FINAL PROJECT

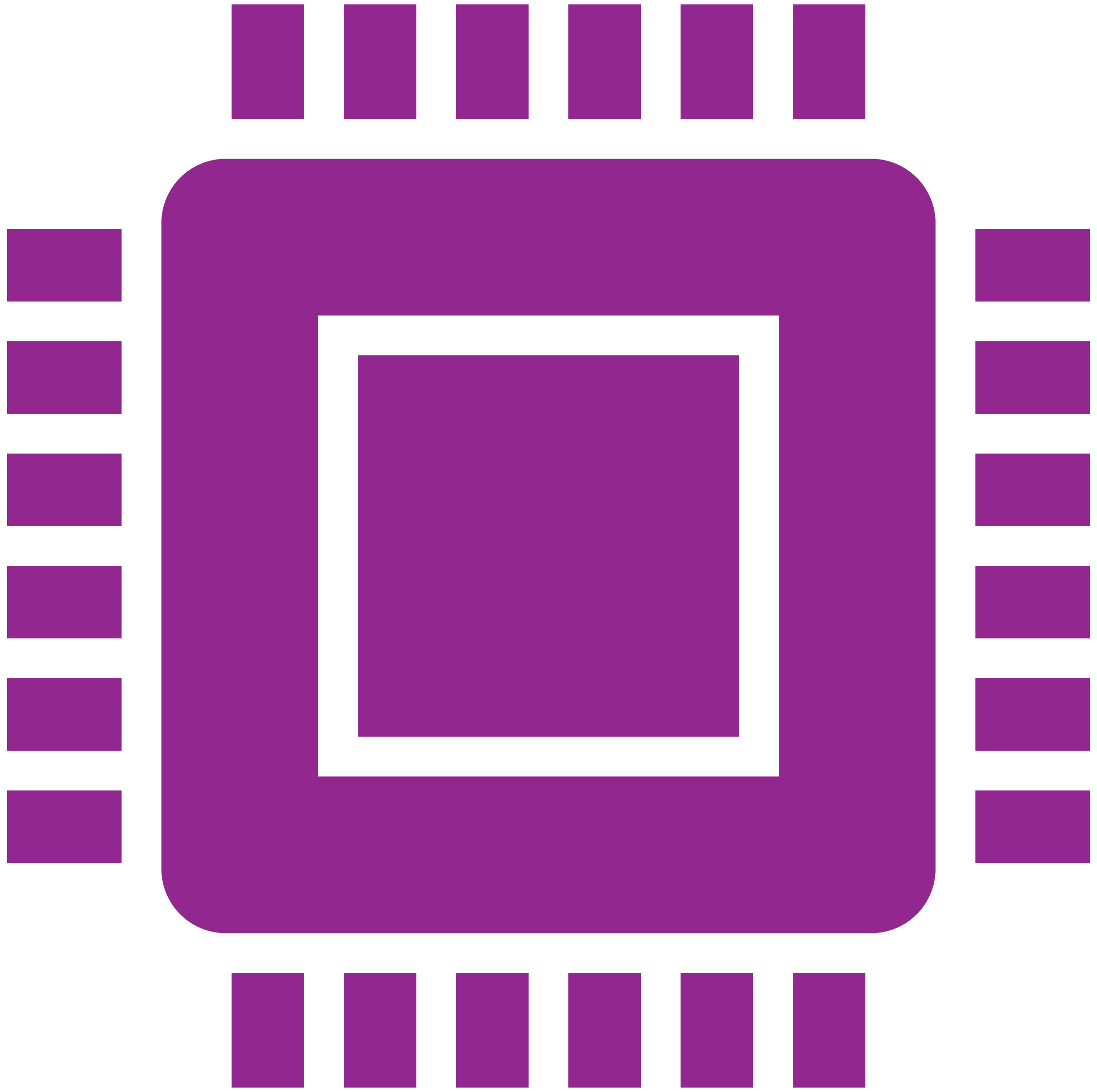
GUI, Events, SVG, Patterns

Final Project

- Create a drawing program
- Use the C# Windows Forms library
- Export SVG files.
- User can use mouse and keyboard
- Provide a visual representation of the document
- Submit as a link to a Github repository

Brief Review of SVG

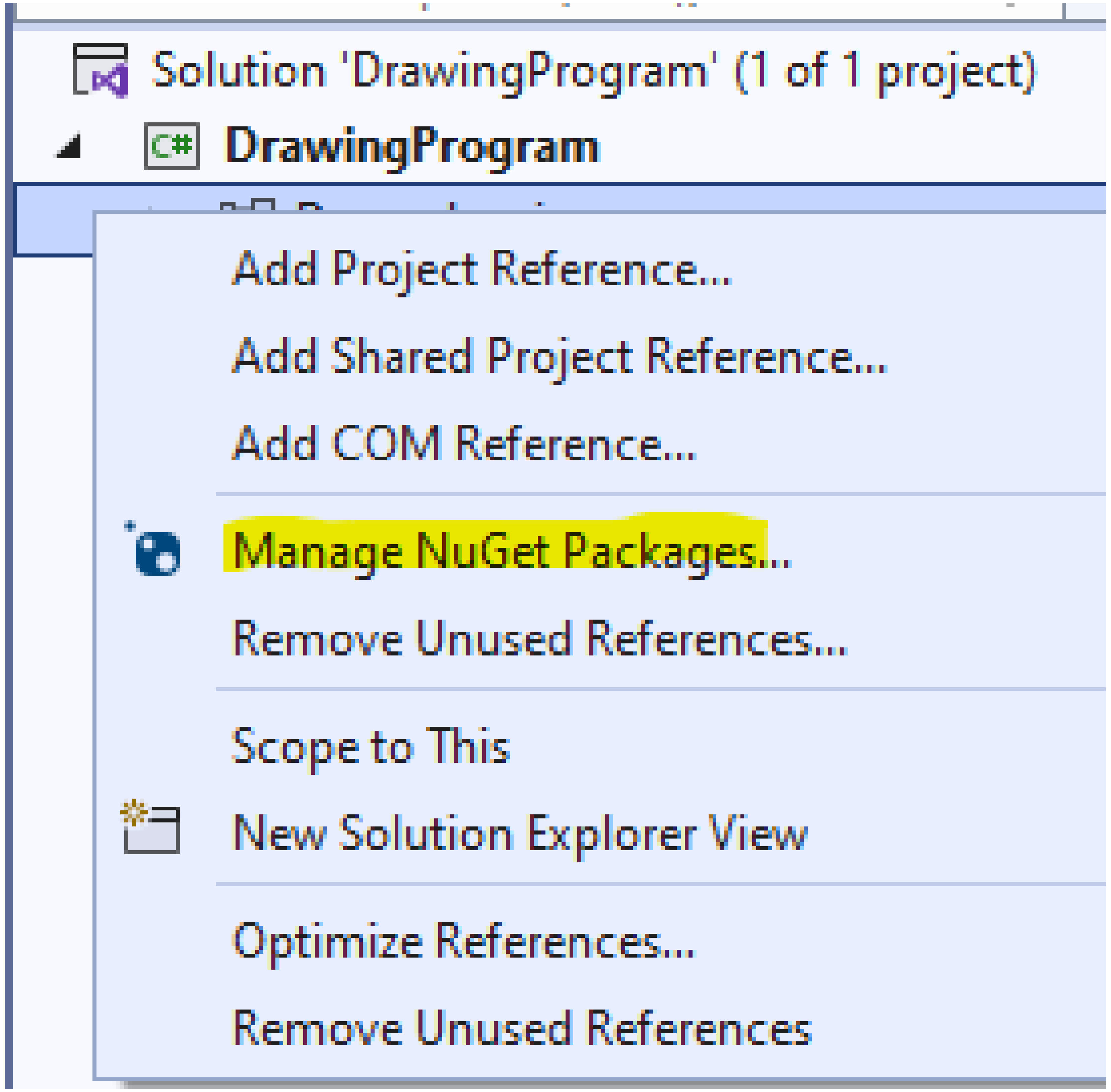
- An XML based format for describing vector graphics
- Stands for “Scalable Vector Graphics”
- https://www.w3schools.com/graphics/svg_intro.asp
- https://developer.mozilla.org/en-US/docs/Web/SVG



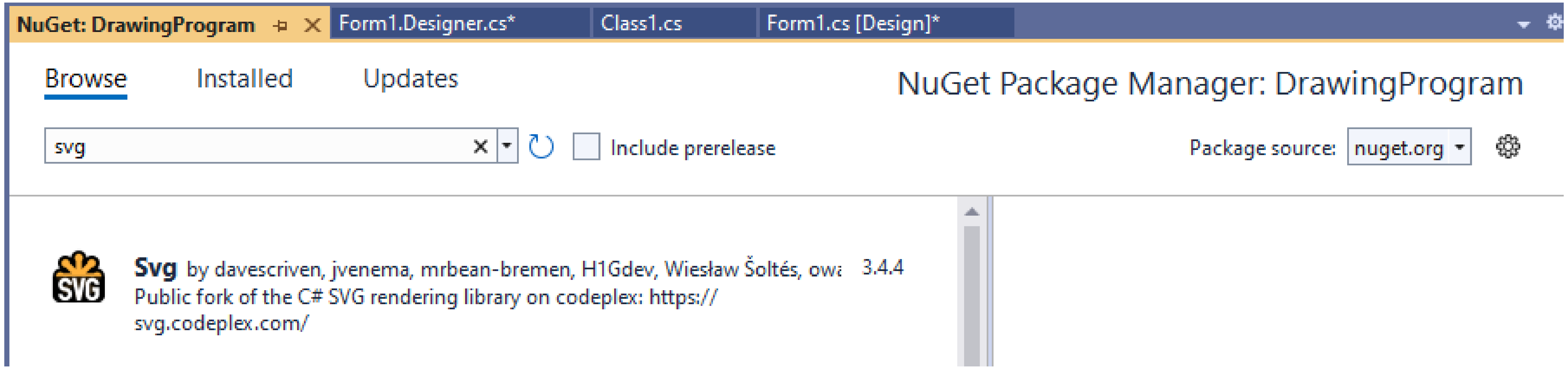
Covered Today

- Learn to create a GUI using Windows Forms
- Event handling
- Design patterns
- Installing 3rd party libraries via Nuget
- Respond to mouse and keyboard events

ADDING NUGET PACKAGE



A Sample Nuget Package



NuGet Package Manager: DrawingProgram

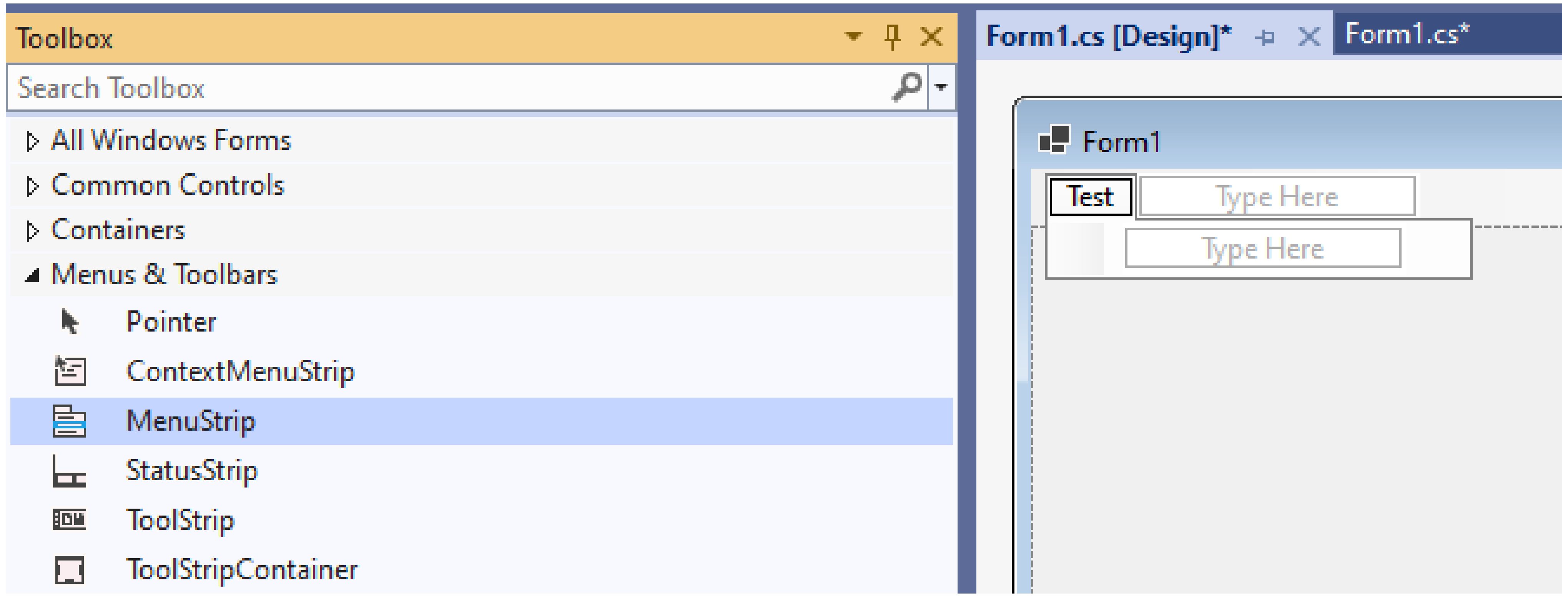
svg

□ Include prerelease

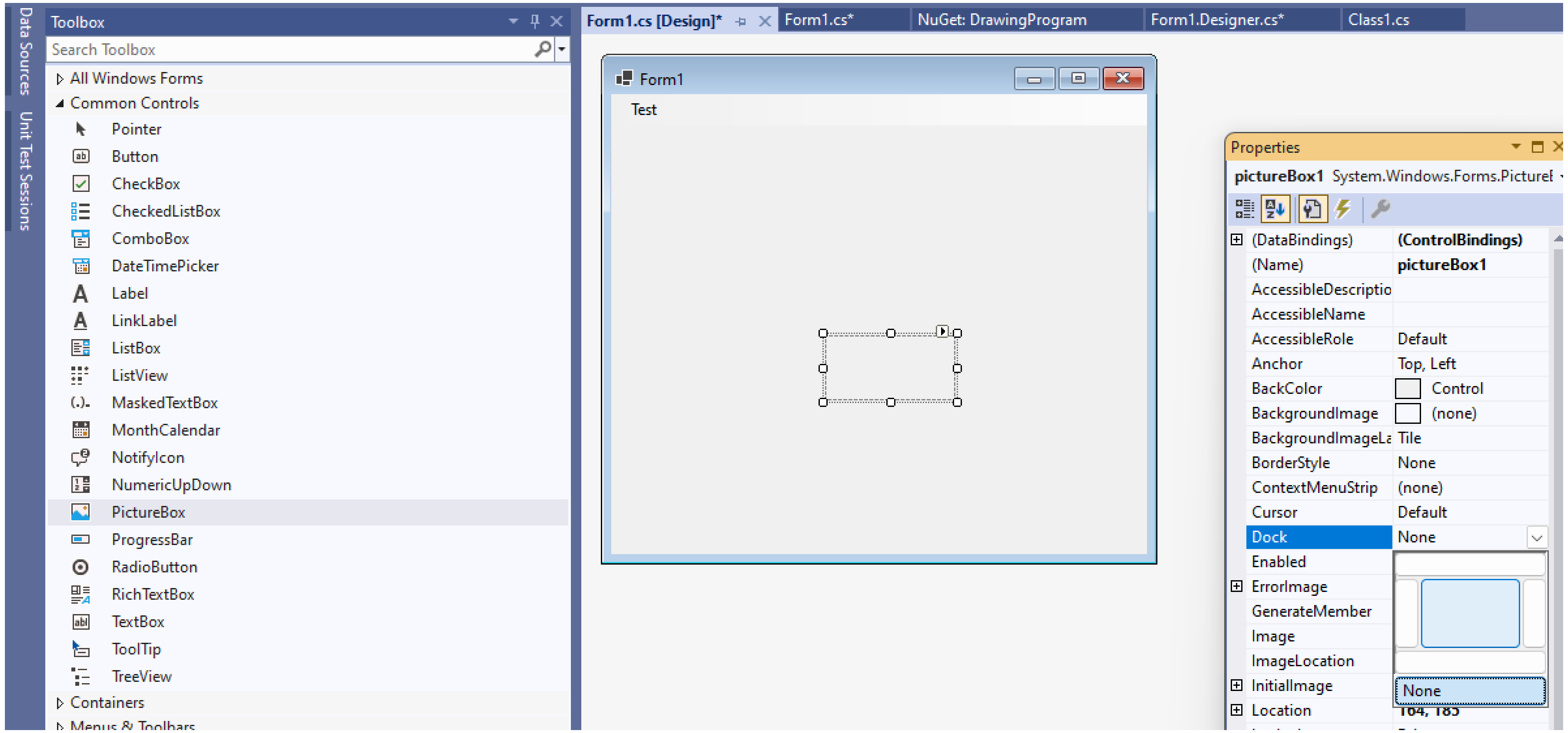
Package source: nuget.org

Svg by davescriven, jvenema, mrbean-bremen, H1Gdev, Wiesław Šoltés, owa 3.4.4
Public fork of the C# SVG rendering library on codeplex: https://svg.codeplex.com/

Add a Menu to the Window



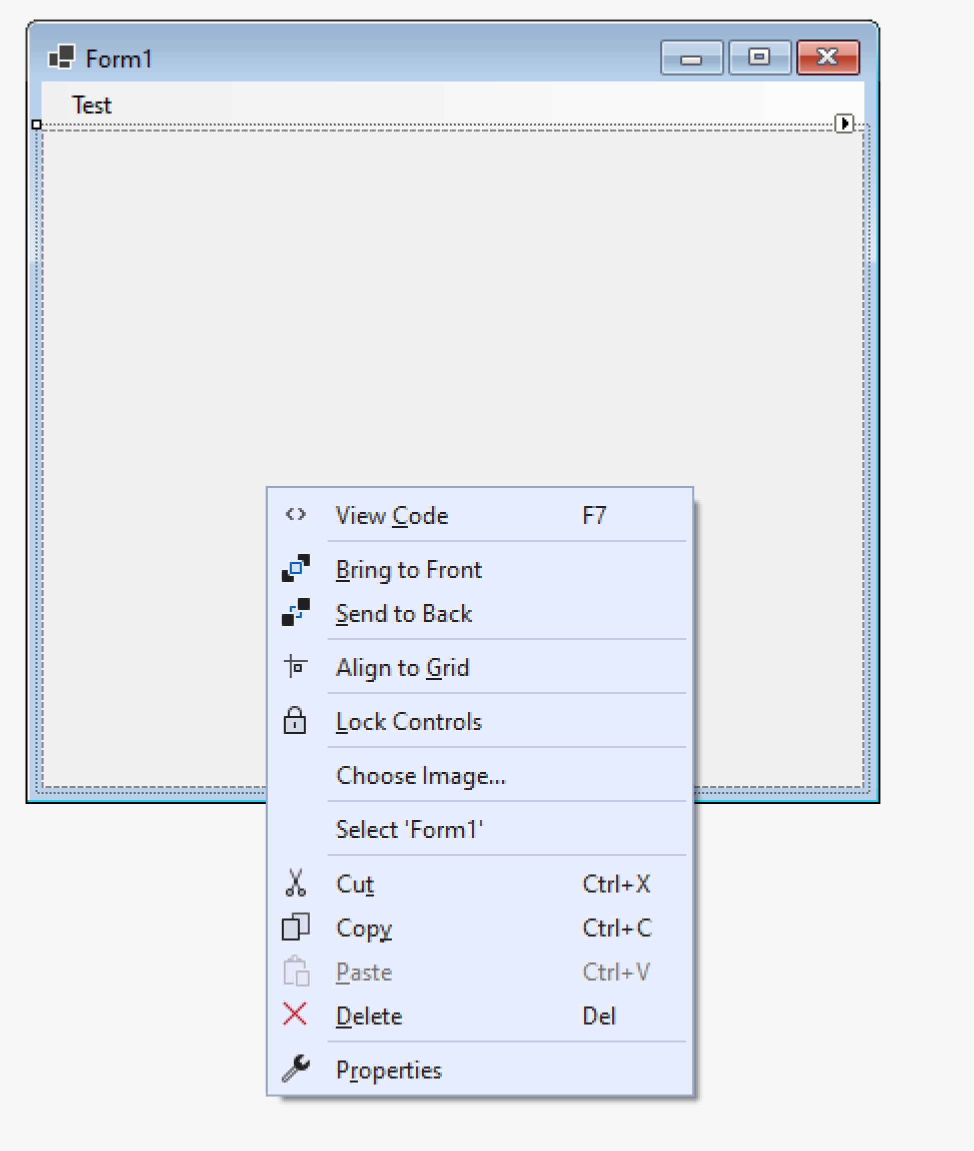
Add a Picture Box
(and set the dock property)



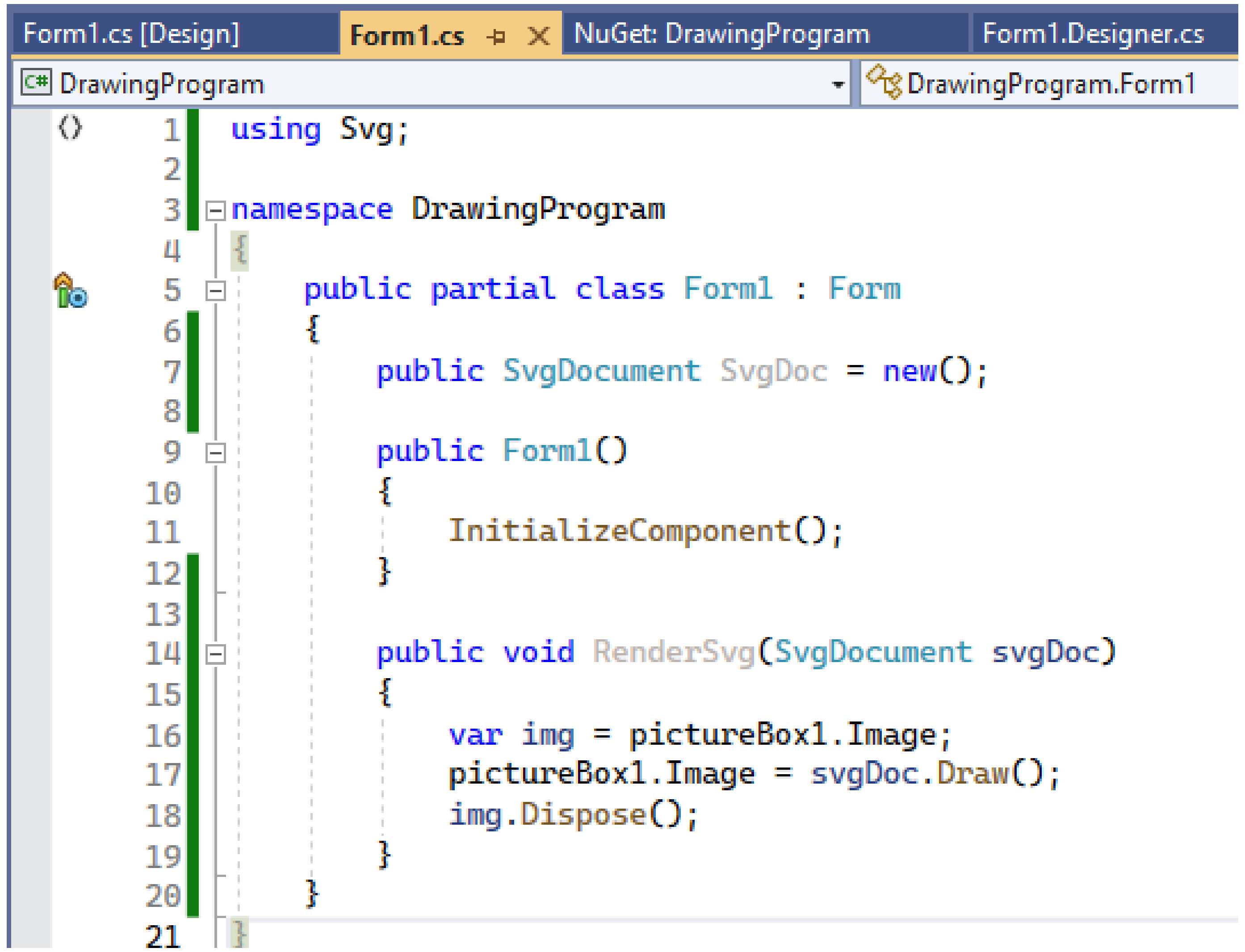
pictureBox1 System.Windows.Forms.PictureB

(DataBindings)	(ControlBindings)
(Name)	pictureBox1
AccessibleDescriptio	
AccessibleName	
AccessibleRole	Default
Anchor	Top, Left
BackColor	Control
BackgroundImage	(none)
BackgroundImageLa	Tile
BorderStyle	None
ContextMenuStrip	(none)
Cursor	Default
Dock	None
Enabled	
ErrorImage	
GenerateMember	
Image	
ImageLocation	
InitialImage	None
Location	164, 185

Switch to the Code Behind View

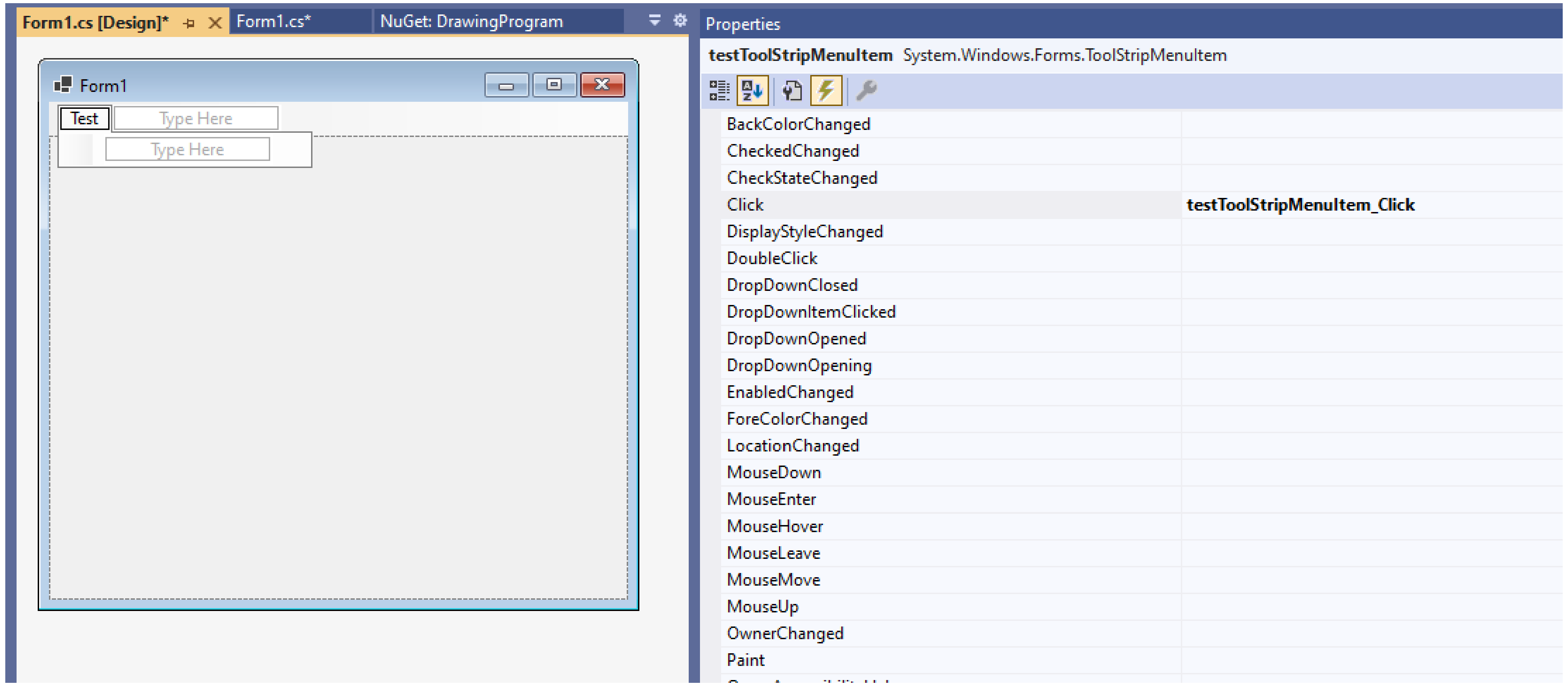


Write Some Code



```
using Svg;

namespace DrawingProgram
{
    public partial class Form1 : Form
    {
        public SvgDocument SvgDoc = new();

        public Form1()
        {
            InitializeComponent();
        }

        public void RenderSvg(SvgDocument svgDoc)
        {
            var img = pictureBox1.Image;
            pictureBox1.Image = svgDoc.Draw();
            img.Dispose();
        }
    }
}
```

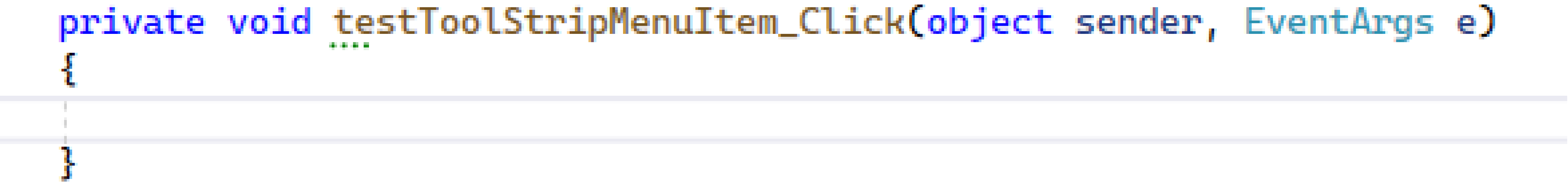
Add an Event for Menu Item



testToolStripMenuItem System.Windows.Forms.ToolStripMenuItem

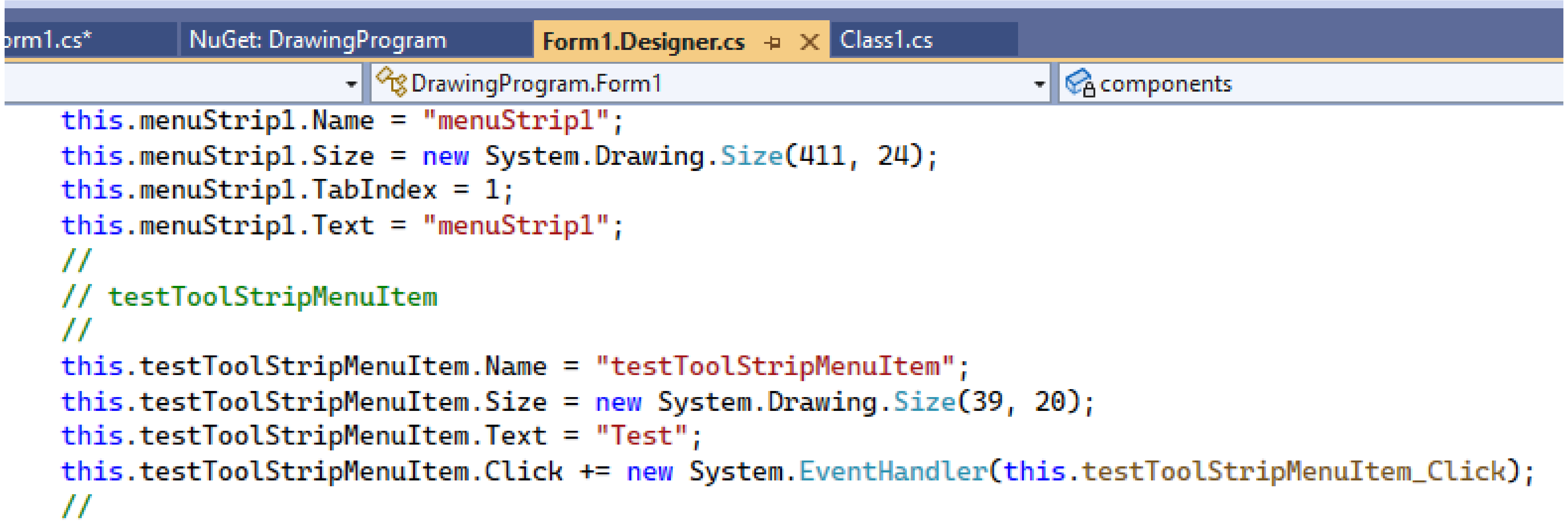
BackColorChanged	
CheckedChanged	
CheckStateChanged	
Click	**testToolStripMenuItem_Click**
DisplayStyleChanged	
DoubleClick	
DropDownClosed	
DropDownItemClicked	
DropDownOpened	
DropDownOpening	
EnabledChanged	
ForeColorChanged	
LocationChanged	
MouseDown	
MouseEnter	
MouseHover	
MouseLeave	
MouseMove	
MouseUp	
OwnerChanged	
Paint	

An Event Handler is Generated



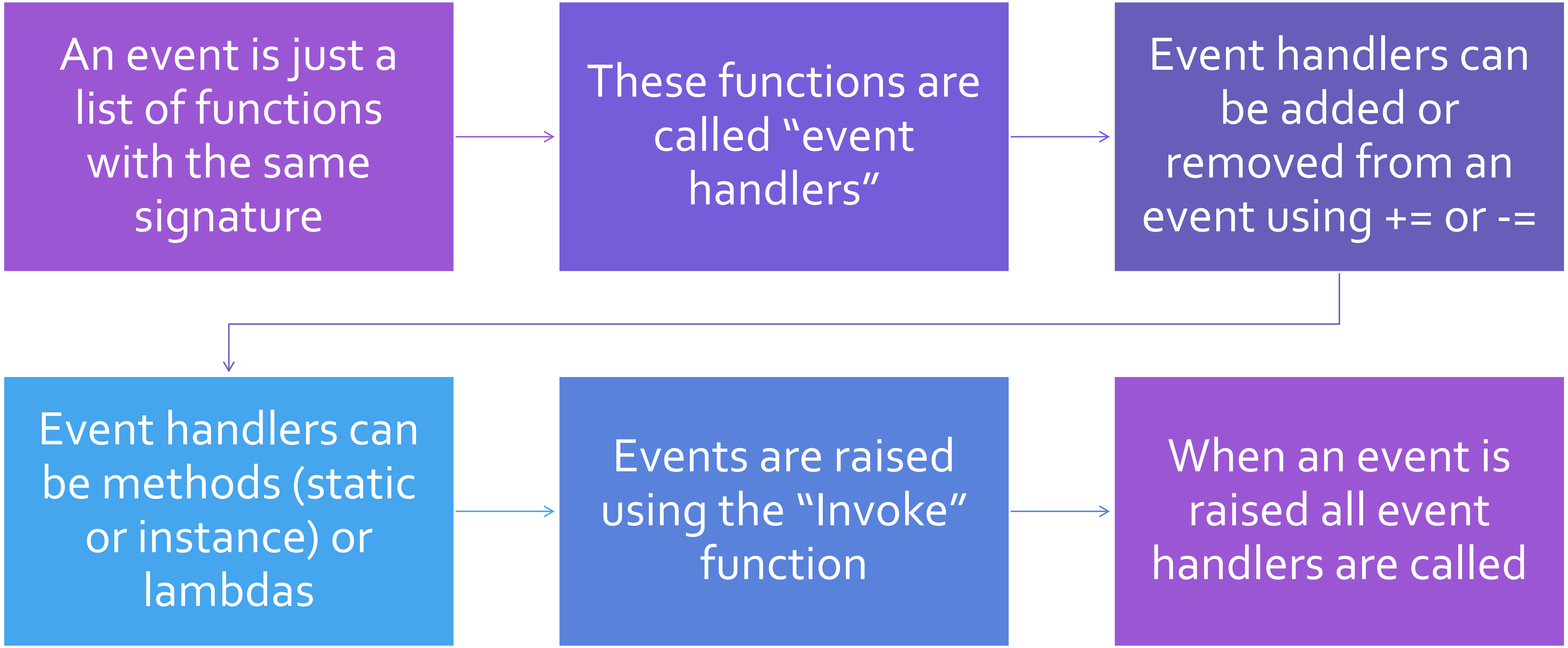
```
private void testToolStripMenuItem_Click(object sender, EventArgs e)
{

}
```

Behind the Scenes
Code is add to the “Designer.cs” file



```
this.menuStrip1.Name = "menuStrip1";
this.menuStrip1.Size = new System.Drawing.Size(411, 24);
this.menuStrip1.TabIndex = 1;
this.menuStrip1.Text = "menuStrip1";
//
// testToolStripMenuItem
//
this.testToolStripMenuItem.Name = "testToolStripMenuItem";
this.testToolStripMenuItem.Size = new System.Drawing.Size(39, 20);
this.testToolStripMenuItem.Text = "Test";
this.testToolStripMenuItem.Click += new System.EventHandler(this.testToolStripMenuItem_Click);
//
```

How Events Work:



An event is just a list of functions with the same signature

These functions are called “event handlers”

Event handlers can be added or removed from an event using += or -=

Event handlers can be methods (static or instance) or lambdas

Events are raised using the “Invoke” function

When an event is raised all event handlers are called

Recommended Reading

- https://learn.microsoft.com/en-us/dotnet/standard/events/
- https://learn.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/event
- https://learn.microsoft.com/en-us/dotnet/csharp/programming-guide/delegates/

What the Heck are Delegates

A delegate is a type that refers to a function as data

Instances of delegates might be methods or lambdas

They are like Func<...> types but named

Observer Pattern

- Events are an example of the observer pattern
- In the observer software design pattern – an object, called the subject, maintains a list of its dependents, called observers, and notifies them automatically of any state changes.
- https://en.wikipedia.org/wiki/Observer_pattern

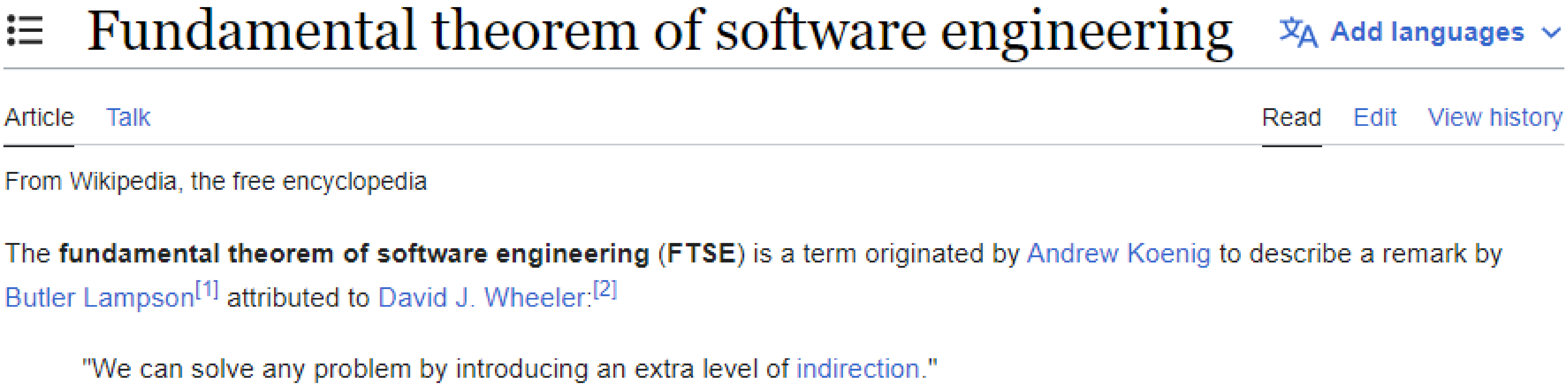
What is a "Software Design Pattern"

Design Patterns
Elements of Reusable Object-Oriented Software
Erich Gamma
Richard Helm
Ralph Johnson
John Vlissides
Foreword by Grady Booch
ADDISON-WESLEY PROFESSIONAL COMPUTING SERIES

- In theory it is a general, reusable solution to a commonly occurring problem within a given context in software design.
- However, much of the industry uses the term to refer to what remains a largely unchanged set of object-oriented patterns that were first published in 1993

Other Patterns for GUI Applications

- Data Binding (MVVM)
- Use a message queue (aka Publisher Subscriber Pattern)
- Model View Update (MVU Pattern)
- Reactive Programming



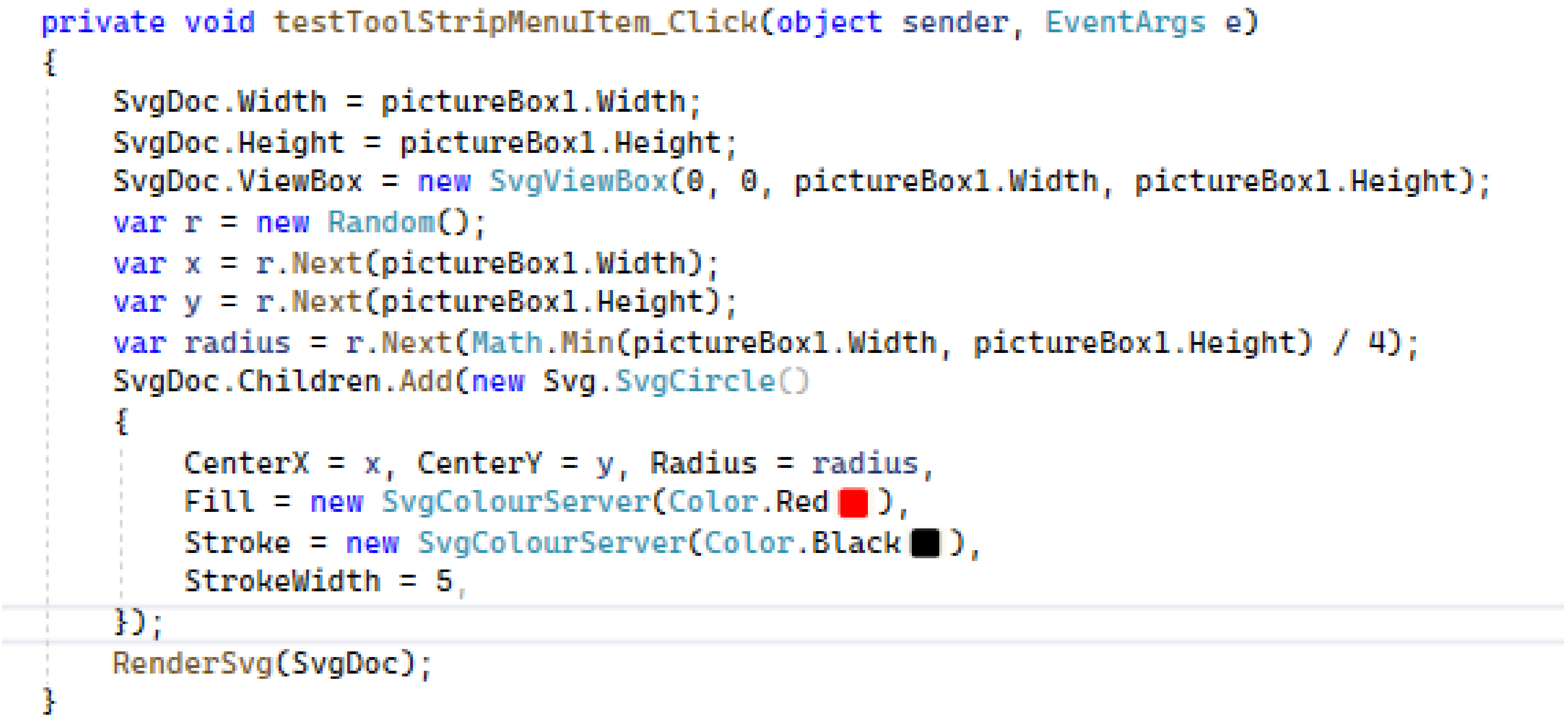
Fundamental theorem of software engineering

From Wikipedia, the free encyclopedia

The **fundamental theorem of software engineering** (**FTSE**) is a term originated by Andrew Koenig to describe a remark by Butler Lampson[1] attributed to David J. Wheeler:[2]

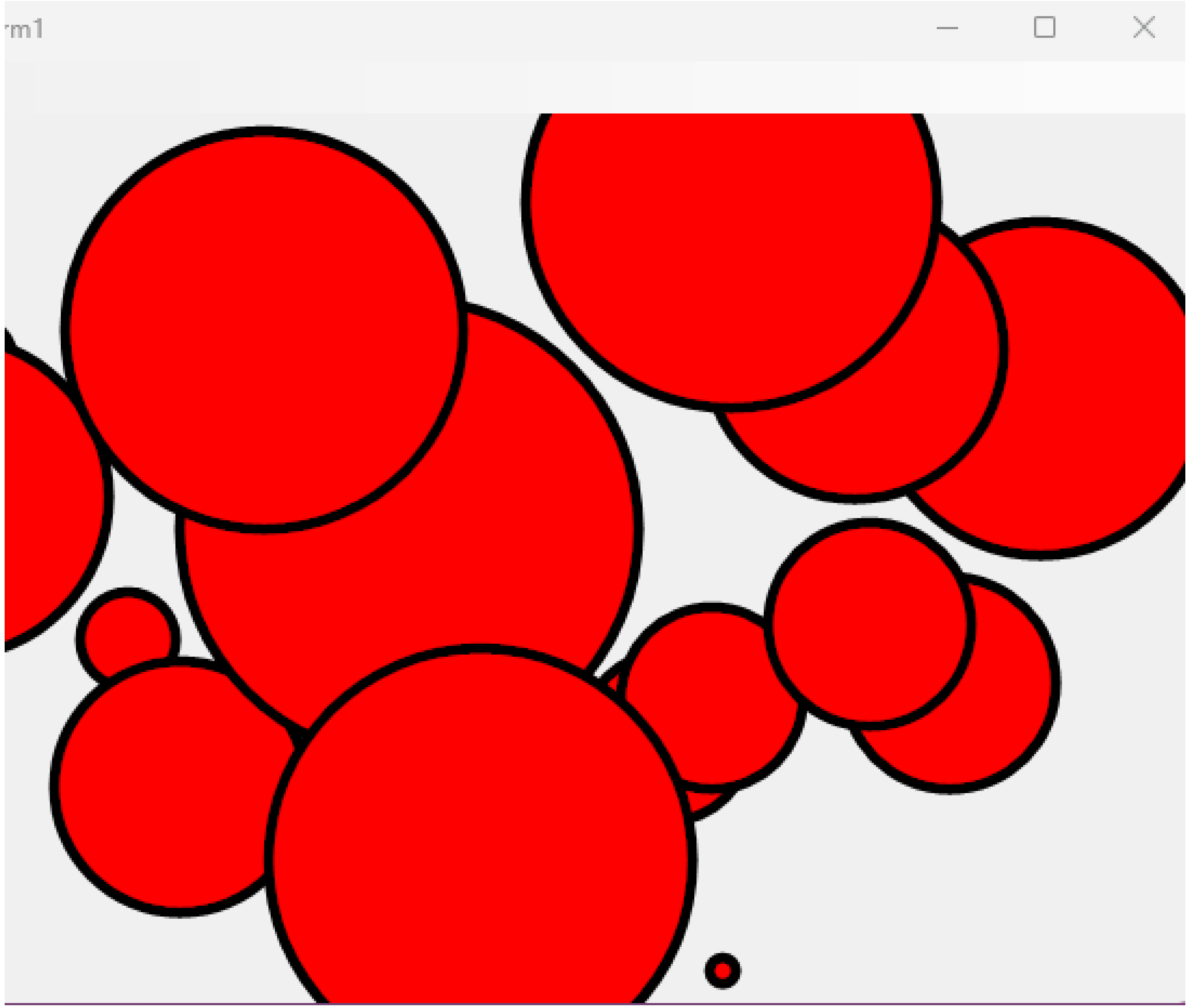
> "We can solve any problem by introducing an extra level of indirection."

BACK TO OUR PROGRAM



```
private void testToolStripMenuItem_Click(object sender, EventArgs e)
{
    SvgDoc.Width = pictureBox1.Width;
    SvgDoc.Height = pictureBox1.Height;
    SvgDoc.ViewBox = new SvgViewBox(0, 0, pictureBox1.Width, pictureBox1.Height);
    var r = new Random();
    var x = r.Next(pictureBox1.Width);
    var y = r.Next(pictureBox1.Height);
    var radius = r.Next(Math.Min(pictureBox1.Width, pictureBox1.Height) / 4);
    SvgDoc.Children.Add(new Svg.SvgCircle()
    {
        CenterX = x, CenterY = y, Radius = radius,
        Fill = new SvgColourServer(Color.Red🟥),
        Stroke = new SvgColourServer(Color.Black⬛),
        StrokeWidth = 5,
    });
    RenderSvg(SvgDoc);
}
```

ADDING SOME CODE



IT BEGINS!

DISCUSS FEATURE IDEAS

Recommended Exercise for Lab

1. Try implementing a basic drawing feature.

2. Click on the mouse once to start drawing a rectangle

3. Move mouse: update rectangle

4. Click again to commit drawing (moving no longer does anything)

5. Press escape to cancel

Click and Move to Draw

- The simplest approach
- Add variables to form: “bool IsDragging”, “Vector2 MouseStart”
- Add handlers for “MouseClick”: start drawing, complete drawing
- “KeyPressed” look for escape key to cancel drawing

Coordinate Spaces: Tricky!

EVERY TIME WE GET COORDINATES

IT IS RELATIVE TO A FRAME OF REFERENCE

THE WINDOW, CONTROL, OR SCREEN

SVG Coordinates

An SVG Element has Width and Height

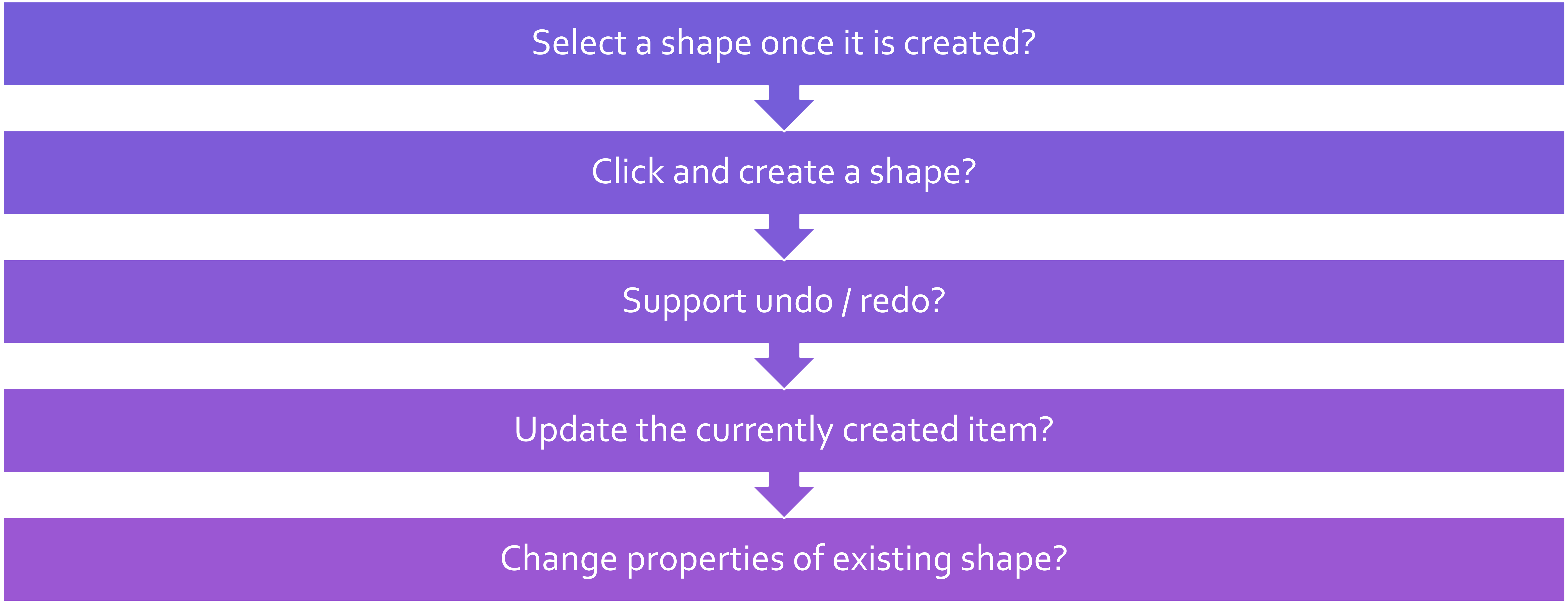
It also has a “Viewbox”

Viewbox provides a virtual coordinate space

SVG .NET Package, Repo and Demos

- Constructing valid SVG documents
- Rendering SVG documents
- Provides sample programs
- To compile and run use: https://github.com/cdiggins/SVG-Forked
- I made it work on .NET 6 only

How do you …



Select a shape once it is created?

Click and create a shape?

Support undo / redo?

Update the currently created item?

Change properties of existing shape?

Command Pattern

A pattern where an object is used to encapsulate information to perform, undo, redo, and log action.

Like a function but with meta-information.

Useful for undo/redo, logging, and scripting.

https://en.wikipedia.org/wiki/Command_pattern