МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«Российский государственный университет им. А.Н. Косыгина   
(Технологии. Дизайн. Искусство)»**

Институт ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ И ЦИФРОВЫЕ ТРАНСФОРМАЦИИ

**Отчет по лабораторной работе № 12**

**по дисциплине «Объектно-ориентированное программирование»**

**Тема: «Использование фреймворка FireMonkey для взаимодействия с 3D-объектами»**

Выполнил: Сидоров Д. С., группа ИТС-123

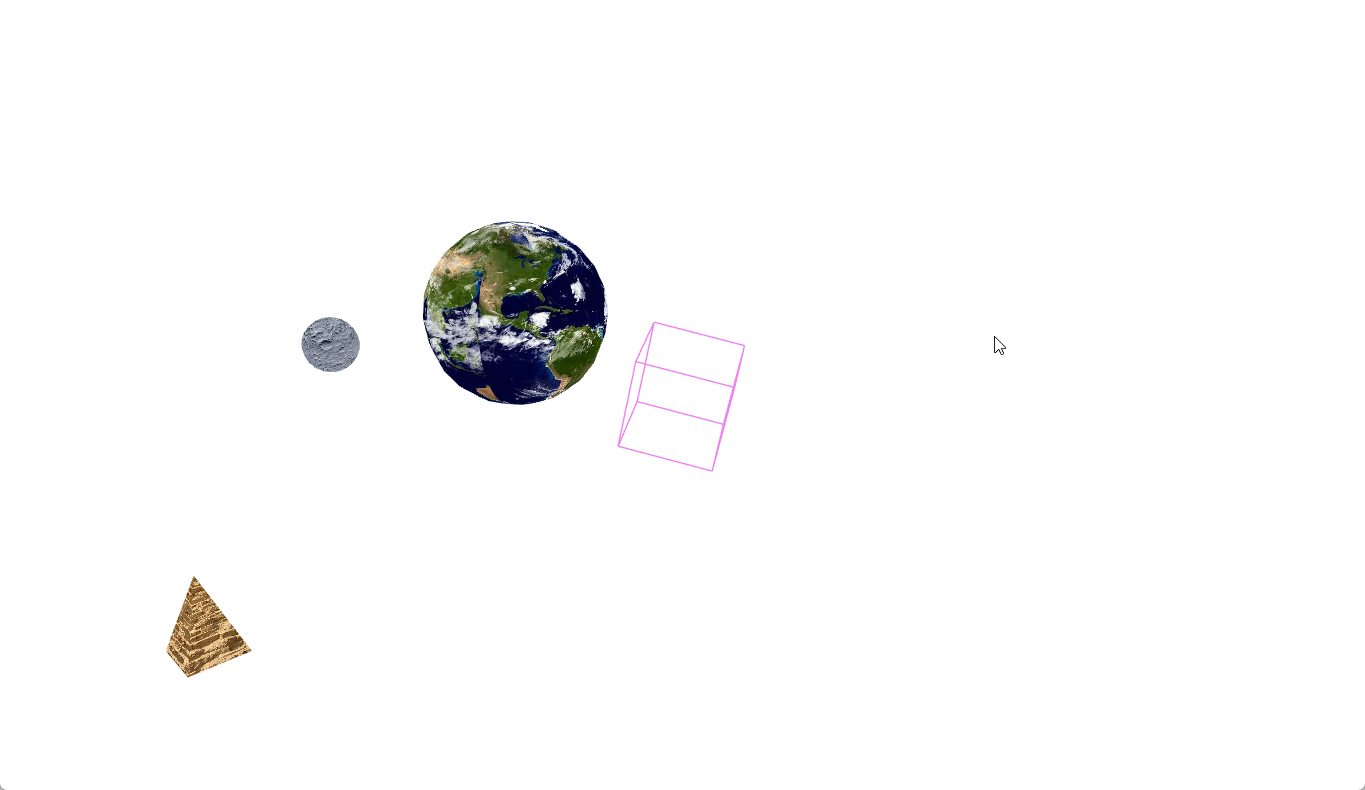
Проверил: к.т.н., доц. Семёнов А. А.

Москва, 2025г

**Использование фреймворка FireMonkey для взаимодействия с 3D-объектами**

Лабораторная работа посвящена изучению классов в ООП на C++. Лабораторную работу рекомендуется выполнить в ИСР C++Builder Community Edition. Задание: изучить и повторить проиллюстрированный ниже пример, демонстрирующий применение фреймворка FireMonkey.

Мы создадим кросс-платформенное приложение, позволяющее работать с компонентами FireMonkey. Необходимо воспроизвести рассмотренный пример, добавить возможность сдвига фигур (влево и вправо), а также: 1) вращения пирамиды; 2) имитации вращения планеты Земля и её спутника Луны.

*Рисунок 1. Демонстрация работы готовой программы.*

Для более показательного примера поместил куб в центр, его можно крутить, как и любую другую фигуру, а пирамиду можно двигать по экрану, зажав alt или ctrl (разные оси). Земля вращается вокруг точки в центре, а также вращается вокруг себя. Аналогично луна вращается вокруг земли и вокруг себя. Всем объектам дал соответствующие текстуры.

**Листинг кода**

**Unit1.cpp**

#include <fmx.h>

#pragma hdrstop

#include "Unit1.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.fmx"

TForm1 \*Form1;

//---------------------------------------------------------------------------

\_\_fastcall TForm1::TForm1(TComponent\* Owner)

: TForm3D(Owner)

{

// Начальные значения для орбиты

FAngle = 0;

FMajorAxis = 12.0; // Размер большой полуоси (по X)

FMinorAxis = 7.0; // Размер малой полуоси (по Z)

FOrbitSpeed = 0.02; // Скорость движения по орбите

FSpinSpeed = 1.0; // Скорость вращения вокруг оси

// Начальные значения для орбиты Луны

FMoonAngle = 0;

FMoonDistance = 5; // Расстояние от Земли до Луны

FMoonOrbitSpeed = 0.07; // Луна движется быстрее вокруг Земли

FMoonSpinSpeed = 0.5; // Луна вращается медленнее

// Создаем таймер

FOrbitTimer = new TTimer(this);

FOrbitTimer->Interval = 16; // примерно 60 FPS

FOrbitTimer->OnTimer = OrbitTimerEvent;

FOrbitTimer->Enabled = true;

// Расположим "Солнце" в центре

StrokeCube1->Position->X = 0;

StrokeCube1->Position->Y = 0;

StrokeCube1->Position->Z = 0;

// Начальное положение "Земли" на орбите

Sphere1->Position->X = FMajorAxis;

Sphere1->Position->Y = 0;

Sphere1->Position->Z = 0;

// Начальное положение "Луны" относительно "Земли"

MoonSphere->Position->X = Sphere1->Position->X + FMoonDistance;

MoonSphere->Position->Y = 0;

MoonSphere->Position->Z = 0;

// Установка размера Луны (меньше Земли)

MoonSphere->Width = Sphere1->Width \* 0.27;

MoonSphere->Height = Sphere1->Height \* 0.27;

MoonSphere->Depth = Sphere1->Depth \* 0.27;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::StrokeCube1MouseDown(TObject \*Sender, TMouseButton Button,

TShiftState Shift, float X, float Y, TVector3D &RayPos, TVector3D &RayDir)

{

FDown = ::PointF(X, Y);

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::StrokeCube1MouseMove(TObject \*Sender, TShiftState Shift, float X,

float Y, TVector3D &RayPos, TVector3D &RayDir)

{

if (Shift.Contains(ssLeft)) {

StrokeCube1->RotationAngle->X = StrokeCube1->RotationAngle->X - ((Y - FDown.Y) \* 0.3);

StrokeCube1->RotationAngle->Y = StrokeCube1->RotationAngle->Y + ((X - FDown.X) \* 0.3);

FDown = ::PointF(X, Y);

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::Sphere1MouseDown(TObject \*Sender, TMouseButton Button, TShiftState Shift,

float X, float Y, TVector3D &RayPos, TVector3D &RayDir)

{

FDown = ::PointF(X, Y);

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::Sphere1MouseMove(TObject \*Sender, TShiftState Shift, float X,

float Y, TVector3D &RayPos, TVector3D &RayDir)

{

if (Shift.Contains(ssLeft)) {

Sphere1->RotationAngle->X = Sphere1->RotationAngle->X - ((Y - FDown.Y) \* 0.3);

Sphere1->RotationAngle->Y = Sphere1->RotationAngle->Y + ((X - FDown.X) \* 0.3);

FDown = ::PointF(X, Y);

}

}

//---------------------------------------------------------------------------

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::Cone1MouseDown(TObject \*Sender, TMouseButton Button, TShiftState Shift,

float X, float Y, TVector3D &RayPos, TVector3D &RayDir)

{

FDown = ::PointF(X, Y);

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::Cone1MouseMove(TObject \*Sender, TShiftState Shift, float X,

float Y, TVector3D &RayPos, TVector3D &RayDir)

{

if (Shift.Contains(ssAlt)) {

// Перемещение по X и Y

Cone1->Position->X = Cone1->Position->X + ((X - FDown.X) \* 0.05);

Cone1->Position->Y = Cone1->Position->Y - ((Y - FDown.Y) \* 0.05);

}

else if (Shift.Contains(ssCtrl)) {

// Перемещение по Z при зажатом Ctrl

Cone1->Position->Z = Cone1->Position->Z + ((X - FDown.X) \* 0.05);

}

if (Shift.Contains(ssLeft)) {

Cone1->RotationAngle->X = Cone1->RotationAngle->X - ((Y - FDown.Y) \* 0.3);

Cone1->RotationAngle->Y = Cone1->RotationAngle->Y + ((X - FDown.X) \* 0.3);

}

FDown = ::PointF(X, Y);

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm1::OrbitTimerEvent(TObject \*Sender)

{

// Увеличиваем угол

FAngle += FOrbitSpeed;

if (FAngle > 6.28318) // 2\*Pi

FAngle -= 6.28318;

// Вычисляем положение на эллипсе

float x = FMajorAxis \* cos(FAngle);

float z = FMinorAxis \* sin(FAngle);

// Обновляем положение "Земли"

Sphere1->Position->X = x;

Sphere1->Position->Z = z;

// Вращаем "Землю" вокруг собственной оси Y

Sphere1->RotationAngle->Y += FSpinSpeed;

if (Sphere1->RotationAngle->Y > 360)

Sphere1->RotationAngle->Y -= 360;

// Увеличиваем угол Луны

FMoonAngle += FMoonOrbitSpeed;

if (FMoonAngle > 6.28318)

FMoonAngle -= 6.28318;

// Вычисляем положение Луны относительно Земли

float moonRelativeX = FMoonDistance \* cos(FMoonAngle);

float moonRelativeZ = FMoonDistance \* sin(FMoonAngle);

// Обновляем положение "Луны" (добавляем к координатам Земли)

MoonSphere->Position->X = Sphere1->Position->X + moonRelativeX;

MoonSphere->Position->Z = Sphere1->Position->Z + moonRelativeZ;

MoonSphere->Position->Y = Sphere1->Position->Y; // Та же высота, что и у Земли

// Вращаем "Луну" вокруг собственной оси Y

MoonSphere->RotationAngle->Y += FMoonSpinSpeed;

if (MoonSphere->RotationAngle->Y > 360)

MoonSphere->RotationAngle->Y -= 360;

}

void \_\_fastcall TForm1::TrackBarMoonSpeedChange(TObject \*Sender)

{

FMoonOrbitSpeed = TrackBarMoonSpeed->Value / 1000.0;

**Unit1.h**

//---------------------------------------------------------------------------

#ifndef Unit1H

#define Unit1H

//---------------------------------------------------------------------------

#include <System.Classes.hpp>

#include <FMX.Controls.hpp>

#include <FMX.Forms.hpp>

#include <FMX.Forms3D.hpp>

#include <FMX.Ani.hpp>

#include <FMX.Controls3D.hpp>

#include <FMX.MaterialSources.hpp>

#include <FMX.Objects3D.hpp>

#include <FMX.Types.hpp>

#include <System.Math.Vectors.hpp>

#include <FMX.Controls.Presentation.hpp>

#include <FMX.StdCtrls.hpp>

//--------------------------------------------------------------------------

class TForm1 : public TForm3D

{

\_\_published: // IDE-managed Components

TSphere \*Sphere1;

TTextureMaterialSource \*TextureMaterialSource1;

TFloatAnimation \*FloatAnimation1;

TStrokeCube \*StrokeCube1;

TFloatAnimation \*FloatAnimation2;

TSphere \*MoonSphere;

TTextureMaterialSource \*MoonMaterial;

TTrackBar \*TrackBarMoonSpeed;

TCone \*Cone1;

TTextureMaterialSource \*PyramidMaterial;

void \_\_fastcall StrokeCube1MouseDown(TObject \*Sender, TMouseButton Button, TShiftState Shift,

float X, float Y, TVector3D &RayPos, TVector3D &RayDir);

void \_\_fastcall StrokeCube1MouseMove(TObject \*Sender, TShiftState Shift, float X,

float Y, TVector3D &RayPos, TVector3D &RayDir);

void \_\_fastcall Sphere1MouseDown(TObject \*Sender, TMouseButton Button, TShiftState Shift,

float X, float Y, TVector3D &RayPos, TVector3D &RayDir);

void \_\_fastcall Sphere1MouseMove(TObject \*Sender, TShiftState Shift, float X, float Y,

TVector3D &RayPos, TVector3D &RayDir);

void \_\_fastcall TrackBarMoonSpeedChange(TObject \*Sender);

void \_\_fastcall Cone1MouseDown(TObject \*Sender, TMouseButton Button, TShiftState Shift,

float X, float Y, TVector3D &RayPos, TVector3D &RayDir);

void \_\_fastcall Cone1MouseMove(TObject \*Sender, TShiftState Shift, float X, float Y,

TVector3D &RayPos, TVector3D &RayDir);

private: // User declarations

TPointF FDown;

float FAngle;

float FMajorAxis;

float FMinorAxis;

float FOrbitSpeed;

float FSpinSpeed;

TTimer\* FOrbitTimer;

float FMoonAngle; // Угол Луны вокруг Земли

float FMoonDistance; // Расстояние от Земли до Луны

float FMoonOrbitSpeed; // Скорость движения Луны вокруг Земли

float FMoonSpinSpeed; // Скорость вращения Луны

void \_\_fastcall OrbitTimerEvent(TObject \*Sender);

public: // User declarations

\_\_fastcall TForm1(TComponent\* Owner);

};

//--------------------------------------------------------------------------

extern PACKAGE TForm1 \*Form1;

//--------------------------------------------------------------------------

#endif

**Вывод:** В ходе выполнения лабораторной работы были изучены основные возможности фреймворка FireMonkey для работы с трехмерными объектами. Фреймворк демонстрирует эффективную реализацию принципов ООП: инкапсуляцию сложной функциональности 3D-графики в удобные компоненты, наследование от базовых классов (TControl3D) и полиморфизм при управлении разными типами 3D-объектов. Компонентный подход FireMonkey позволяет создавать интерактивные 3D-приложения, используя готовые элементы (TCube, TSphere, TCylinder) и настраивая их свойства как визуально, так и программно. Была освоена работа с трансформациями объектов (перемещение, вращение, масштабирование), применение материалов и текстур, а также настройка освещения и камер для улучшения восприятия 3D-сцены. Важным аспектом работы стало изучение обработки событий взаимодействия пользователя с объектами и создание анимаций с помощью специализированных классов. Практические навыки, полученные в ходе лабораторной работы, демонстрируют, как абстрактные принципы ООП применяются для решения конкретных задач разработки графических приложений.