Государственное бюджетное общеобразовательное учреждение города Москвы «ГБОУ Лицей № 1547»

Итоговый индивидуальный проект

***Создание приложения при помощи языка***

***С++ с подробным объяснением***

*Махоткин Артём Сергеевич*

*Гафилин Егор Романович*

*Научный руководитель: Кекконен Эрик Артурович*

Москва

2022

## 

## Оглавление

[Оглавление 2](#_Toc101237071)

[Введение 3](#_Toc101237072)

[Основная часть 3](#_Toc101237073)

[Заключение 9](#_Toc101237074)

[Список литературы 10](#_Toc101237075)

## Введение

## В интернете довольно мало работ, которые будут понятны для людей, которые только приступили к изучению программирования и освоению библиотек для работы с графикой. Поэтому наш проект будет помогать разобраться новичкам в новой среде.

Цель нашего проекта – помочь новичкам в освоении C++ и работе с графикой.

## Для разработки нашего приложения мы решили использовать язык C++, потому что он объектно-ориентированный, довольно быстро работает, есть на многих операционных системах и имеет достаточно много библиотек, расширяющих его возможности, кроме этого, многие языки программирования имеют схожий с ним синтаксис, что в разы упрощает изучение других языков при необходимости, ну и наконец, многие языки программирования, что были придуманы позднее, имеют упрощенные структуру и механизмы, а процесс изучения сводится к освоению синтаксиса и ПО. Это приводит к тому, что появляется множество программистов, которые просто не понимают, что создают в половине кода. Для отрисовки графики мы решили использовать библиотеку “windows.h”, потому что он удобен для работы графикой, ведь использует метод закраски пикселей, которые легко запоминаются в программе и не тратит много памяти, а также позволяет работать со звуком, окнами и другими параметрами, а также OpenGL, так как он довольно удобен и имеет огромный функционал и поддерживается такими крупными фирмами-разработчиками, как Microsoft, Sun Microsystems, Evans & Sutherland, Hewlett-Packard Corporation, Intel Corporation и Intergraph Corporation. OpenGL освобождает программиста от написания программ для конкретного оборудования. Если устройство поддерживает какую-то функцию, то эта функция выполняется аппаратно, если нет, то библиотека выполняет её программно. Используя данную библиотеку, можно создавать объекты из вершин, задать им местоположение в трёхмерном пространстве, настроить другие параметры (например поворот, масштаб), указать свойства объектов (цвет, текстура, материал, шейдер), положение наблюдателя, а сама библиотека OpenGL позаботится о том, чтобы отобразить всё это на экране. Но учитывая то, что OpenGL по большей части позволяет лишь отображать нарисованную графику, windows.h будет отличным дополнением для программы.

## Основная часть

Собственно, как же работает отрисовка 3D моделей? Для начала мы изучили существующие форматы моделей, их сейчас довольно много, например один из самых популярных - OBJ, но увы он не поддерживает анимации, есть ещё похожий по структуре – SMD, но к сожалению он весит очень много, поэтому мы решили остановиться на формате FBX, его преимущество в том, что он поддерживает анимации и мало весит.

Разобрать как он работает было не просто, так как формат этот бинарный, т.е. вся информация хранится в байтах, тем самым файл весит меньше, но разобраться в нём становится куда сложнее. Для его разбора сначала считаем весь файл в массив байтов. Далее первые 23 байта можно пропустить, в них указывается формат файла (Kaydara FBX Binary) далее следующие 4 байта содержат в себе версию модели, различия там не большие, в новых версиях информация кодируется большим количеством байт.

Далее идут сами элементы, там хранится вся информация модели, начиная от информации о файле (кто его создал, в какой момент времени), заканчивая массивами вершин, индексами, текстурами и даже видео. Сами элементы начинаются с числа (смещения), оно означает количество байтов до следующего элемента, если оно равно нулю, то значит это конец файла. После этого идёт ещё одно число (4 байта) и разделитель (4 байта), это количество свойств элемента. После идёт название элемента (4 байта), и сами свойства, где сначала идёт символ (1 байт), например “i” это массив целых чисел. Далее идут сами свойства. После них идут вложенные элементы по той же схеме. После этого, разумеется, это всё нужно считать в память программы.

Далее, чтобы уже нарисовать нашу модель на экране нужно достать из этой информации элемент с названием “Geometry” и достать оттуда вершины (Vertices), индексы (PolygonVertexIndex) и координаты текстур (LayerElementUV -> UV) при необходимости.

Как мы знаем OpenGL может рисовать только примитивные фигуры, к примеру треугольники, четырёхугольники и линии, так что, используя несколько треугольников можно как раз и отобразить модель. В файле с моделью для каждого из её элементов есть свой массив вершин и индексов, к примеру рассмотрим построение куба:

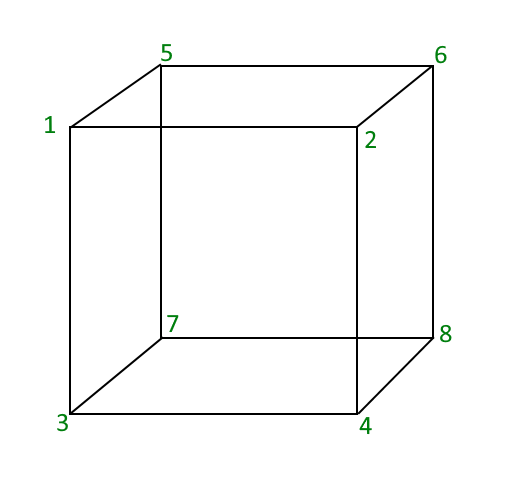


Рис. 1

Как мы видим у каждой вершины есть свой индекс и у каждого из этих индексов есть своё положение в пространстве, и чтобы отобразить подобную модель на экране нужно разбить её на простые составляющие, например выбрать в OpenGL режим рисование четырёхугольников и передать положение каждой вершины, т.е. последовательно передать значения индексов: 1234 5678 1537 2648 3748 1562. Как мы видим каждые 4 индекса образуют квадрат и передав в библиотеку значения этих самых индексов (положение вершины в пространстве) OpenGL сам отобразит куб на экране, ну и, разумеется, вы всё так же сможете поворачивать, масштабировать, перемещать систему координат, тем самым и меняя положение кубика.

Но помимо самих моделей для них есть ведь ещё и анимации, но как же их отображать, если положение всех вершин уже задано, не делать же для каждого кадра отдельную модель? Разумеется нет, в FBX указываются сами объекты и анимации к ним, то есть насколько нужно повернуть, сдвинуть или отмасштабировать объект. Но учитывая то, что мы работаем в OpenGL мы можем сделать всё проще, просто сдвинуть систему координат, повернуть её или отмасштабировать. И так для каждого объекта.

Также с помощью библиотеки windows.h была создана программа, в которой будет описано каждое действие, которое будет понятно для новичка в программировании. Но как же все-таки работает эта библиотека с графикой?  
Для начала, надо создать окно с произвольным разрешением, которое каждый может выбрать под себя. В данном приложение мы использовали разрешение 1280х720. Дальше мы сделали, чтобы окно сворачиваться и разворачиваться, для большей удобности в использовании программы, а также окно может менять свое разрешение в любом формате. С помощью нескольких функций, мы создаем геометрические фигуры методом закрашивания пикселей, запоминающиеся в программе, что очень удобно для сохранения, а также будут использоваться в качестве игровых объектов, а также фоном.   
Создавать объекты — это интересно, но надо ими управлять. И при нажатии клавиш, которые мы задали в качестве управляющих, программа считывает это действие и обрабатывает его, сохраняя его в коде.

## Заключение

В ходе выполнения работы нами были изучены и решены следующие задачи:

* + - 1. Создание полноценной игры на OpenGL и “windows.h”.
      2. Создание библиотеки для отрисовки полноценных 3D моделей с поддержкой анимаций.

Отличительной чертой данного проекта является подробное описание к каждому действию в программе, что может помочь новичкам разобраться как всё работает. Кроме этого, наша библиотека одна из не многих, которая позволяет отрисовывать анимированные модели и при этом весит довольно мало.

Одним из явных недостатков, является отсутствие поддержки скелетных анимаций в нашей библиотеке для отрисовки моделей, а она позволяет хранить в файле с моделью сразу несколько анимаций, тем самым экономя место не повторяя храня одну и ту же модель в разных файлах. Вполне вероятно в будущем мы исправим это.

В заключении можем сказать, что изначальные задачи были выполнены.

## Дальнейшие перспективы

Наш проект в дальнейшем можно совершенствовать, добавляя новый функционал, из дальнейших планов:

1. Удобная система для использования шейдеров
2. Система для создания интерфейсов при помощи HTML+CSS+JavaScript
3. Создание библиотеки для упрощения работы с графикой, где парой строчек можно будет рисовать сложные объекты, графики и многое другое.
4. Создание подобных игр с подробным описанием для большего понимания работы библиотек

## Список литературы

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | C++ | [[ссылка]](https://isocpp.org/) | Язык программирования C++, на котором пишется приложение. |
| 2 | OpenGL | [[ссылка]](https://www.opengl.org/) | Библиотека для работы с графикой. |
| 3 | Фрагментный Шейдер | [[ссылка]](https://www.khronos.org/opengl/wiki/Fragment_Shader) | Программа для отрисовки графики, используется для создания различных эффектов. |
| 4 | Visual Studio | [[ссылка]](https://visualstudio.microsoft.com/) | Среда разработки C++. |
| 5 | windows.h | [[ссылка]](https://docs.microsoft.com/en-us/windows/win32/api/winbase/) | Базовая библиотека для создания приложений на C++. |
| 6 | Книга про  OpenGL | [[ссылка]](http://topuch.ru/opengl-red-book-russkaya-versiya/index3.html) | Официальное руководство по изучению OpenGL. |
| 7 | Книга про  C++ | [[ссылка]](https://arduinoplus.ru/knigi/tehnologiya-programmirovaniya-na-c-litvinenko/) | Руководство по созданию приложений на C++ |
| 8 | Разбор формата  FBX | [[ссылка]](https://code.blender.org/2013/08/fbx-binary-file-format-specification/) | Здесь можно почитать про алгоритм считывания данных из FBX файлов |