**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 4](#_Toc40809808)

[1 СРЕДСТВА СОЗДАНИЯ ИГРОВОГО ПРИЛОЖЕНИЯ 5](#_Toc40809809)

[1.1 Графическая библиотека OpenGL 5](#_Toc40809811)

[1.2 Использование OpenGl в языке программирования С# 6](#_Toc40809812)

[1.3 Интерфейс программирования приложений Windows Form 7](#_Toc40809813)

[1.4 Использование OpenTK в приложениях Windows Form 8](#_Toc40809814)

[Список используемых источников](#_Toc40809823) 11

**ВВЕДЕНИЕ**

В досуге современного человека компьютерные игры занимают достаточно важное место. Для многих игры – это вообще повседневное развлечение. 2020 год показал насколько игры важны для людей и перспективны для развития.

Для сравнения, объем рынка игр уже обогнал другие традиционные развлечения, такие как музыка и фильмы вместе взятые, $145.7 против $21.5 и $101 миллиардов соответственно.

Поэтому сегодня мы поговорим о более сложной и не всем знакомой части игр, это их разработка. Рассмотрим один из языков программирования для разработки игровых приложений, различные методы создания игр и технологии.

Компьютерная графика нашла широкое распространение и применение в повседневной жизни. Учёные используют компьютерную графику для анализа результатов моделирования. Инженеры и архитекторы используют трёхмерную графику для создания виртуальных моделей. Кинематографисты создают спецэффекты или полностью анимированные фильмы («Шрек», «История игрушек» и др.). В последние годы широкое распространение получили также компьютерные игры, максимально использующие трёхмерную графику для создания виртуальных миров.

В нашем проекте мы будем использовать графическую библиотеку OpenGl и средства языка программирования C# Windows Forms.

Я взялся за данные проект, так как хочу освоить и применить на практике различные технологии и средства разработки игровых приложений. В ходе выполнения проекта мы окунемся мир игровой разработки и самое главное получим достаточно большой багаж новых знаний, начиная от опыта работы над серьёзными проектами, заканчивая работой с различными паттернами программирования.

1. **Средства создания игровых приложений**
   1. **Графическая библиотека OpenGL**

Для создания графического интерфейса и отображения объектов в нашем игровом приложении будем использовать компьютерную графику OpenGL.

*OpenGL* (*Open Graphics Library*) – спецификация, определяющая платформонезависимый (независимый от [языка программирования](https://ru.wikipedia.org/wiki/%D0%AF%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F)) [программный интерфейс](https://ru.wikipedia.org/wiki/API) для написания приложений, использующих двумерную и трёхмерную [компьютерную графику](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%8F_%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D0%BA%D0%B0). Разрабатывается в США и Европе, имеет тип лицензий *GNU*-/*EU*/.

Включает более 300 функций для рисования сложных трёхмерных сцен из простых примитивов. Используется при создании [компьютерных игр](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%8F_%D0%B8%D0%B3%D1%80%D0%B0), [САПР](https://ru.wikipedia.org/wiki/%D0%A1%D0%90%D0%9F%D0%A0), [виртуальной реальности](https://ru.wikipedia.org/wiki/%D0%92%D0%B8%D1%80%D1%82%D1%83%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D1%80%D0%B5%D0%B0%D0%BB%D1%8C%D0%BD%D0%BE%D1%81%D1%82%D1%8C), визуализации в научных исследованиях. Эффективные реализации *OpenGL* существуют для *Windows*, *Unix*-платформ, *PlayStation* 3 и *Mac* *OS*. На платформе [*Windows*](https://ru.wikipedia.org/wiki/Windows) конкурирует с [*Direct3D*](https://ru.wikipedia.org/wiki/Direct3D).

На базовом уровне, *OpenGL* – это просто спецификация, то есть документ, описывающий набор функций и их точное поведение. Производители оборудования на основе этой спецификации создают реализации – библиотеки функций, соответствующих набору функций спецификации.

*OpenGL* является низкоуровневым процедурным *API*, что вынуждает программиста диктовать точную последовательность шагов, чтобы построить результирующую растровую графику (императивный подход). Это является основным отличием от дескрипторных подходов, когда вся сцена передается в виде структуры данных (чаще всего дерева), которое обрабатывается и строится на экране. С одной стороны, императивный подход требует от программиста глубокого знания законов трёхмерной графики и математических моделей, с другой стороны – даёт свободу внедрения различных инноваций.

OpenGL ориентируется на следующие две задачи:

* скрыть сложности работы различных 3D ускорителей от разработчика, предоставляя единый API;
* скрыть различия в аппаратных возможностях устройств, требуя программной эмуляции недостающего функционала.

Отличительной особенностью *OpenGL* является поддержка расширений. Стандарт *OpenGL*, с появлением новых технологий, позволяет отдельным производителям добавлять в библиотеку функциональность через механизм расширений. Всякий раз, когда графическая компания выкатывает новую методику или новую большую оптимизацию для рендеринга, это часто встречается в расширении, реализованном в драйверах. Если оборудование, на котором работает приложение, поддерживает такое расширение, то разработчик может использовать функционал, предоставляемый этим расширением, для более продвинутой или эффективной графики. Таким образом, графический разработчик уже может использовать новые методы рендеринга, просто проверяя, поддерживается ли данное расширение видеокартой, при этом не дожидаясь, пока *OpenGL* добавит этот функционал в свою новую версию. Часто, когда расширение является популярным или очень полезным, оно в конечном итоге становится частью новой версии *OpenGL*.

Разработчик должен знать, доступны ли какие-либо из этих расширений, прежде чем их использовать (или использовать библиотеку расширений *OpenGL*). Это позволяет разработчику делать вещи лучше или эффективнее в зависимости от того, доступно ли расширение.

**1.2 Использование OpenGl в языке программирования С#**

Так как для реализации игрового приложения будет использоваться язык программирования C#, будем использовать библиотеку OpenTK. OpenTK – это одна из самых простых и популярных библиотек обеспечивающих доступ к графические инструментам OpenGL.

*Open* *Toolkit* является бесплатным проектом, который позволяет использовать *OpenGL*, *OpenGL* | *ES*, *OpenCL* и *OpenAL* *API*, из управляемых языков.

*OpenTK* начал жизнь в качестве экспериментальной вилки рамках *Тао*, прежде чем летом первоначального намерения 2006. Это был обеспечить чистый обертку, чем *Tao*. *OpenGL*, но он быстро вырос в фокусе: прямо сейчас, он обеспечивает доступ к различным *Khronos* и *Creative*, *API* для ручки и необходимую логику инициализации для каждого API. Таким образом, *Open* *Toolkit*, наиболее похожий на таких проектах, как *Тао*, *SlimDX*, *SDL* или *GLFW*.

В отличие от аналогичных библиотек, попытки *OpenTK* обеспечить единый интерфейс, который использует превосходную удалось выполнения. Вместо того, чтобы нетипизированных указателей, *OpenTK* обеспечивает дженерики. Вместо простых констант, *OpenTK* использует строго типизированные перечисления. Вместо списков простых функций, *OpenTK* отделяет функции за расширение категории. Общая математическая библиотека является интегрированной и непосредственно использовать в каждом *API*.

**1.3 Интерфейс программирования приложений Windows Form**

Отображать графику OpenGL будем с помощью с графического интерфейса Windows forms.

Windows Forms – интерфейс программирования приложений (API), отвечающий за графический интерфейс пользователя и являющийся частью Microsoft .NET Framework. Данный интерфейс упрощает доступ к элементам интерфейса Microsoft Windows за счет создания обёртки для существующего Win32 API в управляемом коде. Причём управляемый код – классы, реализующие API для Windows Forms, не зависят от языка разработки.

То есть программист одинаково может использовать Windows Forms как при написании ПО на C#, С++, так и на др.

На рисунках 1–2 демонстрируется создание приложения в Windows Forms.

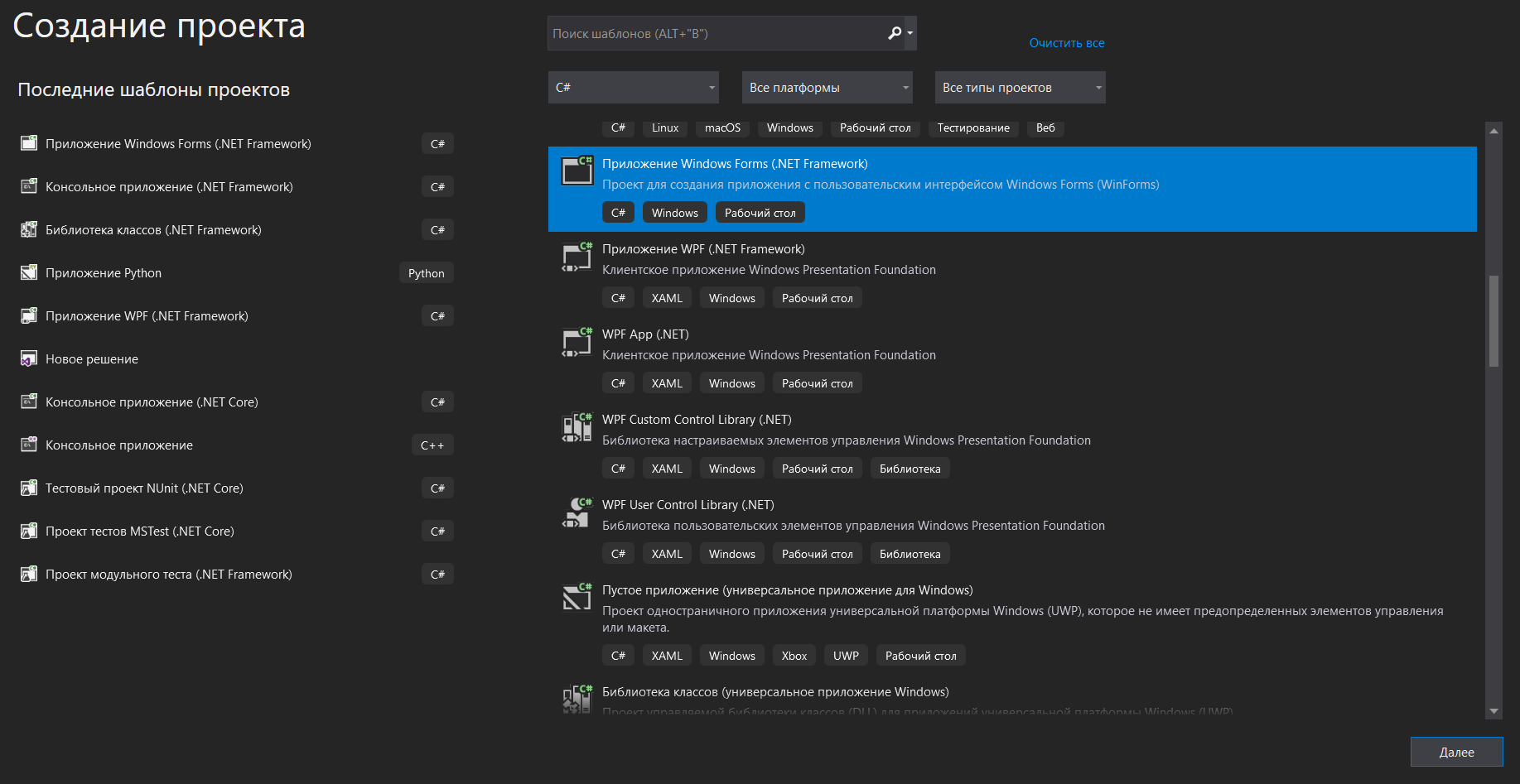


Рисунок 1 – Создание приложения Windows Forms

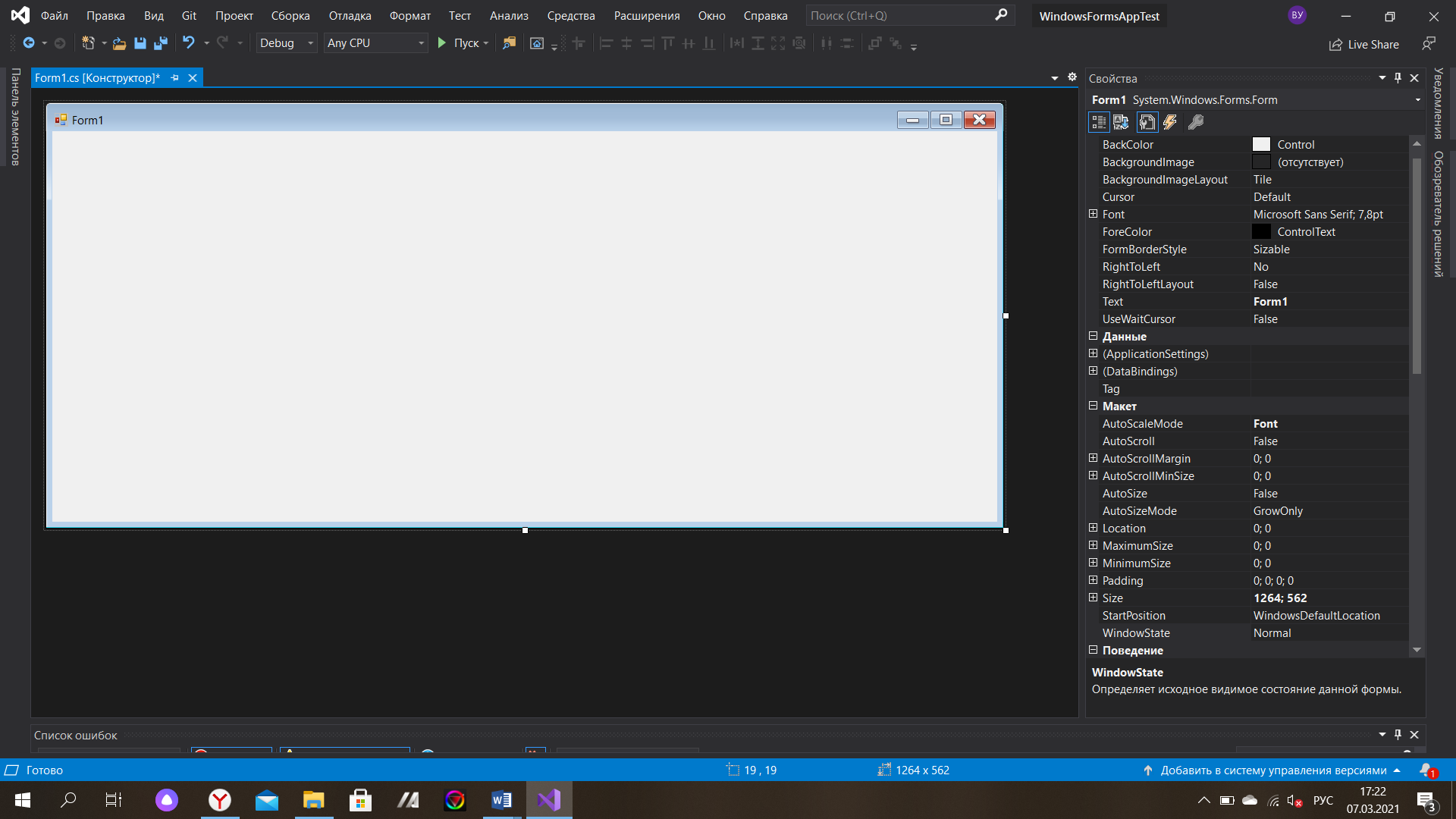


Рисунок 2 – Интерфейс работы с приложением

**1.4 Использование OpenTK в приложениях Windows Form**

Библиотка OpenTKудобна тем, что ставится очень быстро и легко через NuGet и позволяет, как создавать своё окно из консольного проложения, так и встраивать холст для рисования OpenGL графики в WPF и WinForms приложения, сочетая 2D/3D графику со стандартным GUI в одном приложении.

Для подключения OpenTK к нашему приложению воспользуемся NuGet – системой управления пакетами для платформ разработки Microsoft. Чтобы получить весь функционал нам потребуется подключить две библиотеки OpenTK и OpenTK.GLControl.

Библиотека OpenTK – это набор быстрых низкоуровневых привязок C# для OpenGL.

Библиотека OpenTK.GLControl выполняет инициализацию OpenGL с помощью графический компонент GLControl, который размещается на одной из форм приложения. Вывод изображения выпоняется в области компонента GLControl.

Подключение OpenTK в приложение Windows Forms:

1. Заходим в диспетчер пакетов NuGet и устанавливаем пакеты OpenTK и OpenTK.GLControl;

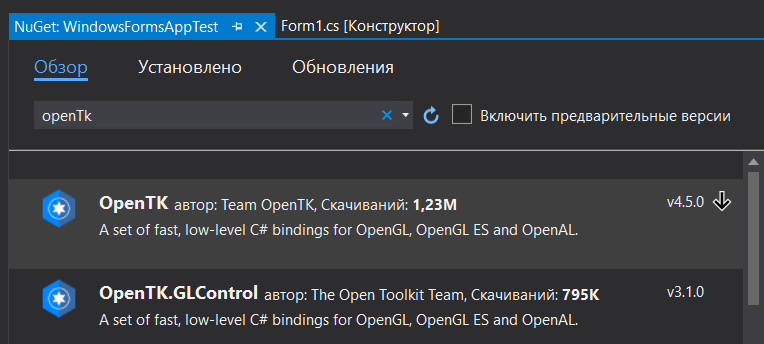


Рисунок 3 – Диспетчер пакетов NuGet

1. Для того чтобы вывести изображение необходимо добавить компонент GLControl на нашу форму. Заходим в панель элементов вкладка OpenTK.

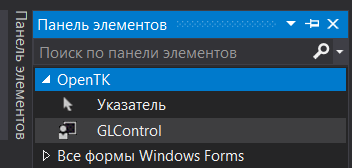


Рисунок 4 – Добавление компонент GLControl

1. Добавим GLControl на форму;

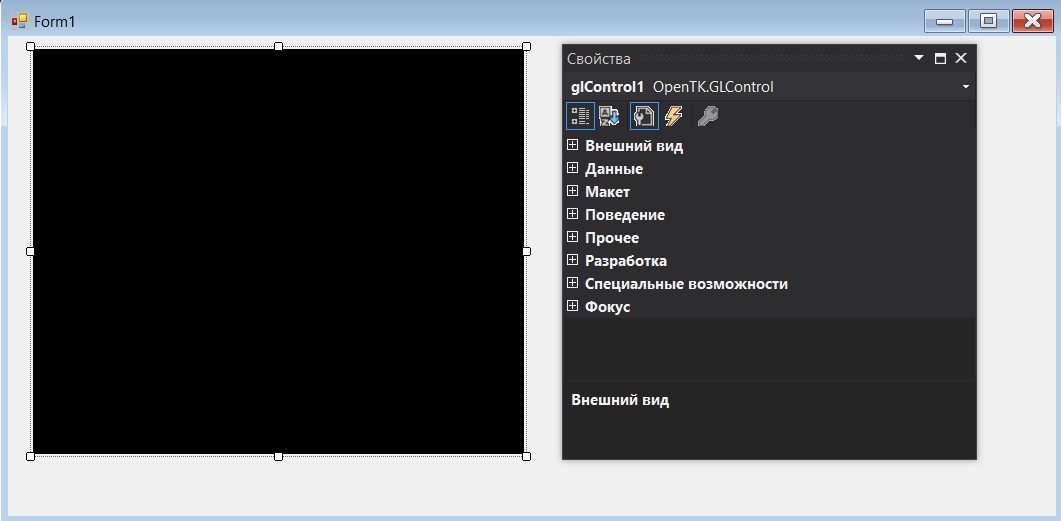


Рисунок 5 – Компонент GLControl

В этой главе рассказывалось о базовых знаниях использовании OpenGl в приложениях Windows Forms. Для наращивания функционала игрового приложения, нужно использовать множество различных методов и функций, о них будет рассказано в следующих главах.

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Программирование компьютерной графики. Современный *OpenGL* / Боресков А.В. – ДМК-Пресс: 2019. – 373с.
2. Паттерны проектирования в C# и .NET – 2017. – Режим доступа: https://metanit.com/sharp/patterns/. – Дата доступа: 04.03.2021.
3. *CLR* via C#. Программирование на платформе *Microsoft* .*NET* *Framework* 4.5 на языке C#/. СПб.:Питер, 2020. –896c.
4. *OpenGL SuperBible* / *Graham Sellers, Richard S. Wright Jr., Nicholas* *Haemel* – *Addison*-*Wesley*: 2015 –105c.