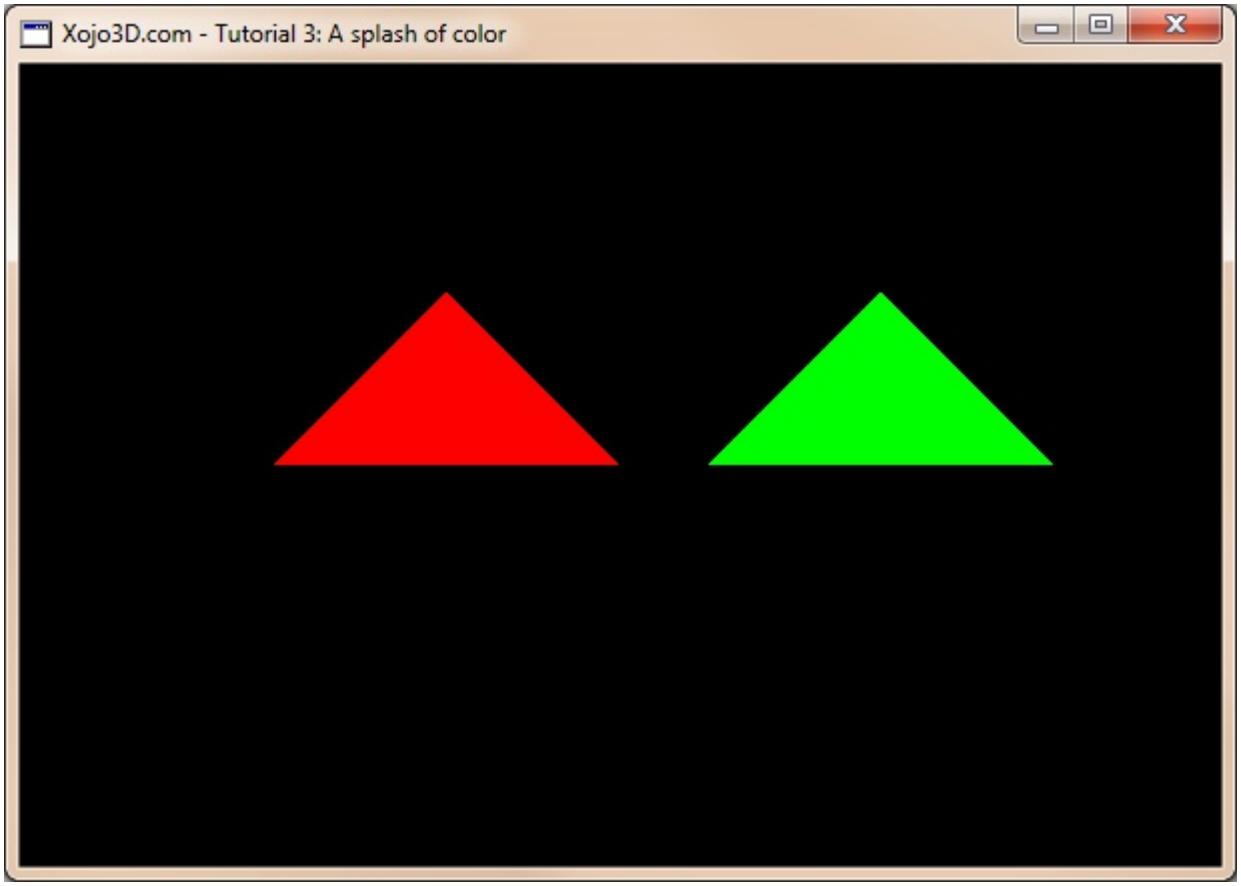
Tutorial 3: A splash of color

Learn how to fill polygons with color.

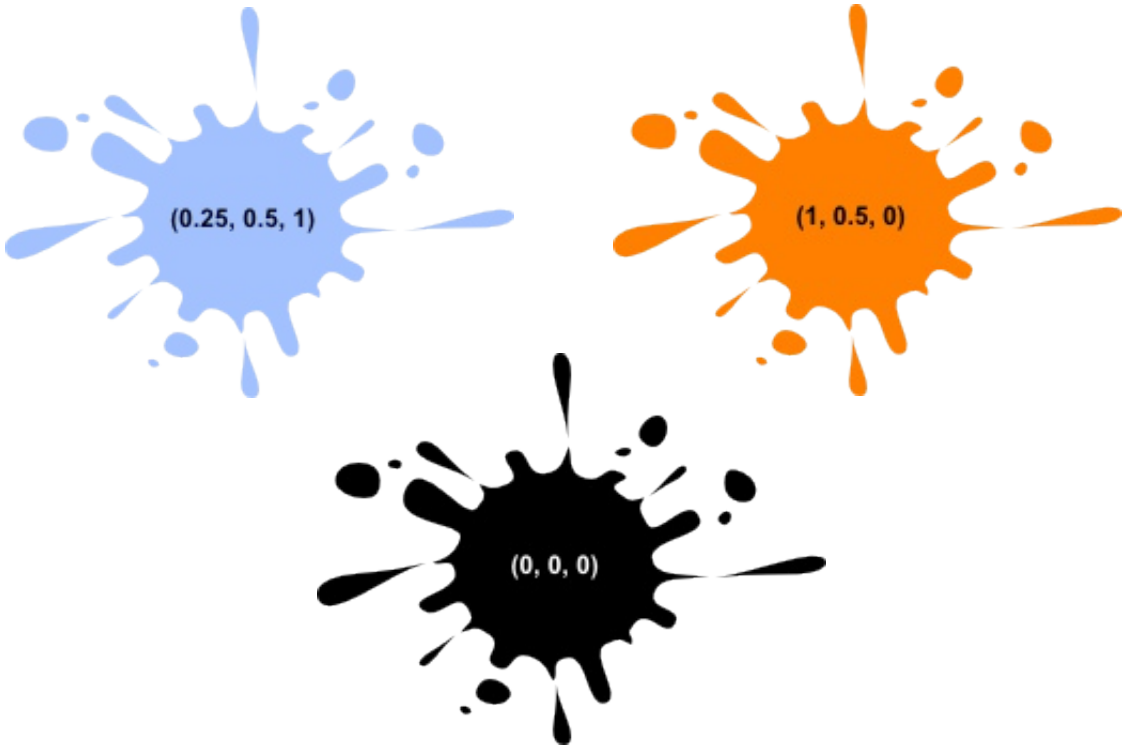


Theory

Different colors are created on a computer by mixing selected quantities of red, green and blue. This color scheme is more commonly known as **RGB**. You can create any imaginable color by mixing these three colors.

Each color component is a real value between 0 and 1. To create a pure red, we use 1 for red and 0 for green and blue. This color is represented as (1,0,0), with the first parameter being red, the second green and the third blue (R,G,B). For a dark green we might use 0.5 for green, and 0 for red and blue (0,0.5,0). White is obtained when all the color components are 1, and black when they are 0.

Below are more examples of some color combinations.



Tutorial Steps

1. Open Xojo.
2. In the Project Chooser select Desktop.
3. Enter "Tutorial003" 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/tut003/x3core.zip.

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

```
X3_SetPerspective Surface
```

11. Add the following code to the Surface.Render event handler:

```
OpenGL.glPushMatrix

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

OpenGL.glTranslatef 0.0, 0.0, -5.0

// continue on next page
```

```
// continued from previous page

OpenGL.glBegin OpenGL.GL_TRIANGLES

OpenGL.glColor3d 1, 0, 0

OpenGL.glVertex3d -1, 1, 1
OpenGL.glVertex3d -2, 0, 1
OpenGL.glVertex3d 0, 0, 1
OpenGL.glColor3d 0, 1, 0

OpenGL.glVertex3d 1.5, 1, 1
OpenGL.glVertex3d 0.5, 0, 1
OpenGL.glVertex3d 2.5, 0, 1
OpenGL.glEnd
```

12. Save and run your project.

Analysis

Surface.Render:

```
OpenGL.glPushMatrix

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

OpenGL.glTranslatef 0.0, 0.0, -5.0

OpenGL.glBegin OpenGL.GL_TRIANGLES

OpenGL.glColor3d 1, 0, 0

OpenGL.glVertex3d -1, 1, 1
OpenGL.glVertex3d -2, 0, 1
OpenGL.glVertex3d 0, 0, 1

OpenGL.glColor3d 0, 1, 0

OpenGL.glVertex3d 1.5, 1, 1
OpenGL.glVertex3d 0.5, 0, 1
OpenGL.glVertex3d 2.5, 0, 1
OpenGL.glEnd

OpenGL.glPopMatrix
```

We use glColor3d to set the red, green and blue components of the drawing color, prior to drawing each polygon. After calling glColor3d, all subsequent vertices are drawn with this color, until the color is changed again with glColor3d.