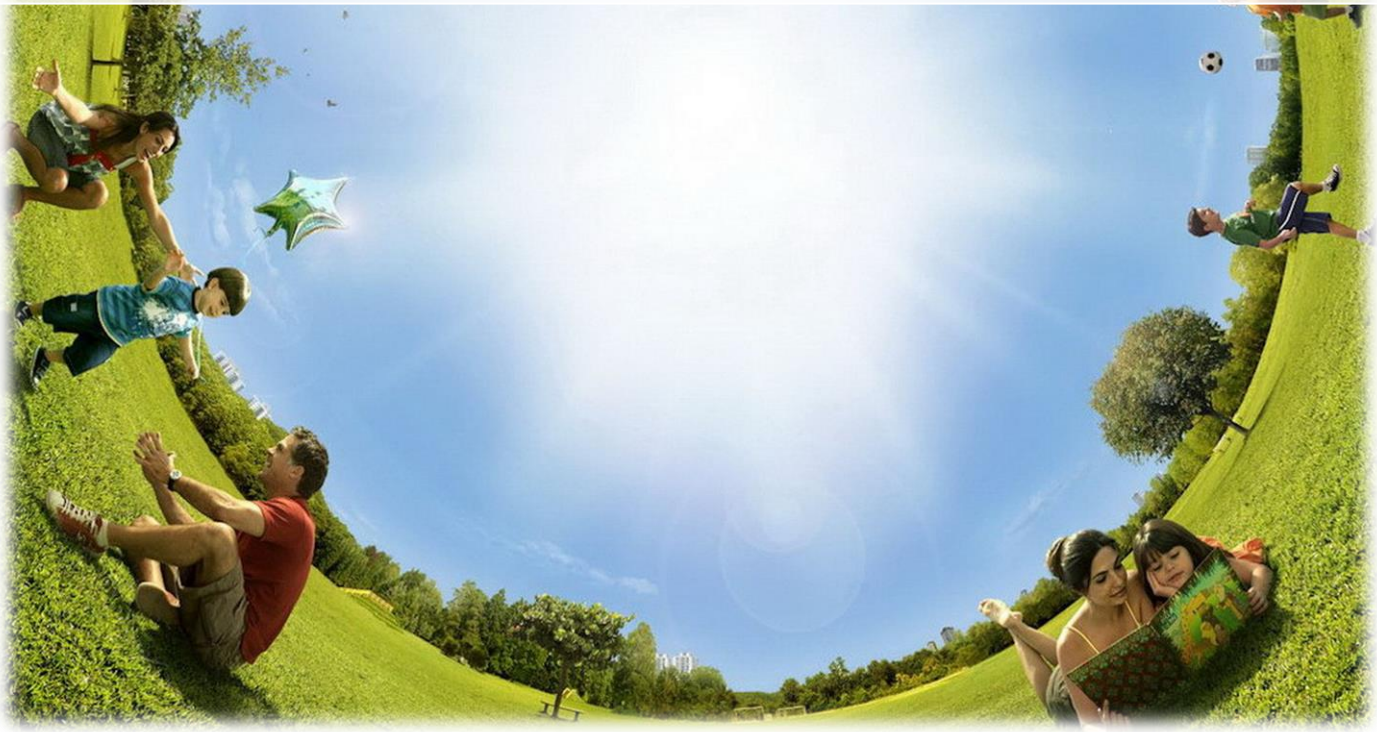
[Mã môn học] – [ĐHMT]

Tháng 10/2021

Giới thiệu OpenGL & SharpGL



Tài liệu này giới thiệu sinh viên làm quen với thư viện lập trình đồ họa OpenGL, hướng dẫn một số thao tác lập trình với giao diện người dùng (GUI) cơ bản và cách sử dụng thư viện SharpGL (C# kết hợp OpenGL).

Bộ môn **[Thị giác máy tính & Khoa học Robot]**
Khoa Công nghệ thông tin
ĐH Khoa học tự nhiên TP HCM

MỤC LỤC

GIỚI THIỆU OPENGL và SHARPGL

1. Tóm tắt lí thuyết

- Giới thiệu về thư viện OpenGL
- Giới thiệu thư viện SharpGL (C# + OpenGL)
- Giới thiệu ứng dụng Windows Forms và ngôn ngữ C#
- Chương trình mẫu và các hàm xử lý chính viết bằng SharpGL và C#

2. Giới thiệu và cài đặt OpenGL và SharpGL

2.1. Giới thiệu OpenGL

- Là thư viện gồm các hàm API cơ bản dùng để biểu diễn đồ họa 2D và 3D, giúp tăng tốc độ xử lý và hiển thị đồ họa. Để phát triển các ứng dụng đồ họa phức tạp, lập trình viên cần phải xử lý nhiều và có thể kết hợp với các thư viện hỗ trợ khác nữa.
- Độc lập ngôn ngữ lập trình (language independent) và sử dụng được trên nhiều hệ thống (multi-flatform).
- Đây là một trong những thư viện được sử dụng rộng rãi nhất trong nhiều ứng dụng: CAD, giả lập thế giới thực, giáo dục, y tế, giải trí, ...

2.2. Giới thiệu SharpGL

- Là thư viện hỗ trợ lập trình bằng ngôn ngữ C# và OpenGL trên môi trường .Net (từ phiên bản 4.0 trở lên).
- SharpGL giúp tạo ra các ứng dụng đồ họa có hỗ trợ giao diện người dùng (Graphical User Interface – GUI) dễ dàng và tiện lợi.
- So sánh cú pháp lệnh của SharpGL và OpenGL

OpenGL	SharpGL
Hằng số	

`GL_TRIANGLES`
 	
`GLU_TESS_COMBINE`
	`OpenGL.GL_TRIANGLES`
	
	`OpenGL.GLU_TESS_COMBINE`

Tiếp đầu ngữ lệnh	

`glLineWidth(3.0f);`
 	
`glPointSize(2.0f);`
	`//  Get a reference to the OpenGL object.`
	`OpenGL gl = openGLCtrl1.OpenGL;`
	
	`//  Set the line width and point size.`
	`gl.LineWidth(3.0f);`
	`gl.PointSize(2.0f);`

Tiếp vĩ ngữ lệnh	

`//  Set the color.`
`glColor3f(0.5f, 0.5f, 0.5f);`
 	
`//  Output some vertices.`
`glVertex3f(2.0f, 3.0f 4.0f);`
	`//  Set the color.`
	`gl.Color3(0.5f, 0.5f, 0.5f);`
	
	`//  Output some vertices.`
	`gl.Vertex3(2.0f, 3.0f 4.0f);`

`glVertex4f(2.0f, 1.0f, 10.0f, 1.0f);`	`gl.Vertex4(2.0f, 1.0f, 10.0f, 1.0f);`

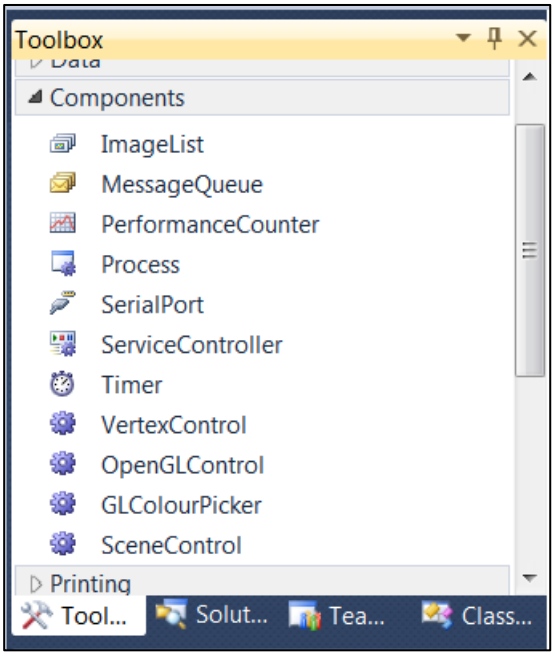
2.3. Hướng dẫn cài đặt và sử dụng

- Vào trang này www.opengl.org/wiki/Getting_Started để cập nhật driver màn hình mới nhất tương ứng với từng hệ điều hành. Sau khi cập nhật xong, bạn sẽ có được thư viện OpenGL bản mới nhất.
- Vào trang này https://drive.google.com/drive/folders/1p2fqA8hb1jji8Rn9mQjyyyHrH-gL75TJ?usp=sharing để tải về thư viện SharpGL (các file thư viện dll)
- Cách sử dụng:

1. Tạo một Project mới, chọn ngôn ngữ C#

2. Chọn project loại Windows Forms Application hoặc WPF Application. Loại project này không có sẵn OpenGLControl nên phải thêm vào.

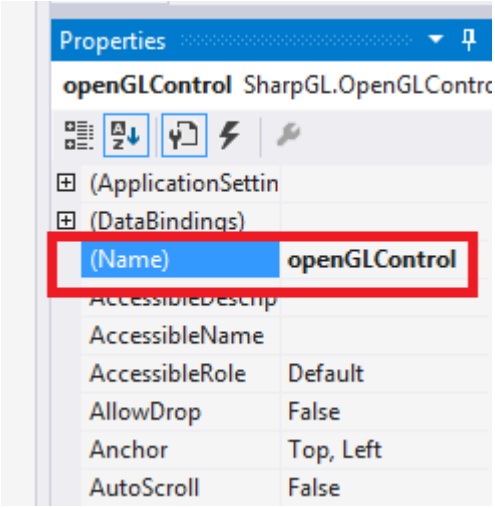
2.1. Kiểm tra thanh Toolbox, thẻ Component xem có OpenGLControl chưa, nếu có rồi thì qua bước 2.4, nếu chưa có thì qua bước tiếp theo.



2.2. Liên kết đến thư viện của SharpGL: trong cửa sổ Solution Explorer → nhấp phải vào project chọn Add, rồi chọn Reference → chọn thẻ Browse → tìm và thêm vào các file SharpGL.dll, SharpGl.WinForms.dll (nếu là Windows Forms) hoặc SharpGL.WPF.dll (nếu là WFP), SharpGLSceneGraph.dll

2.3. Thêm OpenGLControl vào Toolbox: nhấp phải vào thẻ Component trong thanh Toolbox → Choose Item → Browse → tìm và chọn file SharpGl.WinForms.dll (nếu là Windows Forms) hoặc SharpGL.WPF.dll (nếu là WFP)

2.4. Rê chuột kéo OpenGLControl bỏ vào form và sử dụng bình thường. Đặt tên cho OpenGLControl là openGLControl. Click chuột phải lên OpenGLControl, chọn Properties, đổi tên trong mục Name của khung Properties



2.4. Khung chương trình mẫu (hướng dẫn trên Windows Forms, tương tự cho WFP)

- Mở cửa sổ code của form:
  - Nhấp phải Form.cs hoặc SharpGLForm.cs nếu là Windows Forms, chọn View Code
  - Nhấp phải MainWindow.xaml, chọn View Code
- Mỗi chương trình gồm có 3 hàm chính (tham khảo nehe.gamedev.net/tutorial/creating_an_opengl_window_(win32)/13001)

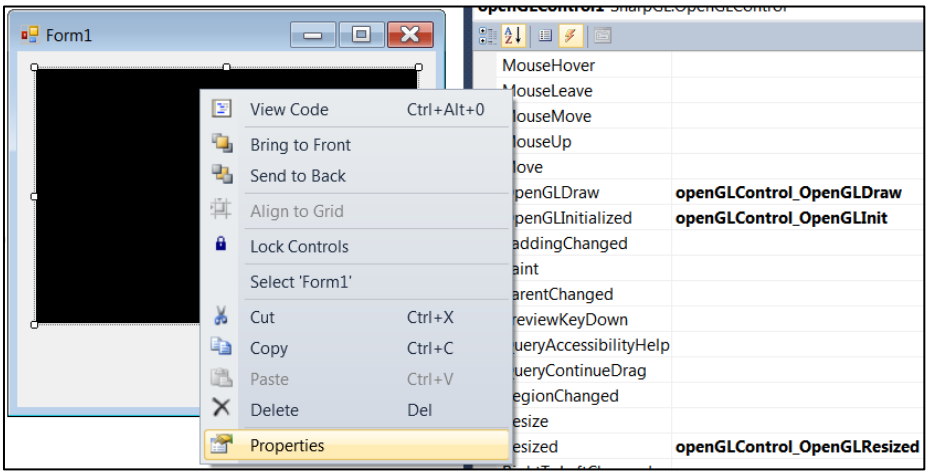
```
private void openGLControl_OpenGLInitialized(object sender, EventArgs e)
{
//  Get the OpenGL object.
            OpenGL gl = openGLControl.OpenGL;

//  Set the clear color.
            gl.ClearColor(0, 0, 0, 0);

//  Set the projection matrix.
            gl.MatrixMode(OpenGL.GL_PROJECTION);
//  Load the identity.
            gl.LoadIdentity();
}
```

```
private void openGLControl_Resized(object sender, EventArgs e)
{
//  Get the OpenGL object.
            OpenGL gl = openGLControl.OpenGL;

//  Set the projection matrix.
            gl.MatrixMode(OpenGL.GL_PROJECTION);
//  Load the identity.
            gl.LoadIdentity();

//  Create a perspective transformation.
            gl.Viewport(0, 0, openGLControl.Width, openGLControl.Height);
            gl.Ortho2D(0, openGLControl.Width, 0, openGLControl.Height);
}
```

```
private void openGLControl_OpenGLDraw(object sender, PaintEventArgs e)
{
//  Get the OpenGL object.
            OpenGL gl = openGLControl.OpenGL;

//  Clear the color and depth buffer.
            gl.Clear(OpenGL.GL_COLOR_BUFFER_BIT | OpenGL.GL_DEPTH_BUFFER_BIT);
```

```
// Vẽ vời chỗ này. Ví dụ:
            gl.Color(1f,0,0,0); // Chọn màu đỏ
            gl.Begin(OpenGL.GL_TRIANGLES);  // Chọn chế độ vẽ tam giác
            gl.Vertex2sv(new short[]{0,0}); // Đỉnh thứ 1 tọa độ 0,0
            gl.Vertex2sv(new short[] { 100, 100 });  // Đỉnh thứ 2 tọa độ 100, 100
            gl.Vertex2sv(new short[]{200, 0});   // Đỉnh thứ 3 tọa độ 200, 0
            gl.End();
            gl.Flush();// Thực hiện lệnh vẽ ngay lập tức thay vì đợi sau 1 khoảng thời gian
}
```

- Cách đăng ký sự kiện:



3. Giới thiệu lập trình giao diện người dùng (GUI) với C#

3.1. So sánh C# và C++

Tham khảo link này http://hyperpolyglot.org/cpp

3.2. Hướng dẫn lập trình với C#

Tham khảo các link sau:

- http://msdn.microsoft.com/en-us/library/67ef8sbd.aspx
- http://msdn.microsoft.com/en-us/library/yyaad03b(v=vs.80).aspx
- http://msdn.microsoft.com/en-us/library/vstudio/dd492135.aspx

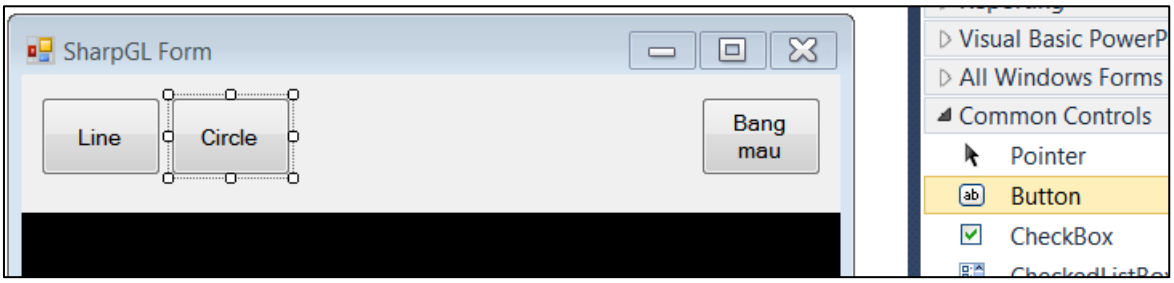
4. Case Study

Viết chương trình vẽ đoạn thẳnng sử dụng SharpGL và hỗ trợ các chức năng sau:

- Cho phép chọn màu từ bảng màu
- Cho phép vẽ bằng cách click chuột vào điểm đầu, giữ và kéo đến điểm cuối

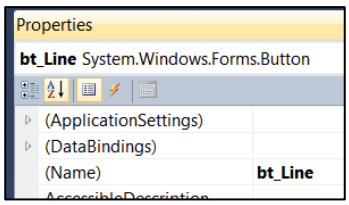
5. Hướng dẫn cụ thể

Bước 1:Điều chỉnh kích thước OpenGLControl và kéo các nút bấm (Button) vào form và sắp xếp lại theo như hình dưới đây



Bước 2: Lần lượt nhấp phải vào từng control, chọn property và cập nhật các giá trị thuộc tính như sau:

- Text: dòng chữ sẽ hiển thị trên giao diện của control. Ví dụ: Draw, X0, Y0, Bảng Màu
- Name: tên biến quản lý control, đặt theo qui ước <loại control>_<Tên control>. Ví dụ
  Nút Line thì Name sẽ là bt_Line
  Nút Bảng Màu thì Name sẽ là bt_BangMau
  ...

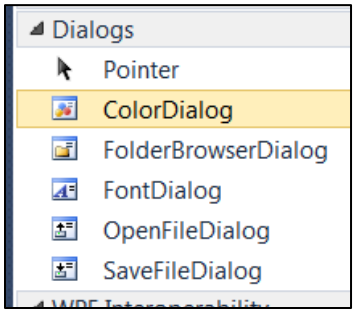


Bước 3: Xử lý sự kiện khi người dùng nhấp vào nút Bảng Màu thì chương trình sẽ hiện ra hộp thoại cho người dùng chọn màu tùy thích và lưu lại

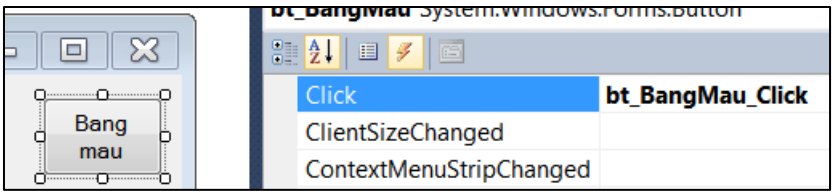
- Tạo 1 biến để lưu màu do người dùng lựa chọn và khởi tạo giá trị ban đầu

```
public partial class SharpGLForm : Form
{
    /// <summary> ...
    Color colorUserColor; //Màu để vẽ hình

    public SharpGLForm()
    {
        InitializeComponent();

        colorUserColor = Color.White; //Gán giá trị màu ban đầu
    }
```

- Chọn ColorDialog từ Toolbox/Dialogs và kéo vào vị trí bất kỳ trong form



- Nhấp phải nút Bang Mau→ Property → nhấp vào nút Events (hình sét đánh)



- Tìm sự kiện Click và đặt tên cho hàm xử lý sự kiện này vào ô kế bên theo qui ước <loạt control>_<Tên control>_<Sự kiện>
- Visual Studio sẽ tự động phát sinh hàm xử lý sự kiện

```
private void bt_BangMau_Click(object sender, EventArgs e)
{
    |
}
```

- Gọi hộp thoại chọn màu và lưu kết quả lựa chọn của người dùng

```
private void bt_BangMau_Click(object sender, EventArgs e)
{
    //Nếu ngdùng chọn xong và bấm OK
    if (colorDialog1.ShowDialog() == DialogResult.OK)
    {
        colorUserColor = colorDialog1.Color; //Lưu lại màu người dùng đã chọn
    }
}
```

- Cập nhật màu khi người dùng vẽ hình

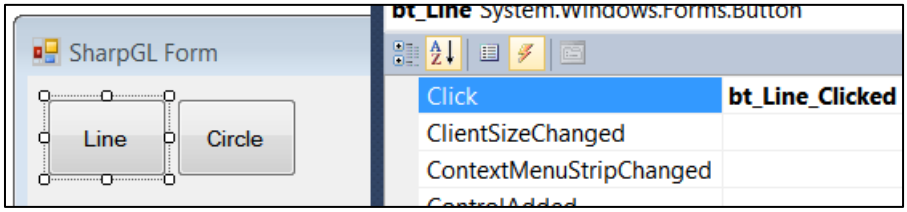
```
private void openGLControl_OpenGLDraw(object sender, PaintEventArgs e)
{
    //  Get the OpenGL object.
    OpenGL gl = openGLControl.OpenGL;

    //  Clear the color and depth buffer.
    gl.Clear(OpenGL.GL_COLOR_BUFFER_BIT | OpenGL.GL_DEPTH_BUFFER_BIT);

    // Chọn màu
    gl.Color(colorUserColor.R / 255.0, colorUserColor.G / 255.0, colorUserColor.B / 255.0, 0);

    // Vẽ vời chỗ này
    // ...|
}
```

Bước 4: Xử lý sự kiện chọn chức năng vẽ đoạn thẳng

- Sử dụng biến để ghi nhớ người dùng muốn vẽ đường thẳng hay đường tròn

```
Color colorUserColor; //Màu để vẽ hình
short shShape; //0 nếu muốn vẽ đường thẳng, 1 nếu đường tròn

public SharpGLForm()
{
    InitializeComponent();

    colorUserColor = Color.White; //Gán giá trị màu ban đầu
    shShape = 0; //Gán giá trị ban đầu
}
```

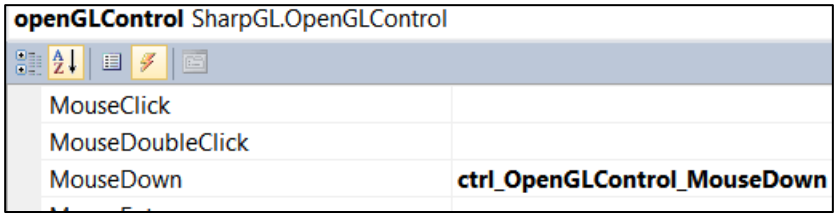
- Tạo sự kiện



- Cập nhật biến mỗi khi người dùng chọn chức năng vẽ đoạn thẳng

```
private void bt_Line_Clicked(object sender, EventArgs e)
{
    shShape = 0;
}
```

- Ghi nhận tọa độ điểm đầu khi người dùng bắt đầu nhấp chuột trái
  - Nhấp vào OpenGLControl và bắt sự kiện



  - Cập nhật lại các biến tọa độ

```
private void form_SharpGLForm_MouseDown(object sender, MouseEventArgs e)
{
    // Cập nhật tọa độ điểm đầu và cuối
    pStart = e.Location; // e là tham số liên quan đến sự kiện này
    pEnd = pStart;
}
```

- Ghi nhận tọa độ điểm cuối khi người dùng kéo chuột đi

```
private void form_SharpGLForm_MouseUp(object sender, MouseEventArgs e)
{
    //Cập nhật tọa độ điểm cuối để vẽ
    pEnd = e.Location;
}
```

- Cập nhật lại hàm OpenGLDraw cho phù hợp

```
private void openGLControl_OpenGLDraw(object sender, PaintEventArgs e)
{
    //  Get the OpenGL object.
    OpenGL gl = openGLControl.OpenGL;

    //  Clear the color and depth buffer.
    gl.Clear(OpenGL.GL_COLOR_BUFFER_BIT | OpenGL.GL_DEPTH_BUFFER_BIT);

    // Chọn màu
    gl.Color(colorUserColor.R / 255.0, colorUserColor.G / 255.0, colorUserColor.B / 255.0, 0);

    //// Vẽ vời chỗ này

    if (shShape == 0) // Nếu muốn vẽ đoạn thẳng
    {
        gl.Begin(OpenGL.GL_LINES);
        //Sinh viên tự giải thích 2 dòng code dưới đây
        gl.Vertex(pStart.X, gl.RenderContextProvider.Height - pStart.Y);
        gl.Vertex(pEnd.X, gl.RenderContextProvider.Height - pEnd.Y);
        gl.End();
        gl.Flush();
    }
    else //if (shShape == 1)
    {
        // Sinh viên tự viết tiếp
    }
}
```

6. Tìm hiểu thêm

Sử dụng hàm thư viện có sẵn

- Vẽ các điểm
- Vẽ đường thẳng
- Vẽ hình tam giác, hình chữ nhật
- Cho phép chọn độ dày của nét vẽ

Tài liệu tham khảo

Wikipedia. *OpenGL*. http://en.wikipedia.org/wiki/OpenGL. 2/2013.