsCross && P[0,1].IsCross && P[0,2].IsCross || P[1,0].IsCross && P[1,1].IsCross && P[1,2].IsCross ||    P[2,0].IsCross && P[2,1].IsCross && P[2,2].IsCross ||   P[0,0].IsCross && P[1,0].IsCross && P[2,0].IsCross ||    P[0,1].IsCross && P[1,1].IsCross && P[2,1].IsCross ||    P[0,2].IsCross && P[1,2].IsCross && P[2,2].IsCross ||    P[0,0].IsCross && P[1,1].IsCross && P[2,2].IsCross ||    P[0,2].IsCross && P[1,1].IsCross && P[2,0].IsCross)   
{active=false;   
     
}  
if ( P[0,0].IsZero && P[0,1].IsZero && P[0,2].IsZero || P[1,0].IsZero && P[1,1].IsZero && P[1,2].IsZero || P[2,0].IsZero && P[2,1].IsZero && P[2,2].IsZero || P[0,0].IsZero && P[1,0].IsZero && P[2,0].IsZero || P[0,1].IsZero && P[1,1].IsZero && P[2,1].IsZero || P[0,2].IsZero && P[1,2].IsZero && P[2,2].IsZero || P[0,0].IsZero && P[1,1].IsZero && P[2,2].IsZero || P[0,2].IsZero && P[1,1].IsZero && P[2,0].IsZero)  
{active=false;   
     
}  
          
void **PoleClientSizeChanged**(object sender, EventArgs e)  
        {  
        //rozmiary i położenie przycisków  
            for (int r=0; r<3; r++)  
                for (int c=0; c<3; c++)  
            {  
                P[r,c].Width = ClientRectangle.Width/4;  
                P[r,c].Height = ClientRectangle.Height/3;  
                P[r,c].Left = c\*P[r,c].Width;  
                P[r,c].Top = r\*P[r,c].Height;  
            }  
        }  
  
public void **Clear**() //czyszczenie {  
    for (int r=0; r<3; r++)  
        for (int c=0; c<3; c++)  
            P[r,c].SetClear();  
    active=true;   
}  
  
void **GameButtonClick**(object sender, EventArgs e)  
{  
    if ((sender as GameButton).IsClear)  
    if (active) //jesli gra sie nie skonczyla  
    {  
        (sender as GameButton).SetCross(); //zacznaczymy krzyzyk                           
            Control(); //sprawdzamy, czy ktos wygral   
            if (++Moves>8) active=false;// jesli ostatni krok to blokujemy  
            if (active)

            { int kh=0, nr=0, ns=0;  
                // komputer szuka miejsce dla kolka

for (int r=0; r<3; r++)  
            for (int s=0; s<3; s++)  
                if (P[r,s].IsClear)  
                { //sprawdzamy w kolumnie  
                if (kh!=2)   
                { kh=0; for (int rr=0; rr<3; rr++)  
                        if (P[rr,s].IsZero)  
                        { kh++;   
if (kh==2) { ns=s; nr=r; }  
                            }  
                    }  
                            //sprawdzamy w rzędzie  
                if (kh!=2)   
                { kh=0;   
                for (int ss=0; ss<3; ss++)   
                    if (P[r,ss].IsZero)  
                     { kh++;   
if (kh==2) { ns=s; nr=r; }   
                            }  
                }  
                        //sprawdzamy na przekątnej   
                if ((kh!=2) && (r==s))   
                { kh=0;   
                for (int ss=0; ss<3; ss++)   
                    if (P[ss,ss].IsZero)  
                            { kh++;   
if (kh==2) { ns=s; nr=r; }   
                            }  
                }  
                        //sprawdzamy na drugiej przekątnej  
                  if ((kh!=2) && (r==2-s))   
                { kh=0;   
                  for (int ss=0; ss<3; ss++)   
                      if (P[ss,2-ss].IsZero)  
            { kh++;   
if (kh==2) { ns=s; nr=r; }   
                            }  
                }  
                }  
if (kh!=2) //jesli kółko nie wygrywa, to sprawdzamy, jak będą grały krzyżyki

for (int r=0; r<3; r++)  
                    for (int s=0; s<3; s++)  
                        if (P[r,s].IsClear)  
                        { // sprawdzamy w kolumnie  
                        if (kh!=2)   
                        { kh=0;   
                        for (int rr=0; rr<3; rr++)   
                            if (P[rr,s].IsCross)  
                            { kh++;   
if (kh==2) { ns=s; nr=r; }   
                                }  
                        }  
                            // sprawdzamy w rzędzie  
                      if (kh!=2)   
                     { kh=0;   
                     for (int ss=0; ss<3; ss++)   
                         if (P[r,ss].IsCross)  
                         { kh++;   
if (kh==2) { ns=s; nr=r; }   
                                }  
                      }  
                        // sprawdzamy po przekątnej  
                      if ((kh!=2) && (r==s))   
                     { kh=0;   
                            for (int ss=0; ss<3; ss++)   
                          if (P[ss,ss].IsCross)  
                          { kh++;   
if (kh==2) { ns=s; nr=r;}   
                                }  
                      }  
      // sprawdzamy po drugiej przekątnej  
                      if ((kh!=2) && (r==2-s))   
                      { kh=0;   
for (int ss=0; ss<3; ss++)   
                          if (P[ss,2-ss].IsCross)  
                          { kh++;   
if (kh==2) { ns=s; nr=r; }   
                                }  
                      }  
                    }  
        if (kh!=2)  // jeśli kółko nie przegra, to randomowy krok  
                {  
          Random RN = new Random();  
                do {nr = RN.Next(3); ns = RN.Next(3);}  
                           while (!P[nr,ns].IsClear);  
                    }  
                    P[nr,ns].SetNull(); Moves++;  
                    Control(); return;  
                }  
        }  
    }//koniec GameButtonClick  
}//koniec UserControl1

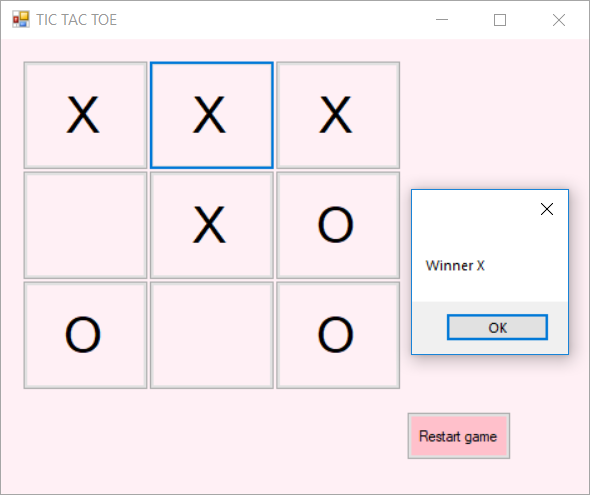
* Kod głównego programu:

public MainForm()  
        {  
            InitializeComponent();  
 }

void Button1Click(object sender, EventArgs e)  
        {     userControl11.Clear();  
        }

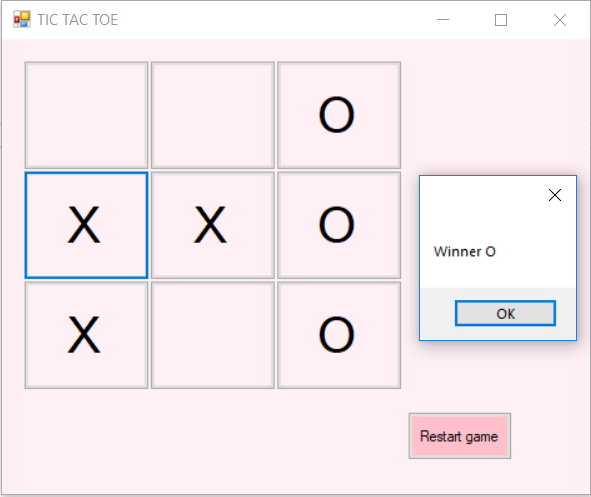
* Testy:

Pokazane dwa przypadki, gdy wygrywa krzyżyk, czyli użytkownik



Rys.3.Wygrywa krzyżyk

i gdy wygrywa kółko, czyli komputer.



Rys.4. Wygrywa kółko