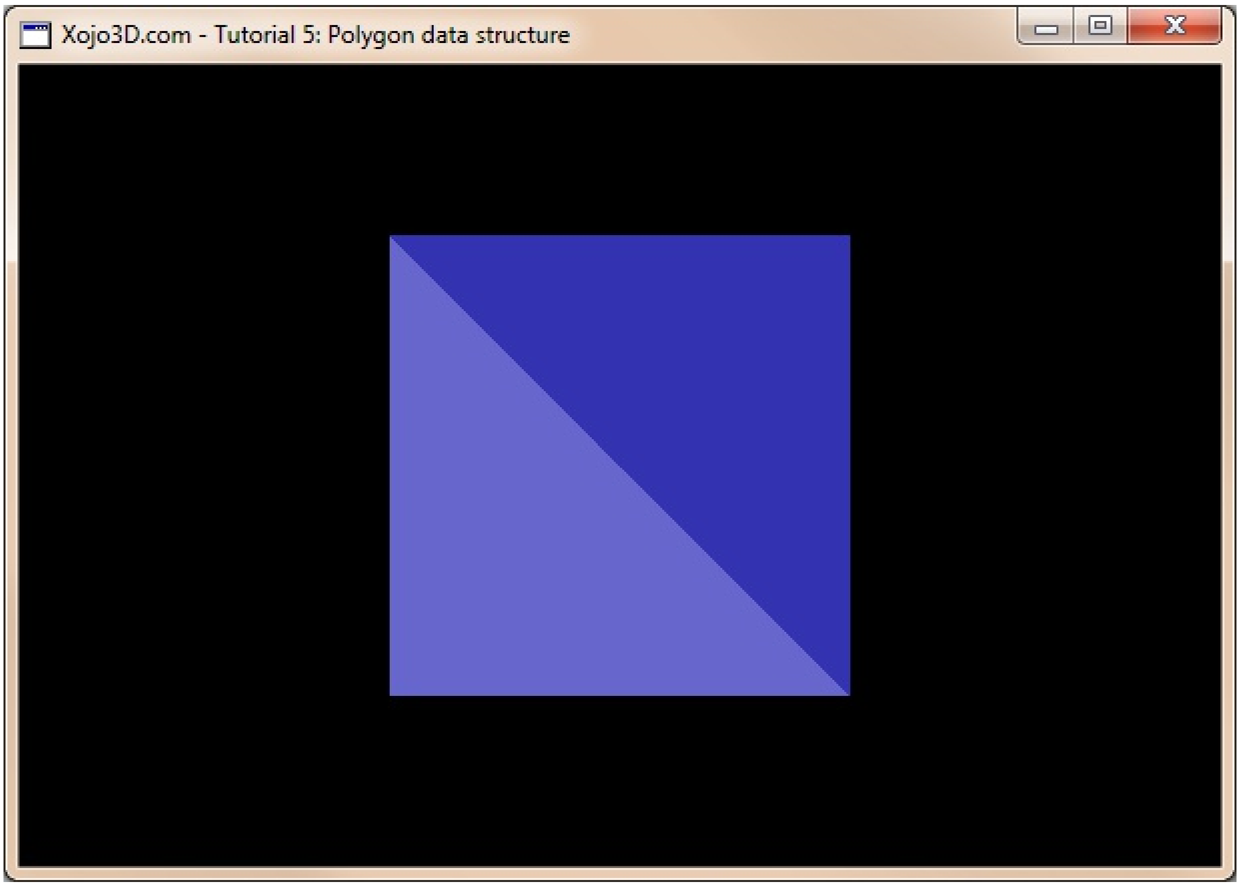
Tutorial 5: Polygon data structure

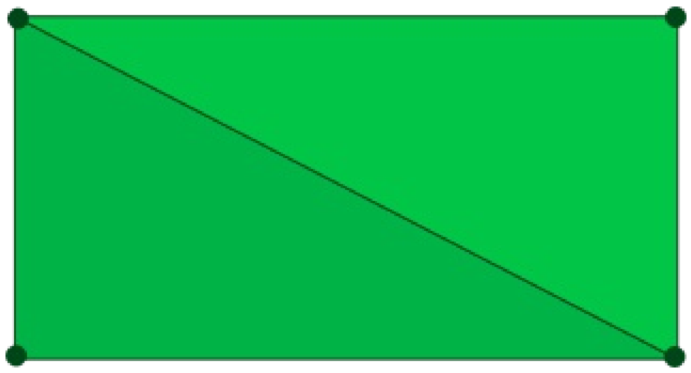
When working with thousands or even millions of polygons, it makes sense to design a polygon structure that simplifies the management of polygons. In this tutorial we design a class to represent polygons.



Theory

A **polygon** is a collection of three or more edges, connected in such a way to form a closed path. The points where the edges of a polygon meet are known as **vertices**.

To simplify 3D algorithms, and optimize for speed, we only use triangular polygons. If we want to create a rectangular shape, we use two triangular polygons as shown below.



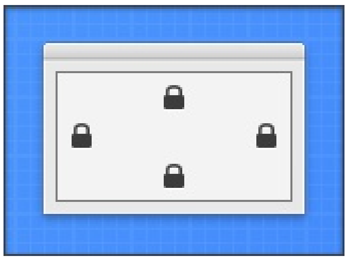
Our polygon data structure is a class that has a vector array to store all the vertices of the polygon and a property to store the color of the polygon.

Tutorial Steps

1. Open Xojo.
2. In the Project Chooser select Desktop.
3. Enter "Tutorial005" as the Application Name, and click OK.
4. Save your project.
5. Configure the following controls:

Control	Name	DoubleBuffer	Left	Top	Maximize Button
Window	SurfaceWindow	-	-	-	ON
OpenGLSurface	Surface	ON	0	0	-

6. Position and size *Surface* to fill the whole window, and set its locking to left, top, bottom and right.



7. Add the following code to the *SurfaceWindow.Open* event handler:

```
Self.MouseCursor = System.Cursors.StandardPointer
```

8. Add the following code to the *SurfaceWindow.Paint* event handler:

```
Surface.Render
```

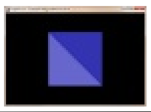
9. Import the X3Core module, created in the previous tutorial.
   You can download the module from http://www.xojo3d.com/tutorials/tut005/x3core.zip.

10. Add the following code to the *Surface.Resized* event handler:

```
X3_SetPerspective Surface
```

11. Add a new class named "X3Color" to module *X3Core*.
12. Add the following properties to *X3Color*:

Name	Type
Red	Double
Green	Double
Blue	Double



13. Add the following method to class *X3Color*:

```
Sub Constructor(initRed As Double,
                    initGreen As Double,
                    initBlue As Double)
  Red = initRed
  Green = initGreen
  Blue = initBlue
End Sub
```

14. Add a new class named "X3Polygon" to module *X3Core*.
15. Add the following properties to *X3Polygon*:

Name	Type
FillColor	X3Color
Vertex()	X3Vector

16. Add the following code to the *Surface.Render* event handler:

```
Dim i, j As Integer
Dim polygon() As X3Core.X3Polygon
Dim poly As X3Core.X3Polygon

polygon.Append new X3Core.X3Polygon()
polygon(0).FillColor = new X3Core.X3Color(0.2, 0.2, 0.7)
polygon(0).Vertex.Append new X3Core.X3Vector(-1, 1, 0)
polygon(0).Vertex.Append new X3Core.X3Vector(1, -1, 0)
polygon(0).Vertex.Append new X3Core.X3Vector(1, 1, 0)

polygon.Append new X3Core.X3Polygon()
polygon(1).FillColor = new X3Core.X3Color(0.4, 0.4, 0.8)
polygon(1).Vertex.Append new X3Core.X3Vector(-1, 1, 0)
polygon(1).Vertex.Append new X3Core.X3Vector(-1, -1, 0)
polygon(1).Vertex.Append new X3Core.X3Vector(1, -1, 0)

OpenGL.glPushMatrix

OpenGL.glClearColor(0, 0, 0, 1)
OpenGL.glClear(OpenGL.GL_COLOR_BUFFER_BIT)

OpenGL.glTranslatef 0.0, 0.0, -3.0

// continue on next page
// continued from previous page

OpenGL.glBegin OpenGL.GL_TRIANGLES

for i = 0 to polygon.Ubound

  poly = polygon(i)

  if poly.FillColor <> nil then
    OpenGL.glColor3d(poly.FillColor.Red, poly.FillColor.Green,
poly.FillColor.Blue)
  else
    OpenGL.glColor3d(1, 1, 1)
  end if

  for j = 0 to polygon(i).Vertex.Ubound
    OpenGL.glVertex3d poly.Vertex(j).X, poly.Vertex(j).Y,
poly.Vertex(j).Z
  next j

next i

OpenGL.glEnd

OpenGL.glPopMatrix
```

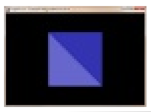
17. Save and run your project.

Analysis

X3Color.Constructor:

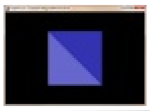
```
Sub Constructor(initRed As Double,
                    initGreen As Double,
                    initBlue As Double)
  Red = initRed
  Green = initGreen
  Blue = initBlue
End Sub
```

The constructor method of X3Color makes it easy to instantiate a new color with predefined values. In the constructor we simply set the Red, Green and Blue properties of the color equal to the given values.



Surface.Render:

```
Dim i, j As Integer
Dim polygon() As X3Core.X3Polygon
Dim poly As X3Core.X3Polygon

poly = new X3Core.X3Polygon()
poly.FillColor = new X3Core.X3Color(0.2, 0.2, 0.7)
poly.Vertex.Append new X3Core.X3Vector(-1, 1, 0) ' top center vertex
poly.Vertex.Append new X3Core.X3Vector(1, -1, 0) ' bottom left vertex
poly.Vertex.Append new X3Core.X3Vector(1, 1, 0) ' bottom right vertex
polygon.Append poly

poly = new X3Core.X3Polygon()
poly.FillColor = new X3Core.X3Color(0.4, 0.4, 0.8)
poly.Vertex.Append new X3Core.X3Vector(-1, 1, 0)
poly.Vertex.Append new X3Core.X3Vector(-1, -1, 0)
poly.Vertex.Append new X3Core.X3Vector(1, -1, 0)
polygon.Append poly

OpenGL.glPushMatrix

OpenGL.glClearColor(0, 0, 0, 1)
OpenGL.glClear(OpenGL.GL_COLOR_BUFFER_BIT)

OpenGL.glTranslatef 0.0, 0.0, -3.0

OpenGL.glBegin OpenGL.GL_TRIANGLES

for i = 0 to polygon.Ubound

  poly = polygon(i)

  if poly.FillColor <> nil then
    OpenGL.glColor3d(poly.FillColor.Red, poly.FillColor.Green,
poly.FillColor.Blue)
  else
    OpenGL.glColor3d(1, 1, 1)
  end if

  // continue on next page
```



```
// continued from previous page

  for j = 0 to polygon(i).Vertex.Ubound
    OpenGL.glVertex3d poly.Vertex(j).X, poly.Vertex(j).Y, 
poly.Vertex(j).Z
  next j

next i

OpenGL.glEnd
```

The polygon() array stores our polygons. The poly object is a temporary object used during the initialization and rendering of the polygons.

Two polygons are created. With each polygon we first instantiate a new X3Polygon object. The FillColor property of each polygon is then set using the constructor method of the X3Color class.

A "for" loop is used to iterate through the polygons stored in our polygon array. During each iteration we use an "if" statement to check if the FillColor of the polygon is set. If the FillColor is not null, we change the current drawing color of OpenGL to the polygon's color, else we set the drawing color to white. A second nested "for" loop is used to iterate through the vertices of the polygon.

We can add as many polygons to the polygon array as we need, without having to change our rendering code.