CS 550 | Fall 2023

# Project 6

# “Shaders "

By

**Pushpak Katkhede**

Email – katkhedp@oregonstate.edu

**OUSID – 934453040**

Instructor – Prof Mike Bailey

## ❖ Description →

**Video Link:** **Click Here**

The InitLists function is critical in initializing the display lists used to render graphics in an OpenGL scene. When called, it first checks to see if debugging is enabled (DebugOn), and if so, it prints a message. The current OpenGL rendering window is then set to the main window (MainWindow).

Following that, the function generates two display lists: SphereList and AxesList. Using glGenLists(1), a unique identifier is assigned to the SphereList, and its contents are defined within the list using glNewList. The OsuSphere function is then invoked with specific parameters to generate the sphere's geometry, which is contained within the display list. This method improves rendering efficiency by allowing the compiled sphere geometry to be reused.

The AxesList is similarly created to store the display list for rendering axes. The line width is set to AXES_WIDTH within this list, and the Axes function is called to generate the axes geometry. The glEndList call completes the display list compilation.In summary, InitLists is in charge of configuring display lists for a sphere and axes, as well as optimizing rendering performance by precompiling their geometry. This initialization process improves efficiency in the Display function's subsequent rendering.

**The following lines showing your keytime values for USc and uTc.**

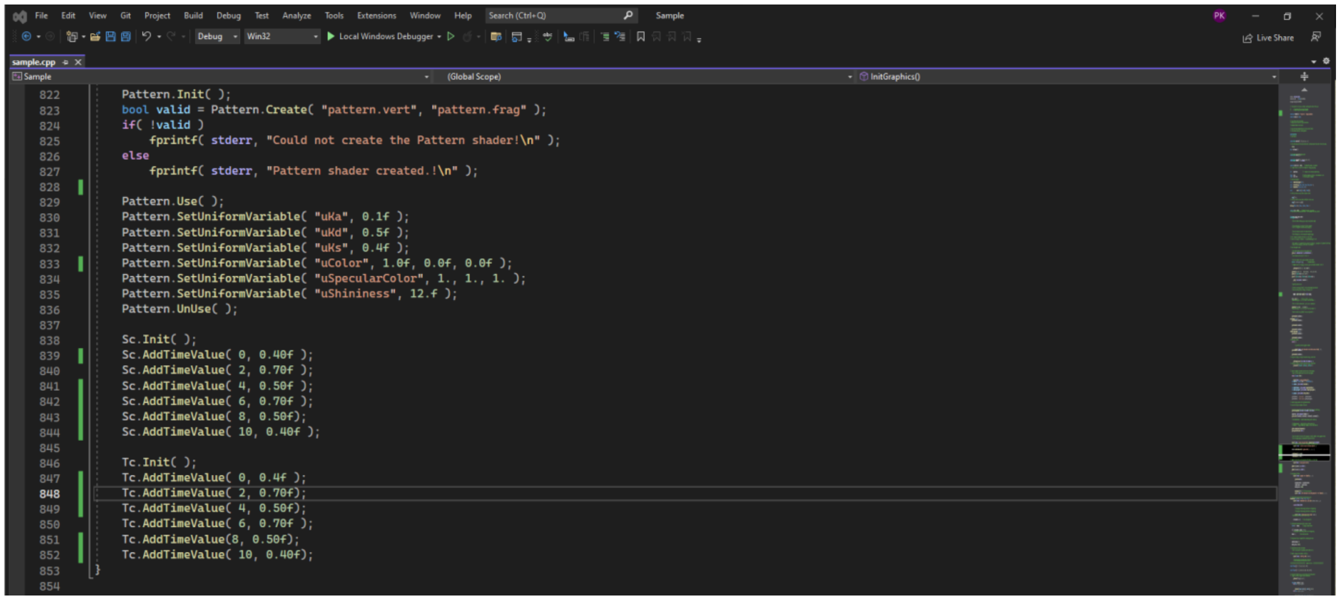
| **Object** | **Time** | **Value** |
|---|---|---|
| Sc | 0 | 0.40 |
| Sc | 2 | 0.70 |
| Sc | 4 | 0.50 |
| Sc | 6 | 0.70 |
| Sc | 8 | 0.50 |
| Sc | 10 | 0.40 |
| Tc | 0 | 0.40 |
| Tc | 2 | 0.70 |
| Tc | 4 | 0.50 |
| Tc | 6 | 0.70 |
| Tc | 8 | 0.50 |
| Tc | 10 | 0.40 |

I initialized two objects, Sc and Tc, in the animation code with time-value pairs ranging from 0 to 10 at 2 intervals. The corresponding values had shown a discernible pattern with varying magnitudes. I needed to see the output to confirm the animation's functionality. The animation dynamically represented the time-value pairs, providing a clear depiction of how the values for Sc and Tc had evolved over time. I had validated whether the programmed behavior aligned with the expected outcome through visual observation, ensuring that the time-value pairs for the objects were accurately portrayed and transitioned in a coherent manner.

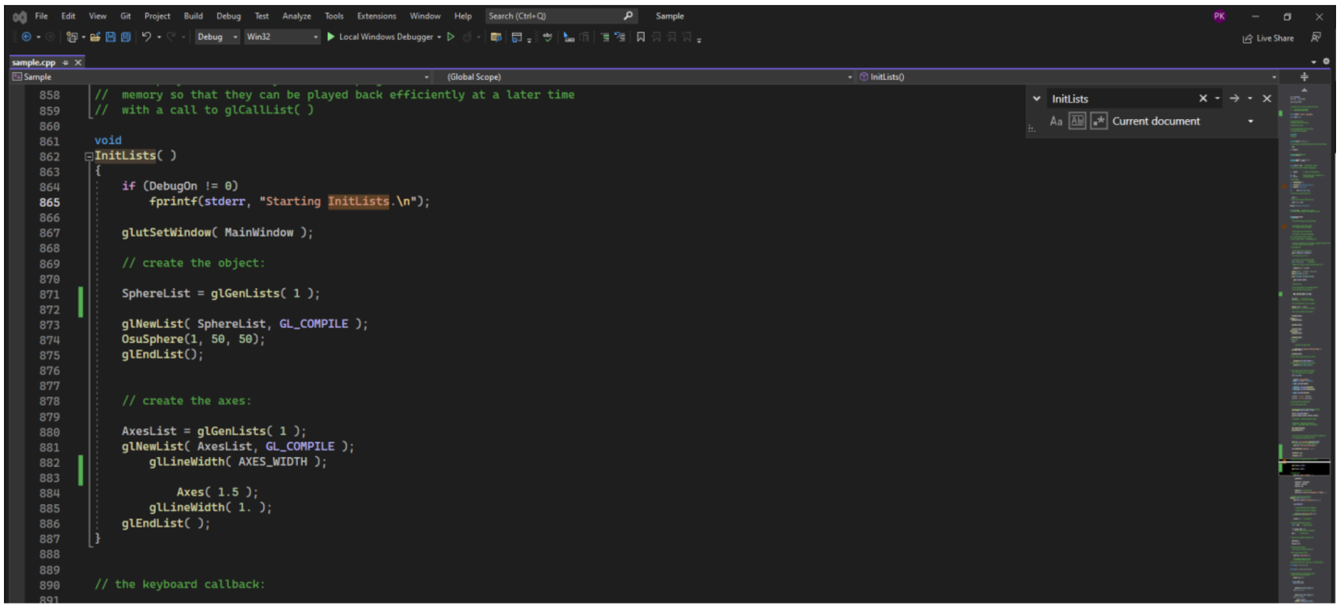
## ❖ Key and Function →

1. 't': Toggle Time based
2. 'k': Toggle Keytime based

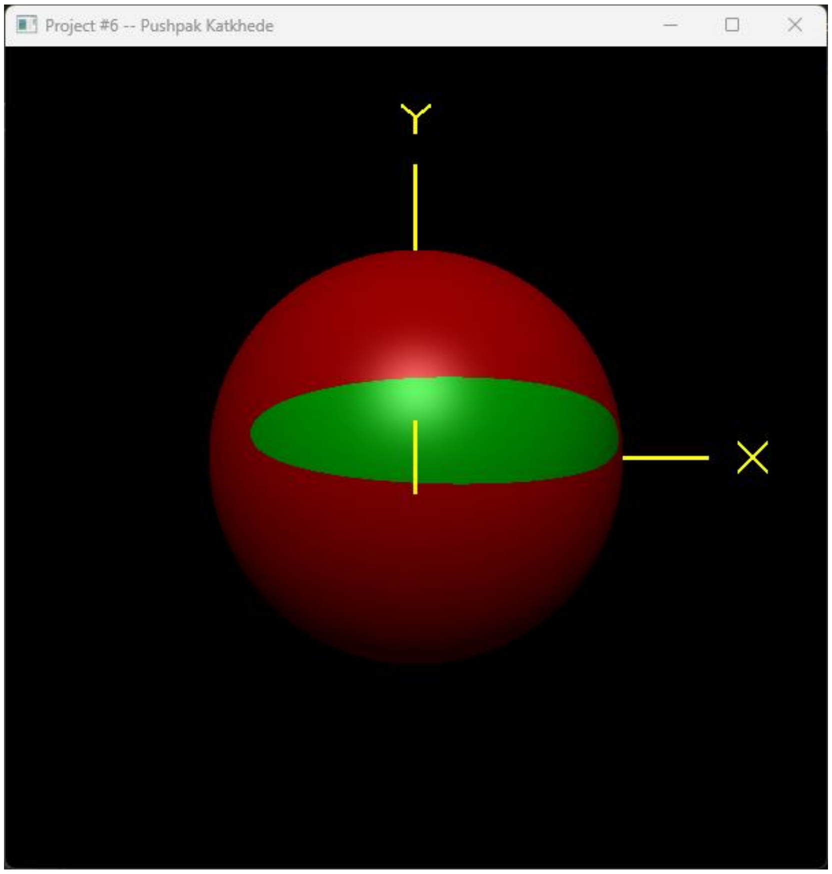
## ❖ Screenshots →

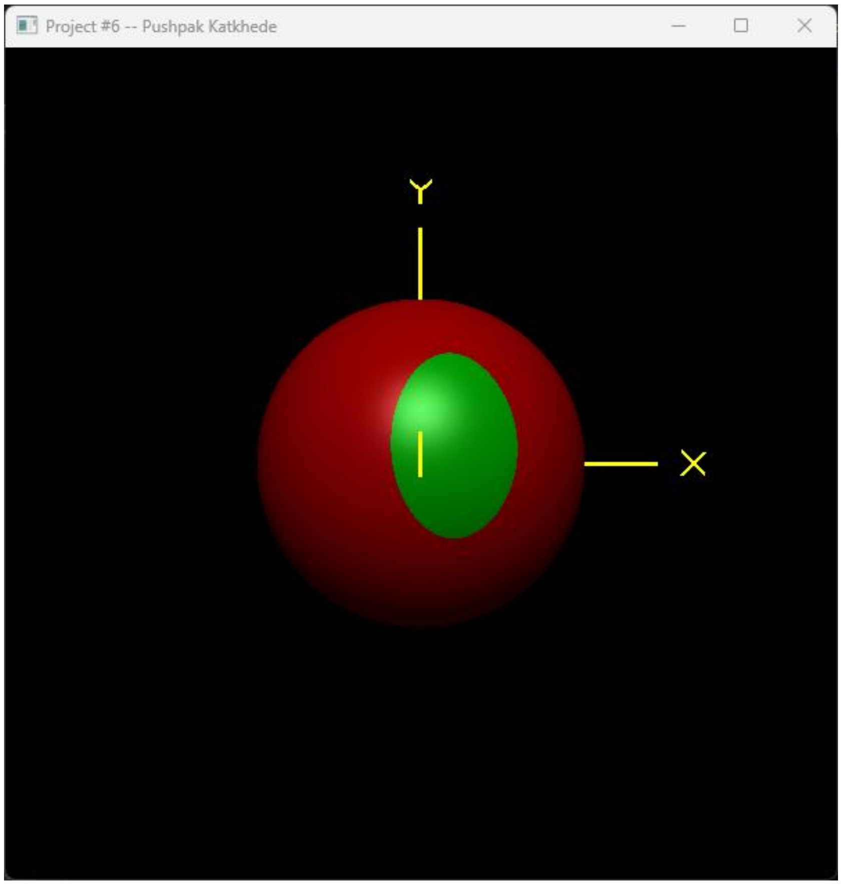


```
    Pattern.Init( );
    bool valid = Pattern.Create( "pattern.vert", "pattern.frag" );
    if( !valid )
        fprintf( stderr, "Could not create the Pattern shader!\n" );
    else
        fprintf( stderr, "Pattern shader created.!\n" );

    Pattern.Use( );
    Pattern.SetUniformVariable( "uKa", 0.1f );
    Pattern.SetUniformVariable( "uKd", 0.5f );
    Pattern.SetUniformVariable( "uKs", 0.4f );
    Pattern.SetUniformVariable( "uColor", 1.0f, 0.0f, 0.0f );
    Pattern.SetUniformVariable( "uSpecularColor", 1., 1., 1. );
    Pattern.SetUniformVariable( "uShininess", 12.f );
    Pattern.UnUse( );

    Sc.Init( );
    Sc.AddTimeValue( 0, 0.40f );
    Sc.AddTimeValue( 2, 0.70f );
    Sc.AddTimeValue( 4, 0.50f );
    Sc.AddTimeValue( 6, 0.70f );
    Sc.AddTimeValue( 8, 0.50f);
    Sc.AddTimeValue( 10, 0.40f );

    Tc.Init( );
    Tc.AddTimeValue( 0, 0.4f );
    Tc.AddTimeValue( 2, 0.70f);
    Tc.AddTimeValue( 4, 0.50f);
    Tc.AddTimeValue( 6, 0.70f );
    Tc.AddTimeValue(8, 0.50f);
    Tc.AddTimeValue( 10, 0.40f);
}
```



```
//  memory so that they can be played back efficiently at a later time
//  with a call to glCallList( )

void
InitLists( )
{
    if (DebugOn != 0)
        fprintf(stderr, "Starting InitLists.\n");

    glutSetWindow( MainWindow );

    // create the object:

    SphereList = glGenLists( 1 );

    glNewList( SphereList, GL_COMPILE );
    OsuSphere(1, 50, 50);
    glEndList();


    // create the axes:

    AxesList = glGenLists( 1 );
    glNewList( AxesList, GL_COMPILE );
        glLineWidth( AXES_WIDTH );

            Axes( 1.5 );
        glLineWidth( 1. );
    glEndList( );
}


// the keyboard callback:
```




Project #6 -- Pushpak Katkhede
Y
X