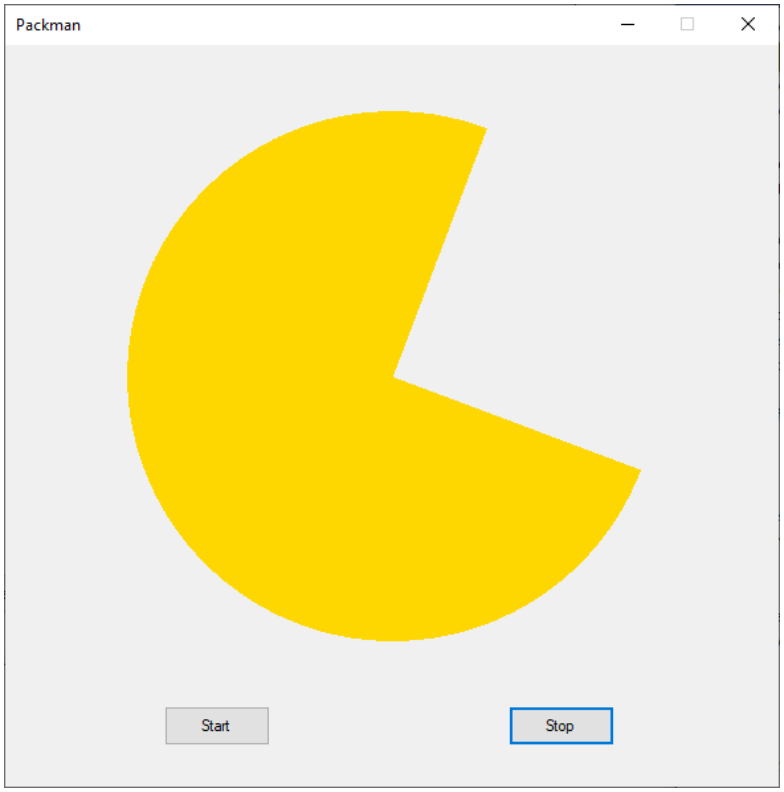
Tutorial 2.

C# Windows Forms Applications

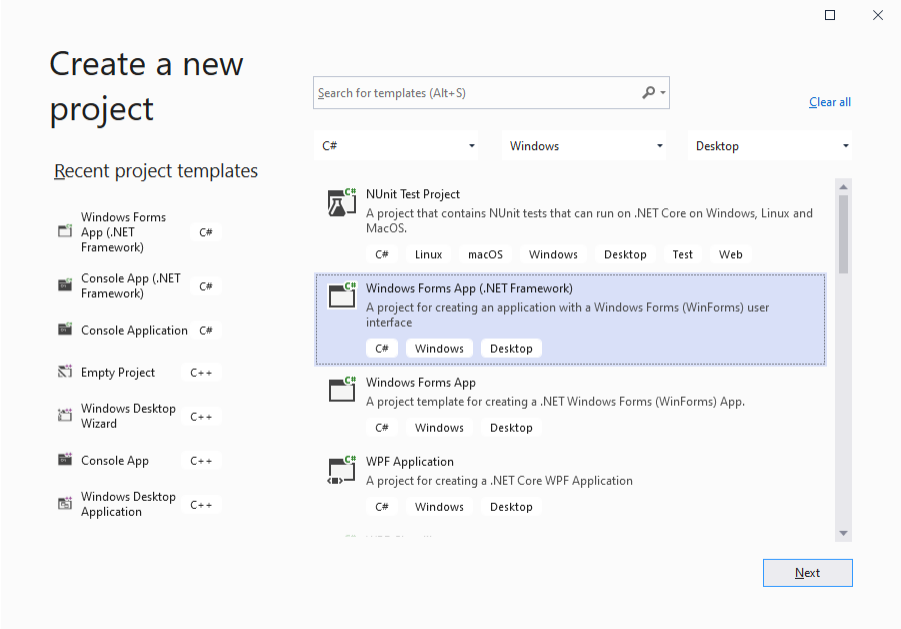
Vom construi cu Visual Studio o aplicație C# care la rulare prezintă pe monitor o formă cu două butoane, etichetate Start și Stop. Primul buton trebuie să lanseze desenarea într-un cerc fix a două raze ortogonale între ele care se rotesc mereu, al doilea trebuie să le oprească:



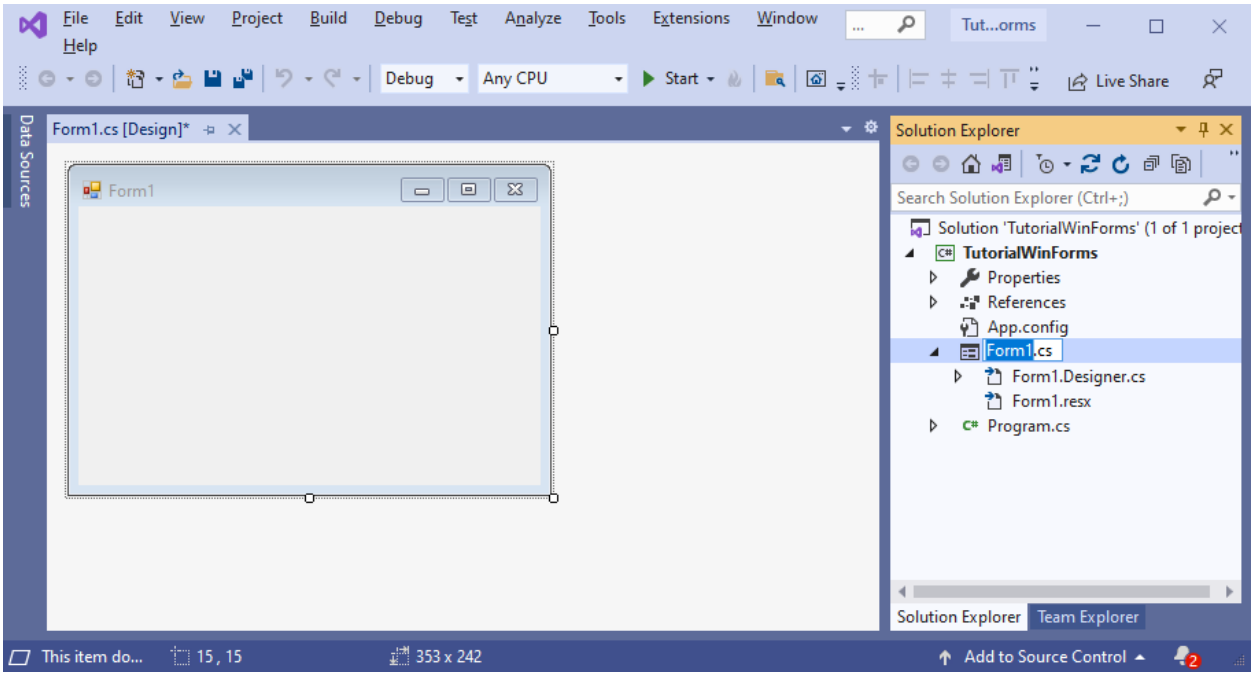
Aici și în cele ce urmează, numim *formă* un obiect din clasa `System.Windows.Forms.Form`, sau dintr-o clasă derivată din aceasta, `System.Windows.Forms.Form` fiind principala clasă .NET pentru realizarea de interfețe grafice GUI (*Graphical User Interface*). Cuvântul `Form` poate fi tradus în limba română prin *formular* sau prin *formă*. Ambele sunt corecte, primul este folosit mai ales în documentația C++ și amintește de întrebuințarea inițială a acestor obiecte folosite ca formulare șablon pentru introducerea de date, iar al doilea, cu sensul de platformă pe care se poziționează diverse alte componente grafice, este folosit în literatura aferentă limbajelor VisualBasic, Delphi și C#.

Inițiem un nou proiect în Visual Studio 2019.

Din fereastra de start alegem `Create a new project` și apoi fixăm în cele trei căsuțe derulante limbajul `C#`, sistemul de operare `Windows` și tipul proiectului `Dektop`, și apoi alegem itemul `C# Windows Forms App(.NET Framework)`.



Salvăm proiectul cu numele `TutorialWinForms`, de exemplu, și în zona editorului găsim afișat *designul* formei `Form1` definită în fișierul `Form1.cs`, fișier *deschis pentru proiectare* vizuală sub denumirea `Form1.cs[design]`. Verificăm editarea vizuală: micșorăm forma cu mausul, de exemplu.

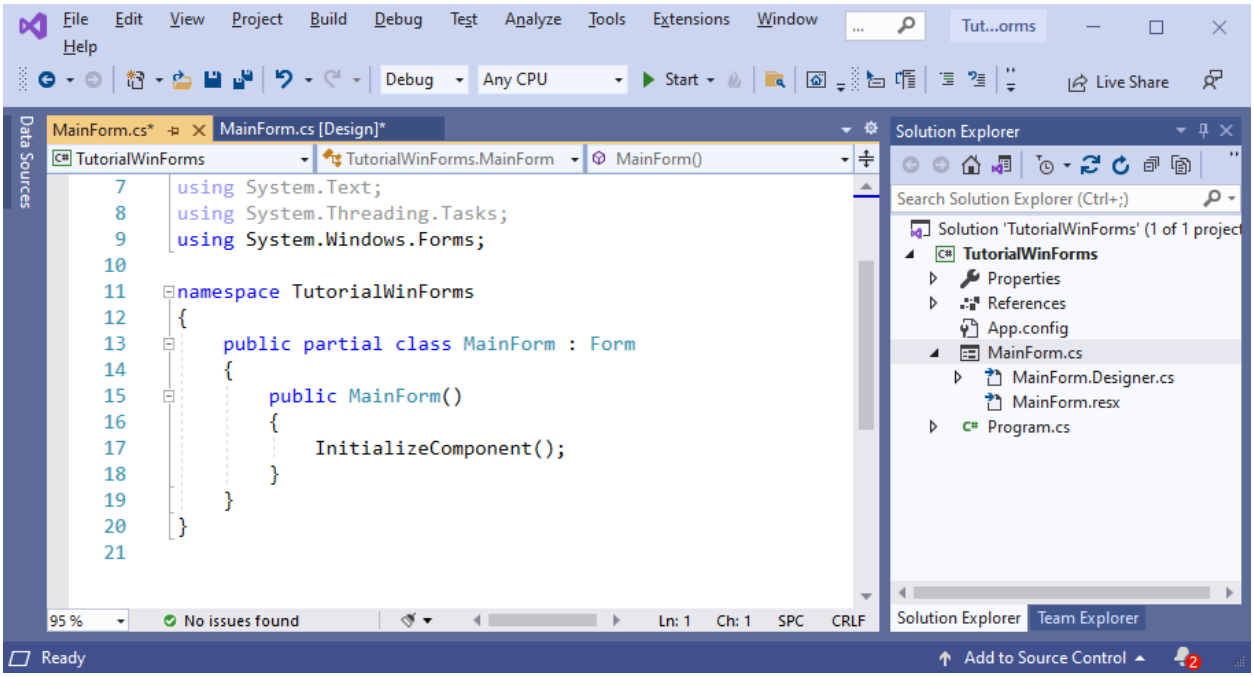


Prima noastră grijă: schimbăm numele fisierului `Form1.cs` în `MainForm.cs`.

Aceasta este o chestiune de stil: într-o astfel de aplicație pot fi folosite mai multe forme, caz în care una trebuie să fie desemnată formă principală și denumită corespunzător.

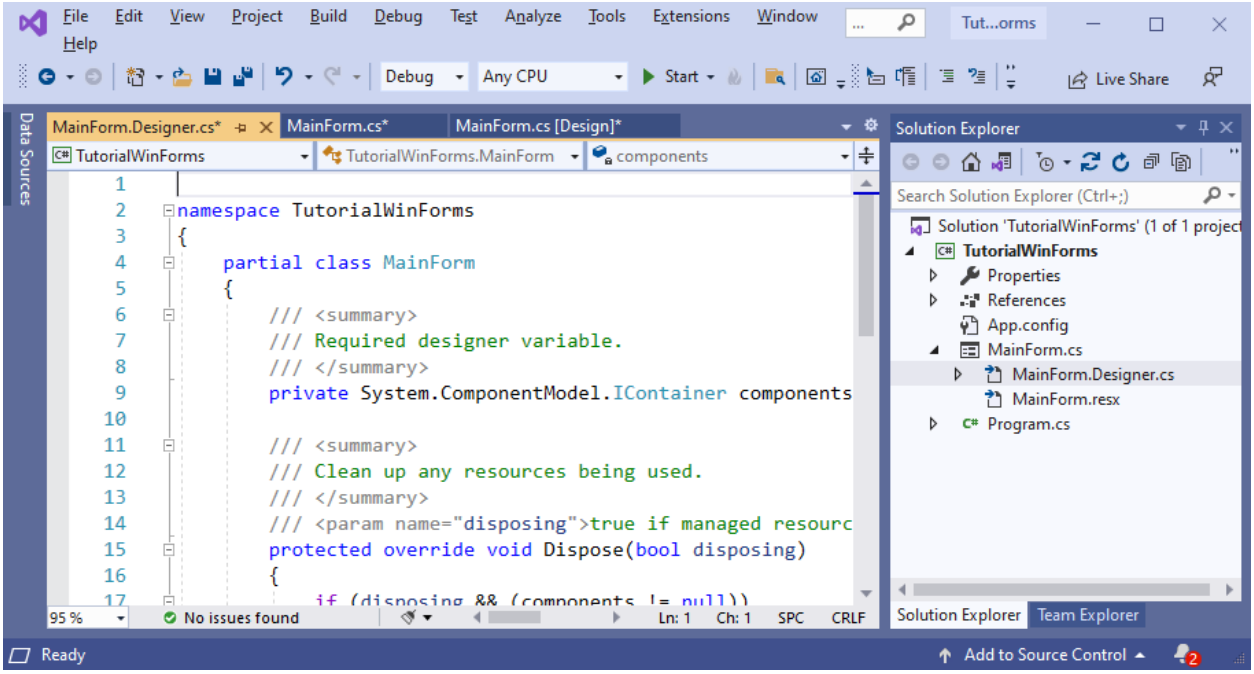
Pentru redenumirea fișierului `Form1.cs` alegem itemul `Rename F2` din meniul contextual atașat (clic-dreapta cu mausul pe numele fișierului în fereastra Solution Explorer), tastăm noul nume, apăsăm `Enter` și răspundem cu `Yes` la întrebarea din ferestra de dialog care apare, astfel încât toate referințele la `Form1` să fie înlocuite cu `MainForm`. Astfel Visual Studio-ul va schimba în mod automat în `MainForm` și numele clasei pe care vrem să o definim în acest proiect.

Verificăm schimbarea *deschizând pentru scriere* fișierul `MainForm.cs` cu `View code (F7)` din meniul contextual atașat:



Proiectarea unei aplicații de tip Windows Forms are loc în două etape. Etapa întâi este o *etapă de proiectare, de design,* în care lucrăm în mod vizual : dimensionăm, de exemplu, forma cu mausul și poziționăm pe ea cu *drag and drop* componentele sale, apoi sta-bilim proprietățile acestor componente și evenimentele la care răspund, alegându-le dintr-un meniu pus la dispoziție de Visual Studio. Etapa a doua este o *etapă de scriere de cod*, în care programăm răspunsul aplicației la evenimentele selectate, completând cu instrucțiunui C# șabloanele handler-elor de evenimente.

In prima etapă suntem asistați de un program, *designerul*, care scrie în locul nostru codul C# corespunzător alegerilor făcute, cod scris în principal în fișierul `MainForm.Designer.cs`. Este bine să studiem acest cod, dar să nu îl modificăm în nici un fel.



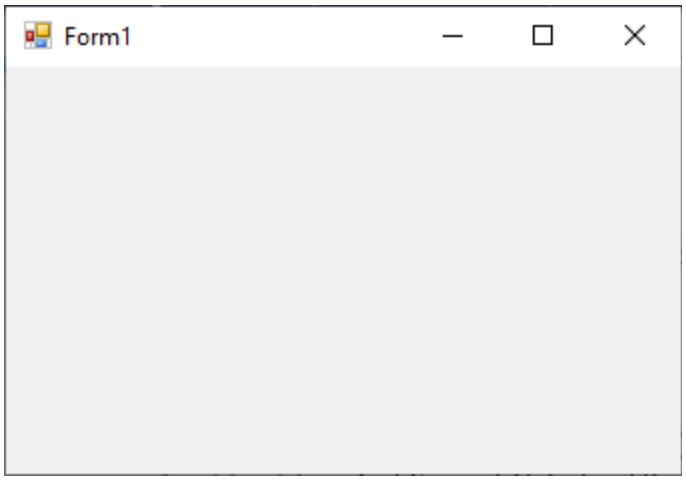
Cele două etape pot fi folosite alternativ, trecerea la etapa de design se face prin deschiderea fișierului `MainForm.cs` cu `View designer (shift + F7)` din meniul contextual atașat, iar trecerea la etapa de scriere a codului cu `View code (F7)`, după cum am văzut deja.

În sfârșit, al treilea fișier de cod este `Program.cs`, gata editat de Visual Studio, în care găsim clasa `Program` , dotată doar cu metoda `Main()` :

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using System.Windows.Forms;
namespace TutorialWinForms
{
    static class Program
    {
        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new MainForm());
        }
    }
}
```

De interes pentru noi este numai ultima instrucțiune, în care este apelată metoda statică `Run` a clasei `Application`, care lansează în funcțiune forma primită ca argument, în acest caz un obiect de tip `MainForm`.

I.1. Setarea proprietăților. Lansăm aplicația, cu `Start Without Debugging, Ctrl + F5`, și pe monitor apare o formă



care nu face nimic, deocamdată, dar care simte și răspunde la următoarele evenimente generate cu mausul : minimizare, maximizare, modificarea dimensiunilor, schimbarea poziției pe ecran și închiderea formei.

Observăm că nu am scăpat total de cuvântul `Form1`, care a mai rămas scris în cod doar în corpul metodei `InitializeComponent()` din fișierul `MainForm.Designer.cs` :

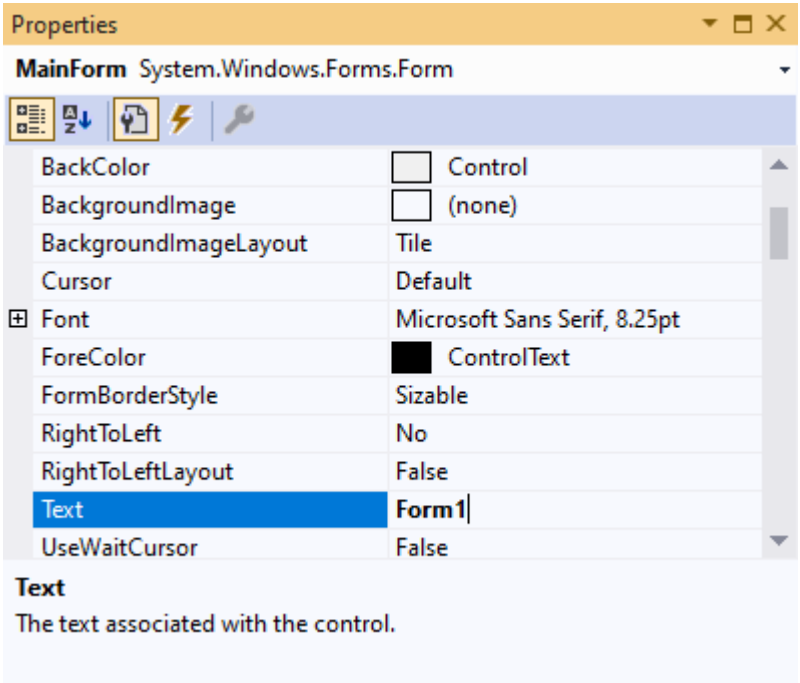
```
private void InitializeComponent()
 {
     this.SuspendLayout();
     //
     // Form1
     //
     this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
     this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
     this.ClientSize = new System.Drawing.Size(337, 203);
     this.Name = "Form1";
     this.Text = "Form1";
     this.ResumeLayout(false);

 }
```

După cum am mai spus, nu este indicat să modificăm manual codul scris de designer, avem voie totuși să înlocuim comentariul[1] `// Form1` cu `// MainForm`. Proprietățile `Name` și `Text` ale clasei `MainForm` le vom modifica cu designerul, după cum urmează.

1 Comentariile cu trei slash-uri `///` sunt folosite de Visual Studio pentru crearea unei documentații XML a proiectului, nu trebuie modificate

Deschidem `MainForm.cs[design]` și selectăm șablonul formei, apoi din meniul View, alegem itemul `Properties Windows, F4`.

In fereastra care apare sunt listate toate[2] proprietățile și toate evenimentele componentei selectate, acum `MainForm`, derivată din `System.Windows.Forms.Form`



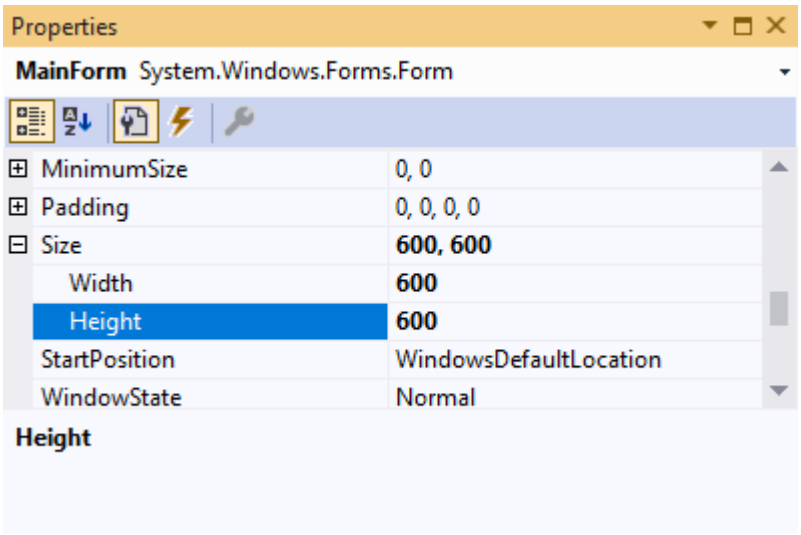
Schimbăm proprietățile `Text` în `Packman` și `Name` în `MainForm`, rulăm aplicația, observăm unde apare `Text`-ul formei, închidem aplicația și apoi verificăm schimbările din fișierul `MainForm.Designer.cs`:

```
private void InitializeComponent()
 {
     this.SuspendLayout();
     //
     // MainForm
     //
     this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
     this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
     this.ClientSize = new System.Drawing.Size(337, 203);
     this.Name = "MainForm";
     this.Text = "Desenator";
     this.ResumeLayout(false);
 }
```

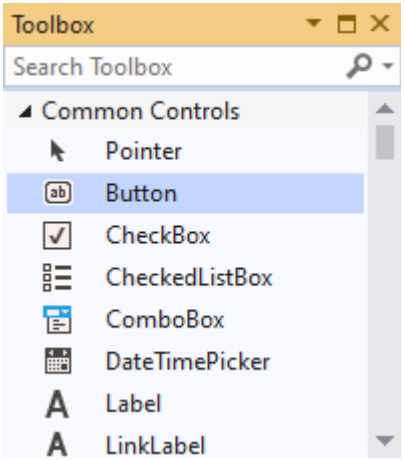
În continuare, vom seta din fereastra Properties alte câteva proprietăți ale formei, cele nemodificate de noi rămânând cu valorile inițializate de constructorul clasei bază.

2 Dacă în fereastra Properties nu apare nimic în listă, se închide fisierul MainForm.cs[design] din editor și se deschide din nou cu dublu-clic pe MainForm.cs în Solution Explorer .

Mai precis, setăm `FormBorderStyle` la **`FixedDialog`** și `MaximizeBox` la **`false`**, ca să interzicem modificarea la rulare a dimensiunilor formei, pe care le setăm apoi la 600 x 600 pixeli, cu `Size.Width` și `Size.Height`:

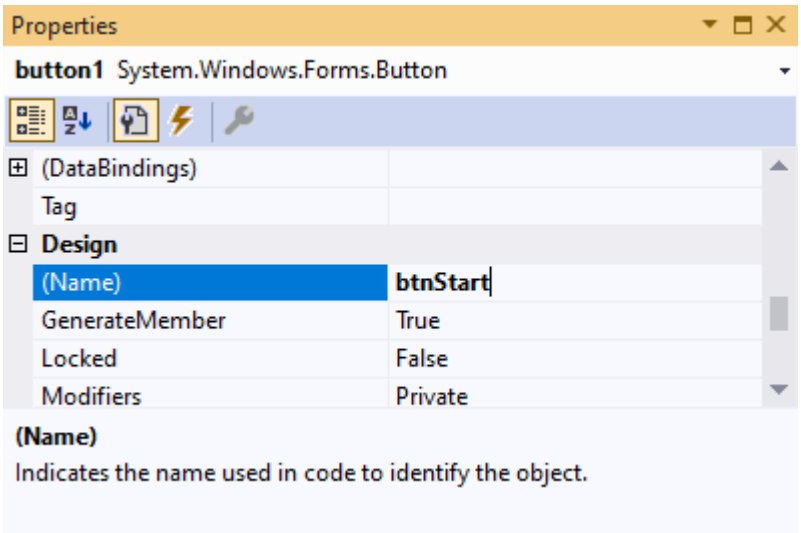


Am terminat cu proprietățile formei, să o dotăm acum cu două butoane. Păstrăm deschis `MainForm.cs[design]` și din meniul View selectăm itemul `Toolbox` `Ctrl+Alt+X` care ne deschide fereastra Toolbox, în care putem alege cu mausul componenta dorită.



Acum ne alegem un buton: cu dublu-clic pe itemul `Button` apare pe formă un dreptunghi pe care scrie `button1`, îl selectăm cu mausul și în fereastra Properties îi setam următoarele: `Name` – **`btnStart`**, `Text` – **`Start`**, `Location.X` – **`120`**, `Location.Y` – **`500`**, `Size.Width` – **`80`** și `Size.Height` – **`30`**.

Aveți grijă ca pe formă să fie selectat butonul, astfel încât în căsuța derulantă de sub bara de titlu a ferestrei Properties să apară numele și clasa componentei setate, în acest caz **`button1`** de tip `System.Windows.Forms.Button`



Adăugăm al doilea buton, tot cu dublu-clic pe itemul `Button` din fereastra Toolbox, iarăși apare pe formă un dreptunghi pe care scrie tot `button1`. Lui îi setăm următoarele proprietăți: `Name` – **`btnStop`**, `Text` – **`Stop`**, `Location.X` – **`380`**, `Location.Y` – **`500`**, `Size.Width` – **`80`** și `Size.Height` – **`30`**.

Verificăm ce am făcut până acum: rulăm aplicația, observăm că pe formă au apărut două butoane care nu fac nimic deocamdată, închidem forma și inspectăm `MainForm.Designer.cs`

```
namespace TutorialWinForms
{
    partial class MainForm
    {
        ...

        #region Windows Form Designer generated code

        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
        {
            this.btnStart = new System.Windows.Forms.Button();
            this.btnStop = new System.Windows.Forms.Button();
            this.SuspendLayout();
            //
            // btnStart
            //
            this.btnStart.Location = new System.Drawing.Point(120, 500);
            this.btnStart.Name = "btnStart";
            this.btnStart.Size = new System.Drawing.Size(80, 30);
            this.btnStart.TabIndex = 0;
            this.btnStart.Text = "Start";
```

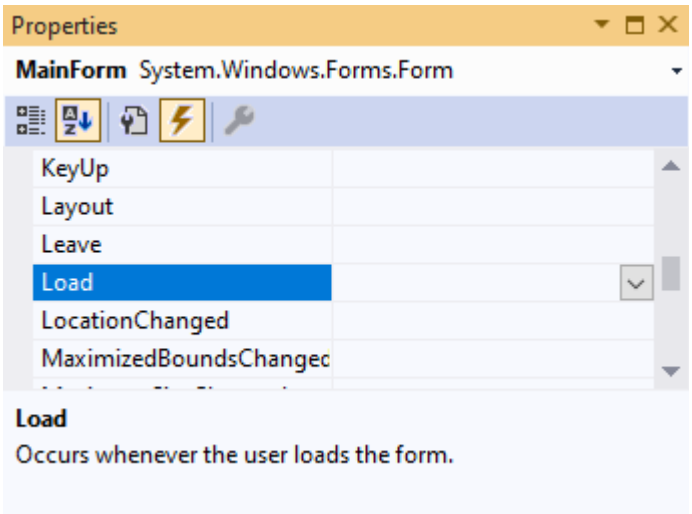
```
            this.btnStart.UseVisualStyleBackColor = true;
            //
            // btnStop
            //
            this.btnStop.Location = new System.Drawing.Point(380, 500);
            this.btnStop.Name = "btnStop";
            this.btnStop.Size = new System.Drawing.Size(80, 30);
            this.btnStop.TabIndex = 1;
            this.btnStop.Text = "Stop";
            this.btnStop.UseVisualStyleBackColor = true;
            //
            // MainForm
            //
            this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
            this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
            this.ClientSize = new System.Drawing.Size(584, 561);
            this.Controls.Add(this.btnStop);
            this.Controls.Add(this.btnStart);
            this.FormBorderStyle = System.Windows.Forms.FormBorderStyle.FixedDialog;
            this.MaximizeBox = false;
            this.Name = "MainForm";
            this.Text = "Desenator";
            this.ResumeLayout(false);

        }

        #endregion

        private System.Windows.Forms.Button btnStart;
        private System.Windows.Forms.Button btnStop;
    }
}
```

În ultimele două linii de cod observăm că acum clasa `MainForm` este dotată cu două câmpuri private, `btnStart` și `btnStop`, ambele de tip `System.Windows.Forms.Button`. Atenție: butoanele nu sunt forme, clasa `Button` este definită în spațiul de nume `System.Windows.Forms`, în care este definită și clasa `Form`, în acest spațiu de nume sunt incluse multe alte clase utilizate la derivarea clasei `Form`.

I.2. Setarea evenimentelor. În continuare vom alege, tot cu ajutorul designerului, evenimentele la care trebuie să modificăm răspunsul implicit al formei și al butoanelor sale.

Începem cu forma: în editor deschidem `mainform.cs[design]` și selectăm forma, iar în fereastra properties selectăm tab-ul `events` (pictograma fulger).

Rularea aplicației începe cu *încărcarea* formei principale pe ecran, și noi dorim ca la acel moment să aibă loc unele acțiuni descrie mai tarziu în codul sursă, prin urmare acum doar selectam evenimentul `Load` și executăm un dublu-clic pe itemul corespunzător.



Se deschide automat fișierul `MainForm.cs`

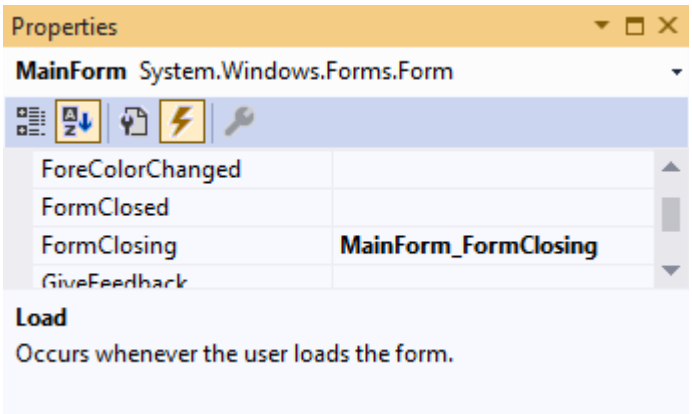
```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace TutorialWinForms
{
    public partial class MainForm : Form
    {
        public MainForm()
        {
            InitializeComponent();
        }

        private void MainForm_Load(object sender, EventArgs e)
        {

        }
    }
}
```

în care observăm că `MainForm` a fost dotată cu metoda `MainForm_Load()`, iar dacă deschidem fișierul în `MainForm.Designer.cs` vedem că în corpul metodei `InitializeComponent()` a apărut instrucțiunea

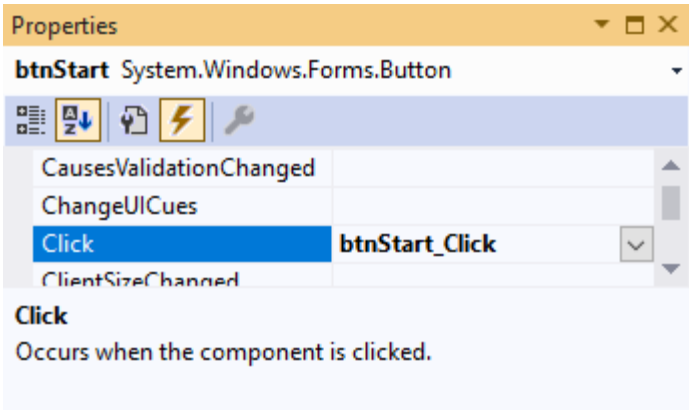
```
this.Load += new System.EventHandler(this.MainForm_Load);
```

care adaugă metoda `MainForm_Load` în lista acțiunilor executate la încărcarea formei pe ecran. Altfel spus, `MainForm_Load()` este un *handler de eveniment* pentru `MainForm.Load`. Vom reveni cu detalii într-un tutorial viitor.

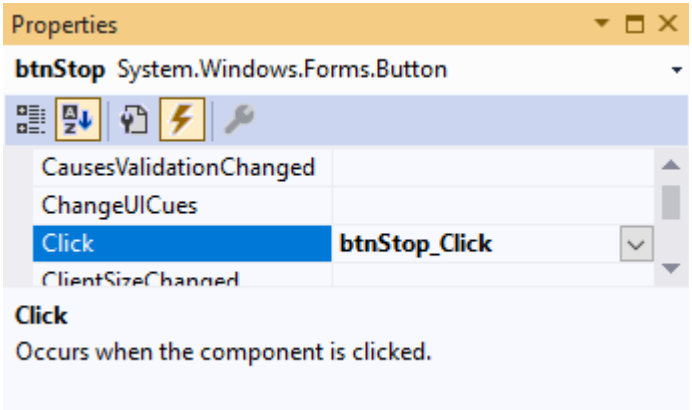
Codul metodei `MainForm_Load()` va fi completat manual în etapa a II-a, acum revenim la `MainForm.cs[Design]` și în fereastra Properties adăugăm, tot cu dublu-clic, un handler pentru evenimentul `FormClosing`:



Trecem la răspunsul butoanelor la evenimente, selectam în editor butonul Start și îi adăugăm un handler pentru evenimentul Click



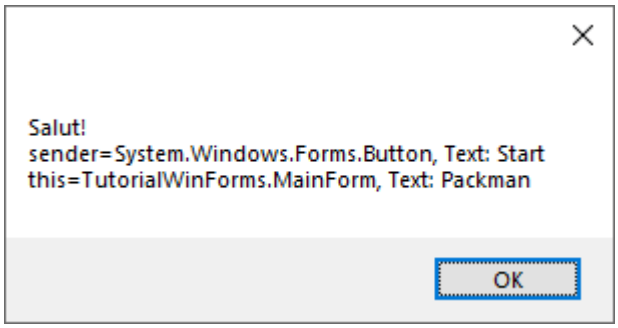
și repetăm operația pentru butonul Stop.



Verificăm ce am realizat până acum: în `MainForm.cs` completăm provizoriu metodele

```
private void btnStart_Click(object sender, EventArgs e)
{
    MessageBox.Show("Salut!\nsender=" + sender + " this=" + this);
}

private void btnStop_Click(object sender, EventArgs e)
{
    MessageBox.Show("Bye-bye!\nsender=" + sender + " this=" + this);
}
```

și lansăm aplicația. Acum butoanele răspund la apăsare:



Ca regulă generală: expeditorul unui mesaj provocat de maus, *sender*-ul, este componenta asupra căreia a acționat mausul, și tot ea trebuie să furnizeze răspunsul la mesaj, folosind eventual informațiile atașate evenimentului în parametrul `EventArgs e`.

II. Scrierea codului. Am ajuns la etapa în care implementăm comportamentul dorit al butoanelor: Start lansează desenarea iar Stop o oprește.

Pentru început vom completa `MainForm.cs` cu codul de mai jos, apoi vom rula programul și la final vom da explicațiile necesare. Tastăm:

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

namespace TutorialWinForms
{
    public partial class MainForm : Form
    {
        Graphics dc;        //Device Context
        Pen pen1, pen2;     //pentru tras linii colorate
        bool seLucreaza;    //semafor
```

```
public MainForm()
{
    InitializeComponent();
    seLucreaza = false;
    pen1 = new Pen(Color.Gold, 1);
    pen2 = new Pen(this.BackColor, 1);

}

private void MainForm_Load(object sender, EventArgs e)
{
    btnStart.Focus();
    dc = this.CreateGraphics();
}

private void Deseneaza()
{
    double raza = 200;
    double x0 = this.ClientSize.Width / 2, y0 = this.btnStart.Location.Y / 2;
    double theta = Math.PI / 2;     //unghiul dintre raze
    double delta = Math.PI / 2000;  //pasul lui t discretizat
    for (double t = 0; seLucreaza; t+=delta)
    {
        double xt = x0 + raza * Math.Cos(t);
        double yt = y0 + raza * Math.Sin(t);
        dc.DrawLine(pen1, (float)x0, (float)y0, (float)xt, (float)yt);
        double xp = x0 + raza * Math.Cos(t + theta);
        double yp = y0 + raza * Math.Sin(t + theta);
        dc.DrawLine(pen2, (float)x0, (float)y0, (float)xp, (float)yp);
        Application.DoEvents();
    }
}

private void btnStart_Click(object sender, EventArgs e)
{
    //MessageBox.Show("Salut!\nsender=" + sender + " this=" + this);
    btnStop.Focus();
    if (seLucreaza) return;
    this.Refresh(); //repicteaza forma la starea initiala
    seLucreaza = true;
    Deseneaza();
    btnStart.Focus();
}

private void btnStop_Click(object sender, EventArgs e)
{
    //MessageBox.Show("Bye-bye!\nsender=" + sender + " this=" + this);
    seLucreaza = false;
    btnStart.Focus();
}
```

```
        private void MainForm_FormClosing(object sender, FormClosingEventArgs e)
        {
            seLucreaza = false;
        }
    }
}
```

Am construit o aplicație condusă de evenimente, o *event-driven application*, la rulare ea afișează pe ecrean o formă care stă și așteaptă să se producă un eveniment care să o vizeze. După ce răspunde la eveniment, revine în starea de așteptare, starea *idle*.

Observăm că la prima afișare *focusul* este deținut de butonul Start, care poate fi actionat astfel și cu tasta *space*. Această setare inițială este scrisă în `MainForm_Load()`.

Apăsarea butonului Start, tratată în `btnStart_Click()`, mută focusul pe butonul Stop, lansează desenarea cu apelul metodei `Deseneaza()`, și la terminarea acestui apel, care are loc numai după apăsarea butonului Stop, își trage focusul înapoi.

Pentru a stopa desenarea am folosit un semafor logic, câmpul `bool seLucreaza`, desenarea are loc atât timp cât acesta are valoarea `true`, la `false` se oprește. Prin urmare, apăsarea butonului Stop setează `seLucreaza` la `false`, iar apăsarea butonului Start îl setează la `true`, înaintea apelului metodei `Deseneaza()`.

Deoarece oprirea ciclării din for-ul din `Deseneaza()` are loc numai prin setarea la `false` a semaforului, dacă vom închide forma în timpul desenării aceasta va fi doar ascunsă, devenind invizibilă, dar aplicația va rămâne în execuție în fundal, consumând pe ascuns resursele sistemului, și va putea fi oprită numai din Task Manager. Ca să prevenim acest *bug* dezastruos utilizăm evenimentul `FormClosing`, care se produce când este apăsat cu mausul căsuța `X` din colțul din dreapta sus al formei. În `MainForm_FormClosing()` setăm `seLucreaza = false`, oprim astfel desenarea dacă este în desfășurare, iar aplicația iese din rulare odată cu închiderea formei.

Ca să desenăm direct pe suprafața formei avem nevoie de un "context de dispozitiv", un *device context,* o interfață între sistemul de operare și placa grafică. În C# acesta este obținut prin utilizarea unui obiect din clasa `System.Drawing.Graphics`, pe care este bine să nu îl instanțiem în constructorul clasei, ci la încărcarea formei, prin apelul metodei `CreateGraphics()`. Astfel suntem siguri că interfața grafică va funcționa corect.

Desenul este format din linii trasate cu metoda clasei `System.Drawing.Graphics`

```
void DrawLine(System.Drawing.Pen pen, float x1, float y1, float x2, float y2);
```

și pentru aceasta folosim două penițe, `pen1` și `pen2`, inițializate în constructor. Metoda unește punctele (x_1,y_1) și (x_2,y_2) cu o linie de culoare și grosime setată în penița folosită. Coordonatele sunt relative la *suprafața client* a formei, cea de sub bara de titlu, originea este în colțul de stânga sus, x crește de la stânga la dreapta, iar y de sus în jos.

În metoda `Deseneaza()` fixăm punctul (x_0,y_0) în centrul suprafeței client a formei și parcurgem cercul cu centrul în (x_0,y_0) și cu raza de 200 pixeli folosind binecunoscutele ecuații parametrice ale cercului:

$$\begin{cases} x(t)=x_0+r\cos t \\ y(t)=y_0+r\sin t \end{cases}$$

O importanță deosebită o are apelul `Application.DoEvents()` din interiorul ciclului `for`, care obligă aplicația să întrerupă desenarea și să proceseze evenimentele care s-au produs între timp, de la precedentul apel, actualizând astfel valoarea semaforului `seLucreaza` dacă a fost apasat butonul Stop. Fără acest apel ciclul `for` ar deveni o buclă infinită cu efect dezastruos la rulare. La prima apăsare a butonului Start forma ar îngheța, aplicația nu ar mai răspunde la mesaje și ar putea fi oprită numai din Task Manger. Încercați!

Proiectul `TutorialWinForms` prezentat aici, ca o introducere în utilzarea clasei `Form`, rezolvă problema dar are un mare neajuns: desenul nostru este volatil, de exemplu dispare dacă minimizăm și apoi maximizăm forma! Pentru a obține o imagine stabilă este nevoie de o abordare mai elaborată, în care să dispunem de un bitmap[3] în care să desenăm, bitmap care să poată fi salvat pe disc, dacă dorim, într-un format grafic, `.bmp` sau `.jpg`, de exemplu.

3 Prin bitmap se înțelege, în general, o matrice de întregi, elementul c(i,j) fiind culoarea pixelului de coordonate (i,j) codificată într-un număr întreg. Aici prin bitmap înțelegem un obiect din clasa System.Drawing.Bitmap.