

# Course 2DCis: 2D-Computer Graphics with C#

## Chapter C1: The Intro Project

Copyright © by V. Miszalok, last update: 09-12-2007

- ↓ [An empty window](#)
- ↓ [DrawString: Hallo World](#)
- ↓ [Print window size with color font](#)
- ↓ [Left, right, top, bottom](#)
- ↓ [Line, Rectangle, Ellipse](#)
- ↓ [Draw a star with random rays and random colors](#)
- ↓ [Draw a polygon](#)
- ↓ [Animate it](#)
- ↓ [Exercises](#)
- ↓ [Mini-programs for training](#)

### An empty window

Guidance for **Visual Studio 2008**:

0) Main Menu after start of VS 2008: `Tools` → `Options` → check lower left checkbox: `Show all Settings` → `Projects and Solutions` → `Visual Studio projects location:` → `C:\temp`

1) Main Menu after start of VS 2008: `File` → `New Project...` → `Visual Studio installed templates: Windows Forms Application`  
 Name: `intro1` → Location: `C:\temp` → `Create directory for solution: switch off` → `OK`  
`Form1.cs[Design]` appears.

2) Two superfluous files must be deleted: `Form1.Designer.cs` and `Program.cs`.  
 You reach these files via the `Solution Explorer` - `intro1-window`:  
 Click the plus-sign in front of branch `intro1` and the plus-sign in front of branch `Form1.cs`.  
 Right-click the branch `Program.cs`. A context menu opens. Click `Delete`. A message box appears:  
 'Program.cs' will be deleted permanently. Quit with `OK`.  
 Right-click the branch `Form1.Designer.cs` and delete this file too.

3) Right-click the gray window `Form1`. A small context menu opens. Click `View Code`.  
 You see now the preprogrammed code of `Form1.cs`. Erase this code completely.

4) Write the following three lines into the empty `Form1.cs`:

```
public class Form1 : System.Windows.Forms.Form
{ static void Main() { System.Windows.Forms.Application.Run( new Form1() ); }
}
```

5) Click `Debug` in the main menu of VS 2008.

A submenu opens. Click `Start Without Debugging Ctrl F5`.

The rudimentary program now automatically compiles, links and starts. Please observe the `Error List`-window of Visual Studio below our program.

Our program starts automatically as stand-alone window containing three parts:

1. main window = `MainFrame` with a blue title row
2. three buttons `Minimize`, `Maximize`, `Close`
3. a narrow frame with 4 movable borders and 4 movable edges. Please enlarge the window by dragging its borders and edges.

This functionality has been inherited from the .NET-class `System.Windows.Forms.Form`.

Minimize VisualStudio, to realize that `intro1.exe` is a stand-alone windows program.

Start the `Explorer`. Branch to `C:\temp\intro1\bin\debug`.

Double click `intro1.exe`. You can start an arbitrary number of instances of `intro1.exe` (You must carefully kill all before You write new versions.). Minimize the `Explorer`.

**Important: Always finish all instances of `intro1` before writing new code and starting it !**

**Start the Task Manager with `Ctrl+Alt+Del` and check if any `intro1.exe`-process is still running. If yes, kill it.**

## DrawString: Hallo World

If You don't see Your code anymore, click the tab **Form1.cs** of Visual Studio.

Read at first the **Important Tip 2** below in this chapter, otherwise the automatic formats of the code editor will drive You into madness.

Please delete everything (including the last brace). No code remains.

Write the following lines into the empty window of **Form1.cs**:

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

public class Form1 : Form
{ static void Main() { Application.Run( new Form1() ); }
  Font arial18 = new Font( "Arial", 18 );
  Brush blackbrush = SystemBrushes.ControlText;
  protected override void OnPaint( PaintEventArgs e )
  { Graphics g = e.Graphics;
    g.DrawString( "Hello world !", arial18, blackbrush, 0, 0 );
  }
}
```

Click the **Debug**-tab of the main menu above the main window.

A sub-menu opens. Click **Start Without Debugging** **Ctrl F5**.

The program automatically compiles, links and starts again.

Let us finish this primitive version and let's program a constructor `public Form1()`:

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

public class Form1 : Form
{ static void Main() { Application.Run( new Form1() ); }
  Font arial18 = new Font( "Arial", 18 );
  Brush blackbrush = SystemBrushes.ControlText;
  public Form1()
  { BackColor = Color.White;
    Text = "introl";
    SetStyle( ControlStyles.ResizeRedraw, true );
    StartPosition = FormStartPosition.Manual;
    Top = 50;
    Left = 50;
    Width = 800;
    Height = 600;
  }
  protected override void OnPaint( PaintEventArgs e )
  { Graphics g = e.Graphics;
    String s0 = "Hello world, here is introl !";
    g.DrawString( s0, arial18, blackbrush, 0, 0 );
  }
}
```

Click the **Debug**-tab of the main menu above the main window.

A sub-menu opens. Click **Start Without Debugging** **Ctrl F5**.

The program automatically compiles, links and starts again.

**Exercises:** Try out other start positions and other window sizes, other strings, other font sizes, other fonts (f.i. **Courier New**), other colors (f.i. `Color.Blue`, `Color.Green`) and other text positions.

**Important Tip:** In case of mistype the compiler presents a Message Box: There were build errors. . . . You quit with **No**. An `Error List`-window with warnings and errors will appear in Visual Studio below Your program. In this error list scroll up to the first error (ignore the warnings !). Double click the line with the first error. The cursor jumps automatically into Your code into the line where the error was detected. Look for mistypes in this line and remove them. (Sometimes You will not find the error in this line but above, where You forgot a comma or a semicolon.) Ignore all errors below the first error (in most cases they are just followers of the first one) and compile. Repeat this procedure until further error message boxes disappear and Your program compiles, links and starts as expected.

If You use Visual Studio 2008 Professional You should switch off the vexatious automatic format- and indent- mechanism of the code editor:

1. Main menu of Visual Studio 2008 Professional: Click menu "Tools".
2. A DropDown-menu appears. Click "Options...".
3. An Options dialog box appears.
4. Click the branch "Projects and Solutions". Click "General". Change all three paths to C:\temp.
5. Click the branch "Text Editor", then "C#".
6. A subtree appears with the following branches: "General, Tabs, Advanced, Formatting, IntelliSense".
7. Click "Tabs". Change "Indenting" to None, "Tab size" and "Indent size" to 1 and switch on the option "Insert spaces".
8. Within the branch "C#" click the plus-sign in front of "Formatting" and change all "Formatting" branches: "General": switch off all check boxes, "Indentation": switch off all check boxes, "New Lines": switch off all check boxes, "Spacing": switch off all check boxes, "Wrapping": switch **on** both check boxes.
9. Quit the dialog with button "OK".

## Print window size with color font

Version2: Finish `introl`.

In the head of class `Form1()` clear both lines: `Font arial18 = new Font("Arial", 18);` and `Brush blackbrush = SystemBrushes.ControlText;` and replace them by:

```
Font arial18      = new Font( "Arial", 18 );
Font arial16      = new Font( "Arial", 16 );
Font courier14    = new Font( "Courier New", 14 );
Brush blackbrush  = SystemBrushes.ControlText;
Brush redbrush    = new SolidBrush( Color.Red );
Brush whitebrush  = new SolidBrush( Color.White );
Pen blackpen      = new Pen( Color.Black, 4 );
```

Delete the 5-line-function protected override `void OnPaint(...)` and rewrite it from scratch:

```
protected override void OnPaint( PaintEventArgs e )
{ //Version 2 *****
  Graphics g = e.Graphics;
  Rectangle cr = ClientRectangle;
  String s0 = "Hello world, here is introl !";
  String s1 = "Change the size of your window by dragging a corner !";
  String s2w = "Form : Width = " + Width.ToString();
  String s3w = "Client: Width = " + cr.Width.ToString();
  String s2h = " Height= " + Height.ToString();
  String s3h = " Height= " + cr.Height.ToString();
  g.DrawString( s0, arial18, blackbrush, 0, 0 );
  g.DrawString( s1, arial16, redbrush, 0, 20 );
  g.DrawString( s2w + s2h, courier14, blackbrush, 0, 40 );
  g.DrawString( s3w + s3h, courier14, blackbrush, 0, 60 );
}
```

Click `Debug` in the main menu above the main window and `Start Without Debugging` `Ctrl F5`.

## Left, right, top, bottom

Version3: Finish `intro1`.

Write 6 additional lines of code in protected override `void OnPaint(...)` below the existing code, but in front of the brace which closes `...OnPaint(...)`:

```
//Version 3 *****
Point mid = new Point( cr.Width/2, cr.Height/2 );
g.DrawString( "left" ,arial16, blackbrush, 0 , mid.Y );
g.DrawString( "right" ,arial16, blackbrush, cr.Width-50, mid.Y );
g.DrawString( "top" ,arial16, blackbrush, mid.X , 0 );
g.DrawString( "bottom",arial16, blackbrush, mid.X , cr.Height-30 );
```

Click Debug and Start Without Debugging Ctrl F5.

## Line, Rectangle, Ellipse

Version4: Finish `intro1`.

Write 8 additional lines of code in protected override `void OnPaint(...)` below the existing code, but in front of the brace which closes `...OnPaint(...)`:

```
//Version 4 *****
g.DrawLine( blackpen, 0, 0, cr.Width, cr.Height );
g.DrawLine( blackpen, cr.Width, 0, 0, cr.Height );
Int32 w5 = cr.Width / 5;
Int32 h5 = cr.Height / 5;
g.FillRectangle( whitebrush, w5, h5, 3 * w5, 3 * h5 );
g.DrawRectangle( blackpen , w5, h5, 3 * w5, 3 * h5 );
g.DrawEllipse ( blackpen , w5, h5, 3 * w5, 3 * h5 );
```

Click Debug and Start Without Debugging Ctrl F5.

## Draw a star with random rays and random colors

Version5: Finish `intro1`.

Define a new Pen in the head of public class `Form1 : Form`

directly below the line `Pen blackpen = new Pen( Color.Black, 4 );`:

```
Pen randompen = new Pen( Color.Black, 20 );
```

Write additional lines of code in protected override `void OnPaint(...)` below the existing code as You did with version 3 and 4:

```
//Version 5 *****
Int16 i, nn = 120;
Int32 red, green, blue;
randompen.EndCap = System.Drawing.Drawing2D.LineCap.DiamondAnchor;
Point[] splash = new Point[nn];
Double arcus_1 = 2.0 * Math.PI / nn;
Double arcus_i, factor, sinus, cosinus;
Double radius_x = 1.35 * w5;
Double radius_y = 1.35 * h5;
Random random = new Random();
for ( i=0; i < nn; i++ )
{
    red = random.Next( Byte.MaxValue );
    green = random.Next( Byte.MaxValue );
    blue = random.Next( Byte.MaxValue );
    randompen.Color = Color.FromArgb( red, green, blue );
    factor = Math.Max( 0.25, random.NextDouble() );
    arcus_i = arcus_1 * i;
    cosinus = radius_x * factor * Math.Cos( arcus_i );
    sinus = radius_y * factor * Math.Sin( arcus_i );
    g.DrawLine( randompen, mid.X, mid.Y, mid.X + (Int32)cosinus, mid.Y + (Int32)sinus );
    splash[i].X = mid.X + (Int32)(cosinus * 0.8);
    splash[i].Y = mid.Y + (Int32)( sinus * 0.8);
}
}
```

Click Debug and Start Without Debugging Ctrl F5.

Try out what happens when You drag one of the window borders.

Theory see: [././Lectures/L02\\_2DVector/2DVector\\_english.htm#a11](http://www.csharp-examples.com/Lectures/L02_2DVector/2DVector_english.htm#a11)

## Draw a polygon

Version6: Finish `intro1`.

Write two additional command lines into protected override `void OnPaint(...)` directly under the brace which closes the `for`-loop:

```
//Version 6 *****
g.FillClosedCurve( redbrush, splash );
g.DrawString( "splash !", arial18, whitebrush, mid.X - 40, mid.Y - 9 );
```

Click `Debug` and `Start Without Debugging` `Ctrl F5`.

Try out what happens when You drag one of the window borders.

## Animate it

Version7: Finish `intro1`.

Write two additional command lines into protected override `void OnPaint(...)` below the last command line `g.DrawString( "splash !", arial18, whitebrush, mid.X - 40, mid.Y - 9 );`, but above the two lines with the closing braces:

```
//Version 7 *****
System.Threading.Thread.Sleep( 100 ); //Wait 100 milliseconds.
Invalidate(); //Ask the operating system to raise the Paint event again.
```

Click `Debug` and `Start Without Debugging` `Ctrl F5`.

## Exercises

Click `Help` in the main menu of Visual Studio. Click the sub-menu `Index`.

Go to `Filtered by:` and choose: `.NET Framework SDK`. Then enter into `Look for:` one of the following key words: `DrawString`, `DrawLine`, `Random`, `FillClosedCurve`, `Pen`, `Brush` etc. You obtain a selection of key words beginning with those characters. Read the help texts. The help window covers Your code. When you finished reading, get rid of it with the X-button in the upper right window corner.

Finish Visual Studio, start the Explorer, delete the complete directory `C:\temp\intro1`.

Start Visual Studio again and try to write this program by heart.

Writing the code use `Drag&Drop` = move and copy using the mouse with and without the `Strg`-key.

Invent and try out new variants (in form of new projects `Intro2`, `Intro3` etc.), f.i. parallel horizontal colored lines, vertical parallel lines, colored rectangles and ellipses at random positions etc.

# Mini-programs for training

## 1. MiniHello:

```
using System;
using System.Drawing;
using System.Windows.Forms;
public class Form1 : Form
{ static void Main() { Application.Run( new Form1() ); }
  Font arial10 = new Font( "Arial", 10 );
  Brush blackbrush = SystemBrushes.ControlText;
  public Form1()
  { SetStyle( ControlStyles.ResizeRedraw, true );
  }
  protected override void OnPaint( PaintEventArgs e )
  { Graphics g = e.Graphics;
    Rectangle cr = ClientRectangle;
    g.DrawString( "Hello", arial10, blackbrush, cr.Width/2-10, cr.Height/2-5 );
  }
}
```

## 2. MiniCross:

```
using System;
using System.Drawing;
using System.Windows.Forms;
public class Form1 : Form
{ static void Main() { Application.Run( new Form1() ); }
  Pen blackpen = new Pen( Color.Black, 4 );
  public Form1()
  { SetStyle( ControlStyles.ResizeRedraw, true );
  }
  protected override void OnPaint( PaintEventArgs e )
  { Graphics g = e.Graphics;
    Rectangle cr = ClientRectangle;
    g.DrawLine( blackpen, 0, 0, cr.Width, cr.Height );
    g.DrawLine( blackpen, cr.Width, 0, 0, cr.Height );
  }
}
```

## MiniCircle with random colors:

```
using System;
using System.Drawing;
using System.Windows.Forms;
public class Form1 : Form
{ static void Main() { Application.Run( new Form1() ); }
  Pen blackpen = new Pen( Color.Black, 4 );
  public Form1()
  { SetStyle( ControlStyles.ResizeRedraw, true );
  }
  protected override void OnPaint( PaintEventArgs e )
  { Graphics g = e.Graphics;
    Rectangle cr = ClientRectangle;
    Random r = new Random();
    Byte red = (Byte)r.Next( Byte.MaxValue );
    Byte green = (Byte)r.Next( Byte.MaxValue );
    Byte blue = (Byte)r.Next( Byte.MaxValue );
    Brush rbrush = new SolidBrush( Color.FromArgb( red, green, blue ) );
    g.FillEllipse( rbrush, cr.Width/5, cr.Height/5, (3*cr.Width)/5, (3*cr.Height)/5 );
    g.DrawEllipse( blackpen, cr.Width/5, cr.Height/5, (3*cr.Width)/5, (3*cr.Height)/5 );
  }
}
```