Всероссийский конкурс исследовательских и проектных работ школьников «Высший пилотаж» Всероссийский конкурс-конференция школьников «Авангард»

Создание приложения для просмотра файлов карт игры “Warhammer 40000 Dawn of War Soulstorm” на мобильные устройства

Проект

Автор: Шиян Артур,
Учащийся 10 класса,
ЧОУ «Первая гимназия Максимум»
Г. Пенза

2024 г.

Оглавление.

Введение.

Для людей, любящих отдыхать в старых компьютерных играх, существует огромное множество проблем, которые не дают в полной мере насладиться игровым процессом, что приводит к оттоку игроков из онлайн проектов, недостатку новых, постепенному исчезновению сообществ. Мой проект нацелен на

Целью моего проекта является создание приложения в кроссплатформенной среде разработки игр Unity для просмотра файлов карт “Warhammer 40000 Dawn of War Soulstorm” на мобильные устройства, чтобы упростить доступ к информации об игре и повысить интерес к ней.

В ходе проекта будет изучена структура файла, хранящего информацию об игровой местности – картах, и написан интерфейс для их отображения.

Результатом работы будет скомпилированное под смартфоны Android приложение, с вышеупомянутым функционалом.

Глава 1 Теоретическая часть.

1.1 Бинарные файлы

Бинарный файл – это файл, в котором информация представлена в виде последовательности бит, то есть в виде нулей и единиц, понятных компьютеру. Именно в таких файлах хранятся данные о картах. Такие файлы нельзя прочесть в текстовом редакторе, понадобятся дополнительные программы для их декодирования.

1.2 010 Editor

Основной инструмент для работы с бинарными файлами. Утилита сразу представляет содержимое в 16-ричном формате для более удобного чтения, перевод информации в кодировку UTF-8 для изучения структуры файла и понятный интерфейс.

1.3 Python

Очень популярный интерпретируемый язык программирования, будет использован для написания небольших программ, которые помогут ускорить процесс анализа файлов карт.

1.4 W40kME

Сам редактор карт, в котором будет проведено больше всего времени. В этой программе можно видоизменять любые параметры файлов начиная с ландшафта и заканчивая эффектами при ходьбе по местности.

1.5 Unity

Лучшая кроссплатформенная среда разработки компьютерных игр или просто игровой “движок” для одиночной разработки на данный момент. Предоставляет весь функционал, который нужен при написании игр на смартфоны.

Глава 2 Практическая часть

2.1 Изучение структуры .sgb файлов

2.1.1 Хранение карт в памяти

Каждая карта в игре “Dawn of war” состоит из нескольких файлов:

1. .sgb файла
2. Одного или нескольких .tga файлов

.tga отвечают за отображение карт в меню игры, поэтому они нам не понадобятся. Что касается .sgb, то именно эти файлы отвечают за описание внешнего вида карты, с ними мы и будем работать.

2.1.2 Реверс инжиниринг .sgb

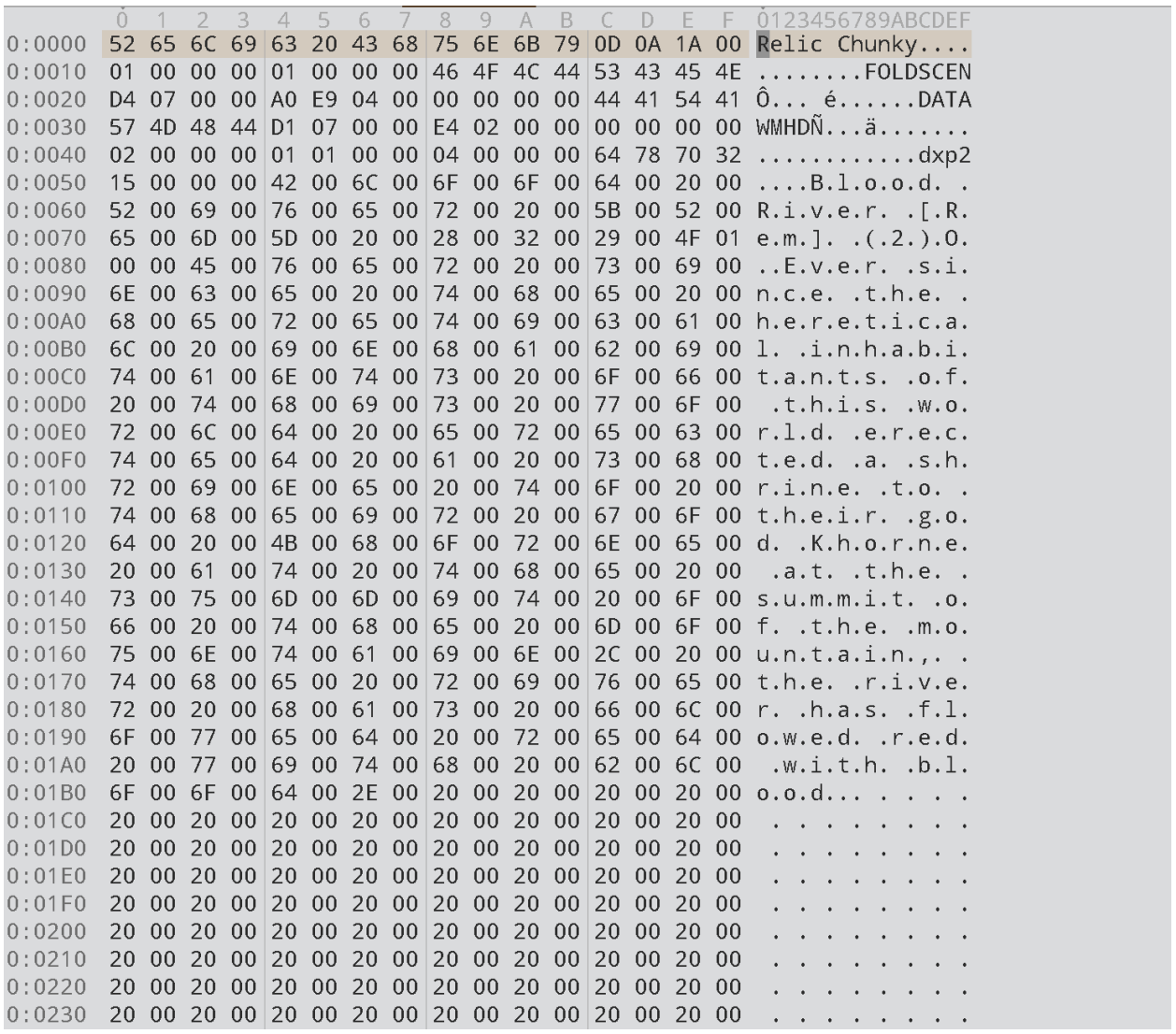
Обратная разработка или же “реверс инжиниринг” файла карты.

Первое, что бросается в глаза при просмотре игровой директории – это размеры .sgb разных карт, они не являются фиксированными.

Можно предположить, что в самих файлах есть “метки”, специальная последовательность байтов, которая оповещает игру об областях в памяти.

Для подтверждения данной теории нужно открыть случайный .sgb и попробовать проанализировать содержимое.

Слева находятся адреса первого байта каждой строки (строка – 16 байт), сверху - номера байтов в строке, справа – колонка с автоматическим переводом байтов в UTF-8, которой мы и будем пользоваться.

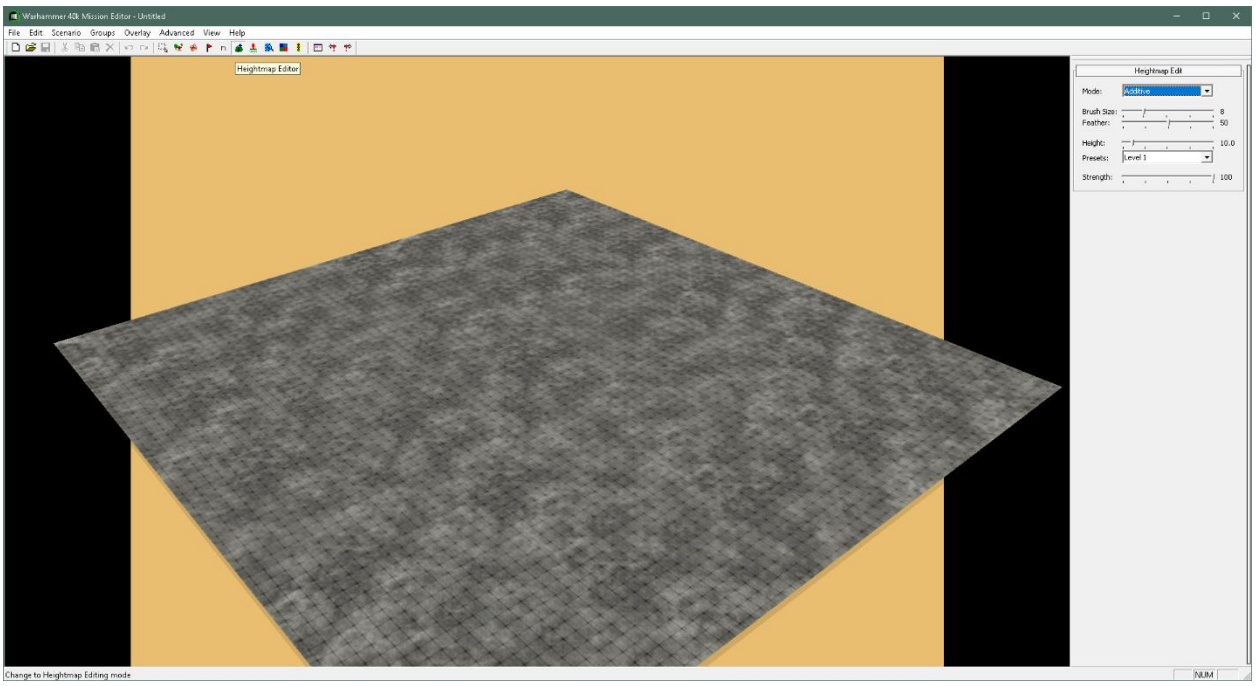


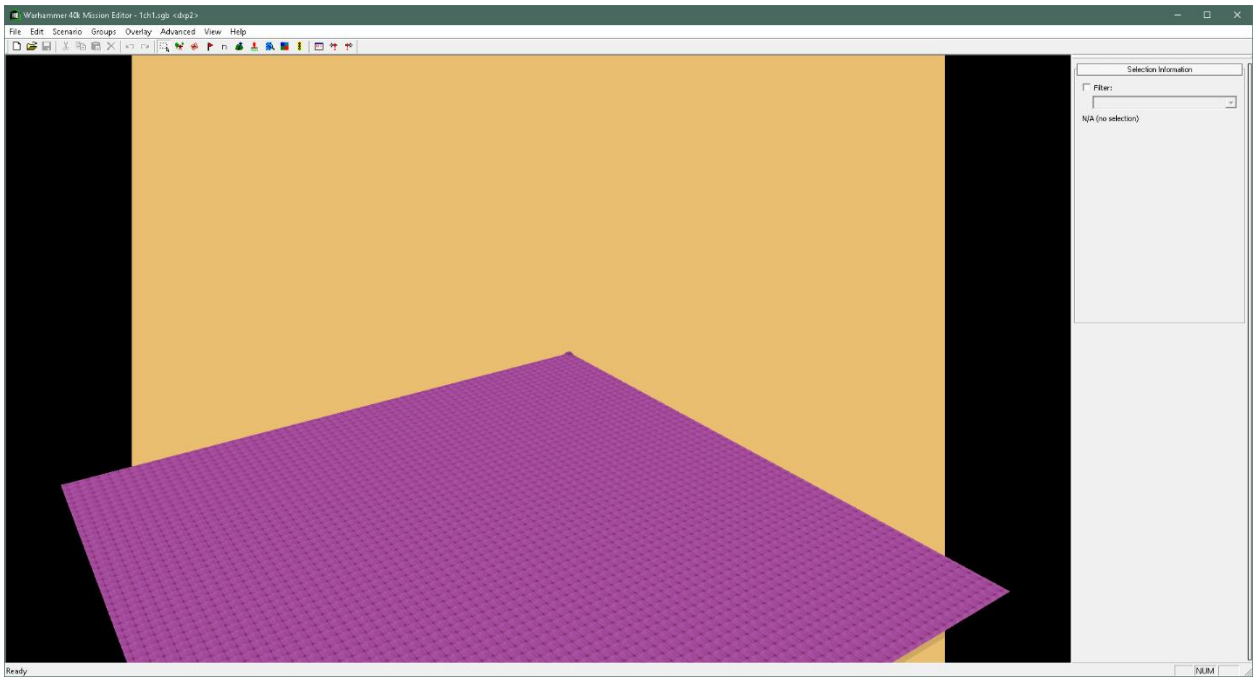
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	0123456789ABCDEF
0:0000	52	65	6C	69	63	20	43	68	75	6E	6B	79	0D	0A	1A	00	Relic Chunky....
0:0010	01	00	00	00	01	00	00	00	46	4F	4C	44	53	43	45	4E	........FOLDSCEN
0:0020	D4	07	00	00	A0	E9	04	00	00	00	00	00	44	41	54	41	Ô... é......DATA
0:0030	57	4D	48	44	D1	07	00	00	E4	02	00	00	00	00	00	00	WMHDÑ...ä.......
0:0040	02	00	00	00	01	01	00	00	04	00	00	00	64	78	70	32	............dxp2
0:0050	15	00	00	00	42	00	6C	00	6F	00	6F	00	64	00	20	00	....B.l.o.o.d. .
0:0060	52	00	69	00	76	00	65	00	72	00	20	00	5B	00	52	00	R.i.v.e.r. .[.R.
0:0070	65	00	6D	00	5D	00	20	00	28	00	32	00	29	00	4F	01	e.m.]. .(.2.).O.
0:0080	00	00	45	00	76	00	65	00	72	00	20	00	73	00	69	00	..E.v.e.r. .s.i.
0:0090	6E	00	63	00	65	00	20	00	74	00	68	00	65	00	20	00	n.c.e. .t.h.e. .
0:00A0	68	00	65	00	72	00	65	00	74	00	69	00	63	00	61	00	h.e.r.e.t.i.c.a.
0:00B0	6C	00	20	00	69	00	6E	00	68	00	61	00	62	00	69	00	l. .i.n.h.a.b.i.
0:00C0	74	00	61	00	6E	00	74	00	73	00	20	00	6F	00	66	00	t.a.n.t.s. .o.f.
0:00D0	20	00	74	00	68	00	69	00	73	00	20	00	77	00	6F	00	.t.h.i.s. .w.o.
0:00E0	72	00	6C	00	64	00	20	00	65	00	72	00	65	00	63	00	r.l.d. .e.r.e.c.
0:00F0	74	00	65	00	64	00	20	00	61	00	20	00	73	00	68	00	t.e.d. .a. .s.h.
0:0100	72	00	69	00	6E	00	65	00	20	00	74	00	6F	00	20	00	r.i.n.e. .t.o. .
0:0110	74	00	68	00	65	00	69	00	72	00	20	00	67	00	6F	00	t.h.e.i.r. .g.o.
0:0120	64	00	20	00	4B	00	68	00	6F	00	72	00	6E	00	65	00	d. .K.h.o.r.n.e.
0:0130	20	00	61	00	74	00	20	00	74	00	68	00	65	00	20	00	.a.t. .t.h.e. .
0:0140	73	00	75	00	6D	00	6D	00	69	00	74	00	20	00	6F	00	s.u.m.m.i.t. .o.
0:0150	66	00	20	00	74	00	68	00	65	00	20	00	6D	00	6F	00	f. .t.h.e. .m.o.
0:0160	75	00	6E	00	74	00	61	00	69	00	6E	00	2C	00	20	00	u.n.t.a.i.n.,. .
0:0170	74	00	68	00	65	00	20	00	72	00	69	00	76	00	65	00	t.h.e. .r.i.v.e.
0:0180	72	00	20	00	68	00	61	00	73	00	20	00	66	00	6C	00	r. .h.a.s. .f.l.
0:0190	6F	00	77	00	65	00	64	00	20	00	72	00	65	00	64	00	o.w.e.d. .r.e.d.
0:01A0	20	00	77	00	69	00	74	00	68	00	20	00	62	00	6C	00	.w.i.t.h. .b.l.
0:01B0	6F	00	6F	00	64	00	2E	00	20	00	20	00	20	00	20	00	o.o.d... . . . .
0:01C0	20	00	20	00	20	00	20	00	20	00	20	00	20	00	20	00	. . . . .
0:01D0	20	00	20	00	20	00	20	00	20	00	20	00	20	00	20	00	. . . . .
0:01E0	20	00	20	00	20	00	20	00	20	00	20	00	20	00	20	00	. . . . .
0:01F0	20	00	20	00	20	00	20	00	20	00	20	00	20	00	20	00	. . . . .
0:0200	20	00	20	00	20	00	20	00	20	00	20	00	20	00	20	00	. . . . .
0:0210	20	00	20	00	20	00	20	00	20	00	20	00	20	00	20	00	. . . . .
0:0220	20	00	20	00	20	00	20	00	20	00	20	00	20	00	20	00	. . . . .
0:0230	20	00	20	00	20	00	20	00	20	00	20	00	20	00	20	00	. . . . .

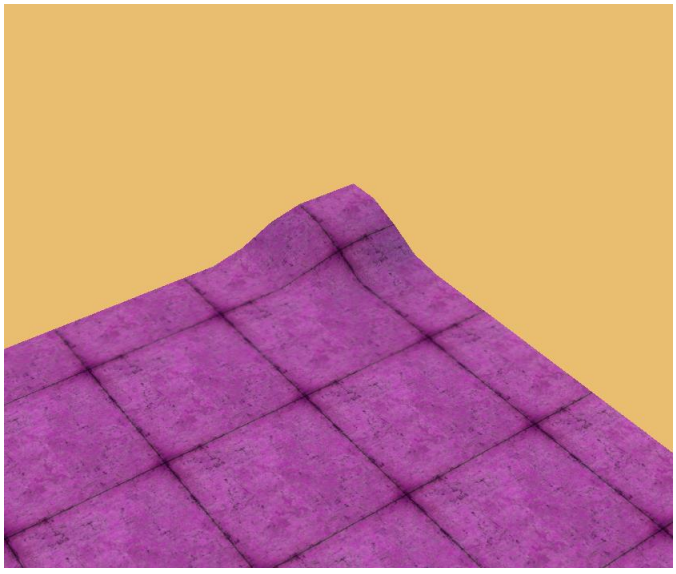
Единственное, что мы пока можем разобрать – это описание карты, идущее после ключевого слова dxp2. DXP2 – это модуль, отвечающий за локализацию в игре, значит, гипотеза, связанная с метками частично подтверждена, однако остальная информация в файле нечитабельна, поэтому нужно обратиться к другой утилите для формулировки новой теории.

Ознакомимся с редактором карт W40kME, который будет играть решающую роль при декомпиляции .sgb. Я создал две простые карты 256x256 “1.sgb” и “1ch1.sgb”, одну оставил полностью плоской, другую немного модифицировал, добавив небольшую горку в углу.

Из-за того, что на втором примере редактор не находит .tga пару у .sgb, он подсвечивает карту красным. Я сделал это специально для визуального различия, содержимое файлов от этого никак не меняется и результаты эксперимента не пострадают.


Warhammer 40k Mission Editor - Untitled
File Edit Scenario Groups Overlay Advanced View Help
Heightmap Editor
Heightmap Edit
Mode:
Additive
Brush Size:
8
Feather:
50
Height:
10.0
Presets:
Level 1
Strength:
100
Change to Heightmap Editing mode
NUM


File Edit Scenario Groups Overlay Advanced View Help
Selection Information
Filter:
N/A (no selection)
Ready
NUM



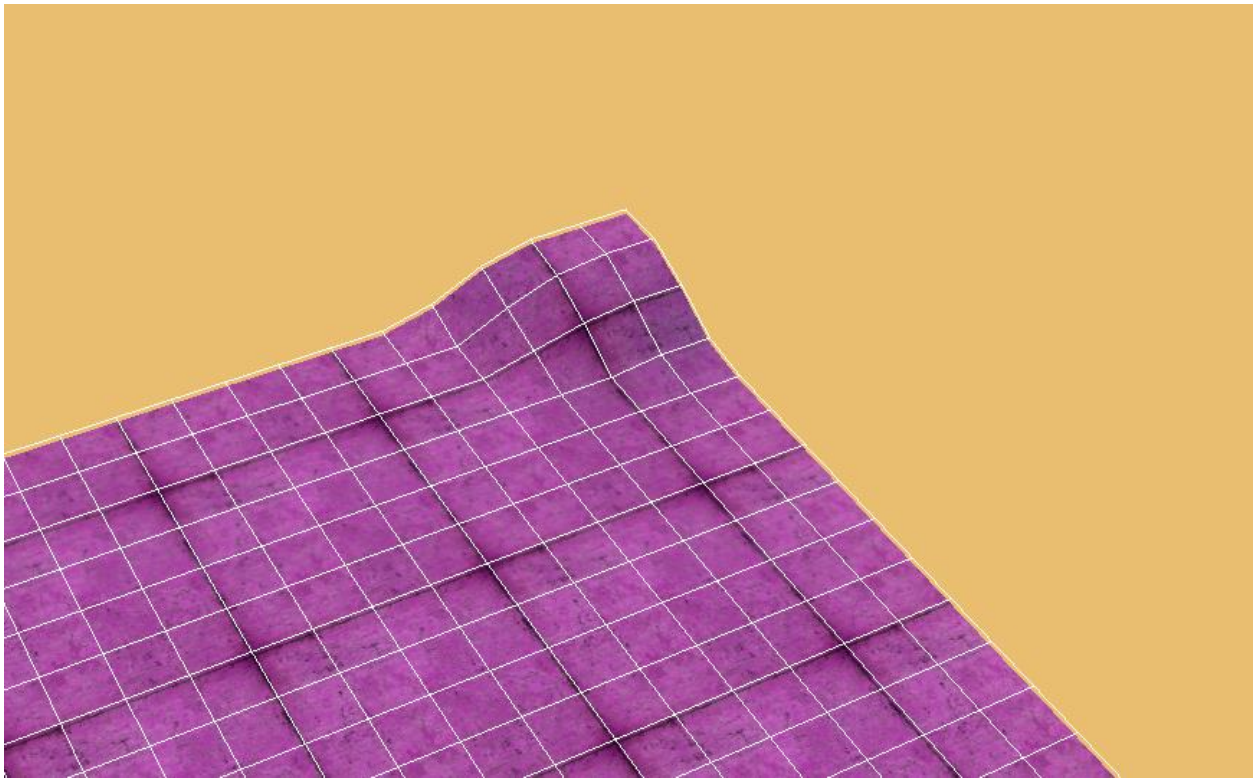
На выходе мы получаем два файла, на вид они очень похожи, их вес совпадает, но из-за созданной в углу горки должны будут наблюдаться отличия. Я написал алгоритм на Python, который проверяет каждый на наличие отличий.

```python
def hex_convert(n):
    n = hex(n)[2:].upper()

    if len(n) == 1:
        return '0' + n

    return n


def get_formatted_address(n):
    address = hex_convert(n)

    if len(address) <= 4:
        return '0:' + ('0000' + address)[-4:]

    return address[:-4] + ':' + address[-4:]


original = open(input(), mode='rb')
changed = open(input(), mode='rb')

c = 0
changes_c = 1

for obyte, cbyte in zip(original.read(), changed.read()):
    if obyte != cbyte:
        print(f'{get_formatted_address(c)}: {hex_convert(obyte)}, {hex_convert(cbyte)} - {changes_c}')

        changes_c += 1

    c += 1

original.close()
changed.close()

```

Функция hex_convert отвечает за перевод чисел в 16-ричную систему счисления в определенном формате, get_formatted_address за перевод числа в формат адреса, который представлен в 010 Editor для наглядности. Основная часть программы принимает две строки: название первого файла и второго, далее идет перебор каждого байта, если между байтами в первом и втором есть отличия, то программа выводит адрес этого байта, сами несовпадающие байты и количество изменений.

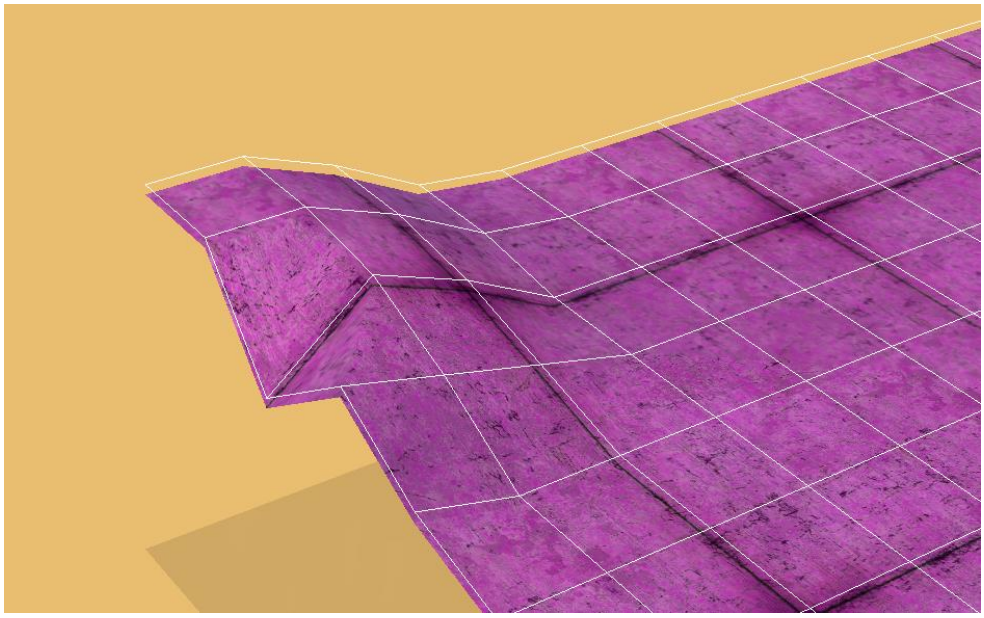
```
C:\Users\Arthur\dow-map-viewer>py comparator.py
1.sgb
1ch1.sgb
0:FE10: 28, 2A - 1
0:FF11: 28, 2E - 2
0:FF12: 28, 2C - 3
0:FF13: 28, 2A - 4
1:0012: 28, 30 - 5
1:0013: 28, 2F - 6
1:0014: 28, 2C - 7
1:0113: 28, 30 - 8
1:0114: 28, 30 - 9
1:0115: 28, 2D - 10
1:0116: 28, 29 - 11
1:0214: 28, 30 - 12
1:0215: 28, 30 - 13
1:0216: 28, 2D - 14
1:0217: 28, 29 - 15
```

Всего нашлось 15 отличий, чтобы понять причину, по которой именно эти байты отличаются, нужно снова обратиться к редактору карт и включить режим “детального изменения ландшафта”.



Если мы посчитаем количество точек, которые входят в состав горки, то есть которые находятся над нулевым уровнем, то получится, что их тоже 15. Из этого мы можем сделать вывод, что в .sgb хранятся именно координаты точки по y, именно в этих 15 точках и было отличие.

Для того, чтобы окончательно убедиться в этом, нужно изменить одну из координат в файле:



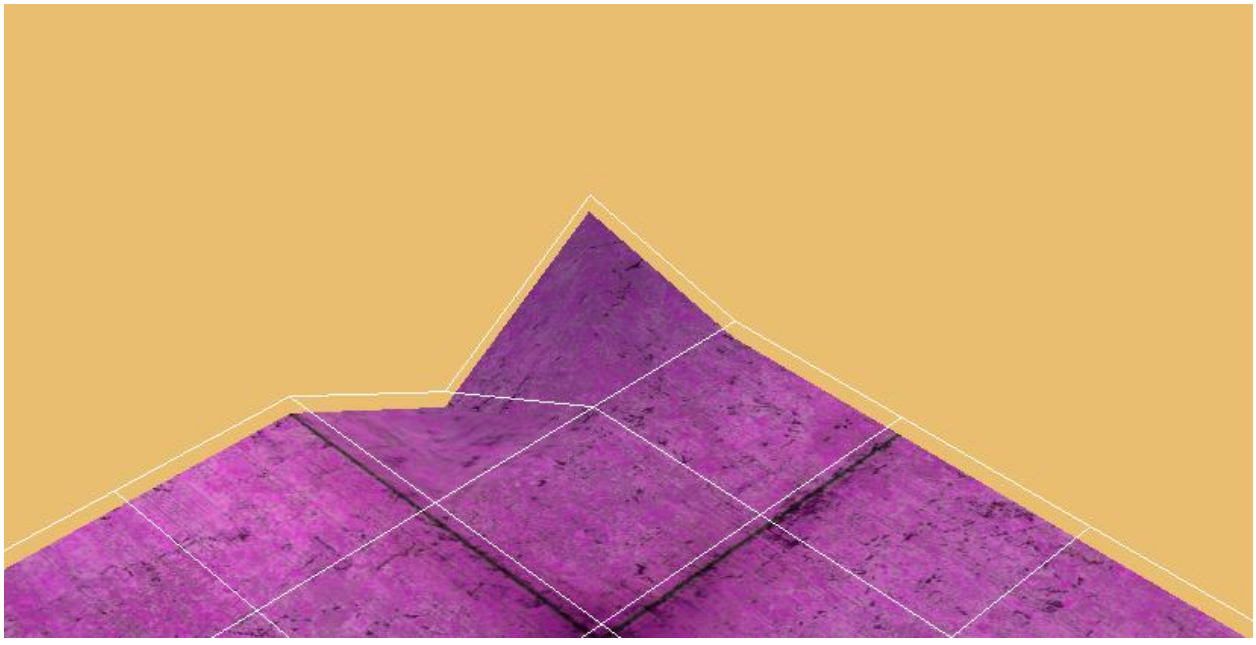
Как и ожидалось, появилась яма.

Теперь, когда мы знаем как хранится информация о точках, нужно понять где она хранится. Под “где” я имею ввиду порядок отрисовки точек, то есть первая точка, описанная в

памяти находится в правом верхнем, правом нижнем, левом нижнем или левом верхнем углу квадрата, а также под какой меткой в .sgb находится геометрия карты.

Сначала попробуем ответить на второй вопрос: если посмотреть на файл 1.sgb, то можно заметить, что после слова “DATAHMAPÐ” и 19 байтов начинается повторение байта $28 (под знаком “$” я записываю числа 16-ричной системы счисления), можно сделать предположение, что это и есть метка, описывающая геометрию карты. Для проверки гипотезы, нужно изменить первые два байта после метки.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	0123456789ABCDEF
0:0000	52	65	6C	69	63	20	43	68	75	6E	6B	79	0D	0A	1A	00	Relic Chunky....
0:0010	01	00	00	00	01	00	00	00	46	4F	4C	44	53	43	45	4E	........FOLDSCEN
0:0020	D4	07	00	00	0E	03	07	00	00	00	00	00	44	41	54	41	Ô...........DATA
0:0030	57	4D	48	44	D1	07	00	00	1E	00	00	00	00	00	00	00	WMHDÑ..........
0:0040	00	00	00	00	01	01	00	00	04	00	00	00	64	78	70	32	............dxp2
0:0050	01	00	00	00	31	00	00	00	00	00	02	00	00	00	46	4F	....1.........FO
0:0060	4C	44	57	53	54	43	D1	07	00	00	74	FC	06	00	00	00	LDWSTCÑ...tü....
0:0070	00	00	46	4F	4C	44	54	45	52	52	D0	07	00	00	43	D2	..FOLDTERRÐ...CÒ
0:0080	05	00	00	00	00	00	46	4F	4C	44	54	46	41	43	D0	07	......FOLDTFACÐ.
0:0090	00	00	26	C9	01	00	00	00	00	00	44	41	54	41	56	44	..&É......DATAVD
0:00A0	41	54	D3	07	00	00	36	00	00	00	00	00	00	00	81	00	ATÓ...6.........
0:00B0	00	00	81	00	00	00	00	00	00	40	01	00	00	00	00	00	.........@......
0:00C0	80	3E	1E	00	00	00	00	01	00	00	00	01	00	00	08	00	€>..............
0:00D0	00	00	02	00	00	00	10	00	00	00	04	00	00	00	00	00	....
0:00E0	80	3F	01	01	46	4F	4C	44	43	48	41	4E	D1	07	00	00	€?..FOLDCHANÑ...
0:00F0	63	C4	01	00	00	00	00	00	44	41	54	41	48	4D	41	50	cÄ......DATAHMAP
0:0100	D0	07	00	00	09	02	01	00	00	00	00	00	01	01	00	00	Ð...............
0:0110	01	01	00	00	2A	24	28	28	28	28	28	28	28	28	28	28	....*$((((((((((
0:0120	28	28	28	28	28	28	28	28	28	28	28	28	28	28	28	28	((((((((((((((((
0:0130	28	28	28	28	28	28	28	28	28	28	28	28	28	28	28	28	((((((((((((((((
0:0140	28	28	28	28	28	28	28	28	28	28	28	28	28	28	28	28	((((((((((((((((
0:0150	28	28	28	28	28	28	28	28	28	28	28	28	28	28	28	28	((((((((((((((((
0:0160	28	28	28	28	28	28	28	28	28	28	28	28	28	28	28	28	((((((((((((((((
0:0170	28	28	28	28	28	28	28	28	28	28	28	28	28	28	28	28	((((((((((((((((

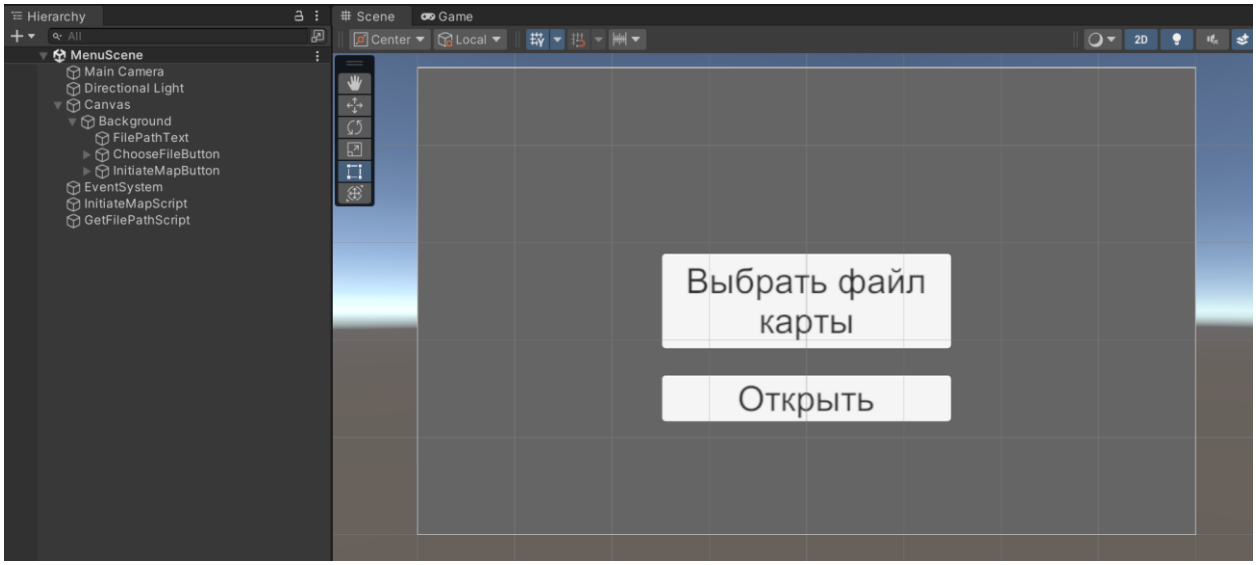


Результаты теста показывают, что был изменен левый нижний угол карты, первая точка на 2 единицы возвышена над плоскостью, а вторая опущена на 4 единицы. Можно понять, что все точки прорисовываются слева-направо и снизу-вверх.

2.2 Написание приложения для отображения карт

Для начала создадим новый 3D-проект и 2 сцены: MapScene и MenuScene, в первой будет меню выбора файла, а во второй – сама карта.

Я наверстал небольшое окно:



На Canvas'е расположены 3 объекта: текст, который будет хранить путь к файлу пользователя, кнопка выбора файла и кнопка открытия этого файла. Также на сцене расположены "пустышки" с прикрепленными к ним скриптами.

Перейдем к коду:

Assets > Scripts > GetFilePath.cs

```
1   using System.Collections;
2   using System.Collections.Generic;
3   using UnityEngine;
4   using UnityEngine.UI;
5
6   public class GetFilePath : MonoBehaviour
7   {
8       public Text FilePathText;
9
10      public void InitClick()
11      {
12          NativeFilePicker.Permission permission = NativeFilePicker.PickFile((path) =>
13          {
14              if (path != null)
15              {
16                  StaticData.mapPath = path;
17                  FilePathText.text = path;
18              }
19          }, new string[] { NativeFilePicker.ConvertExtensionToFileType("sgb") });
20      }
21  }
22
```

GetFilePath отвечает за добычу пути к .sgb. В данном примере я использую плагин NativeFilePicker (https://github.com/yasirkula/UnityNativeFilePicker) для удобной работы с файлами на Android. На этапе инициализации переменной permission у пользователя запрашивается доступ на считывание содержимого его памяти, потом асинхронно вызывается мое лямбда-выражение, где статическому полю mapPath из класса StaticData присваивается путь, полученный от пользователя. Потом меняется текст объекта FilePathText. Вторым

аргументом в метод PickFile передается строковый массив из типов, которые может выбрать пользователь.

Assets > Scripts > InitiateMap.cs

```
1   using System.Collections;
2   using System.Collections.Generic;
3   using UnityEngine;
4   using UnityEngine.SceneManagement;
5
6   public class InitiateMap : MonoBehaviour
7   {
8       public void Initiate()
9       {
10          if (StaticData.mapPath != null)
11          {
12              SceneManager.LoadScene("MapScene");
13          }
14      }
15  }
16
```

InitiateMap отвечает за смену сцены в случае, если пользователь выбрал путь.

Теперь воспользуемся знаниями, которые были приобретены экспериментальным путем в прошлом параграфе: мы знаем, что в файле присутствуют заголовки, они говорят нам, где начинается и заканчивается определенная секция в файле. DATAHMAPÐ – метка, обозначающая начало геометрии карты, а DATATTYPÒ – начало новой секции или же конец нашей.

```
7   public class DrawMap : MonoBehaviour
8   {
9       void Start()
10      {
11          byte[] mapData = File.ReadAllBytes(StaticData.mapPath);
12
13          byte[] geometryHeader = {68, 65, 84, 65, 72, 77, 65, 80, 208, 7};
14          byte[] geometryEnd = {68, 65, 84, 65, 84, 84, 89, 80, 210, 7};
15
16          bool readingMode = false;
17
18          int p = 0;
19
20          List<byte> geometryData = new List<byte>();
21
```

Считываем всю информацию из файла пользователя в байтовом виде и записываем ее в массив байтов mapData, geometryHeader и geometryEnd отвечают за хранение DATAHMAPÐ и DATATTYPÒ в байтовом виде, флаг readingMode будет указывать программе на то, можно ли приступить к чтению, p (pointer) выступает в качестве индекса для перебора mapData, а список geometryData будет хранить в себе заветную геометрию.

```
while (true)
{
    if (!readingMode)
    {
        if (mapData[p] == geometryHeader[0] &
            mapData[p + 1] == geometryHeader[1] &
            mapData[p + 2] == geometryHeader[2] &
            mapData[p + 3] == geometryHeader[3] &
            mapData[p + 4] == geometryHeader[4] &
            mapData[p + 5] == geometryHeader[5] &
            mapData[p + 6] == geometryHeader[6] &
            mapData[p + 7] == geometryHeader[7] &
            mapData[p + 8] == geometryHeader[8] &
            mapData[p + 9] == geometryHeader[9])
        {
            readingMode = true;

            p += 27;
        }
    }

    else
    {
        if (mapData[p] == geometryEnd[0] &
            mapData[p + 1] == geometryEnd[1] &
            mapData[p + 2] == geometryEnd[2] &
            mapData[p + 3] == geometryEnd[3] &
            mapData[p + 4] == geometryEnd[4] &
            mapData[p + 5] == geometryEnd[5] &
            mapData[p + 6] == geometryEnd[6] &
            mapData[p + 7] == geometryEnd[7] &
            mapData[p + 8] == geometryEnd[8] &
            mapData[p + 9] == geometryEnd[9])
        {
            break;
        }

        geometryData.Add(mapData[p]);
    }

    p++;
}
```

В этом цикле мы записываем информацию о карте после DATAHMAPĐ и выходим из него после DATATTYPÒ. К р прибавляется 27, а не 10, потому что после метки присутствует еще ненужных нам 17 байтов.

Перейдем к отображению карты. 90% игровых локаций изображены в виде квадратов, поэтому работаем с этой формой. Чтобы узнать количество точек в одном ряду, нужно вычислить арифметический корень из числа байтов в mapData. Далее нужно определиться со способом отображения “плиток”, из которых и состоит карта, это можно реализовать при помощи треугольников (3 точки в пространстве всегда лежат на плоскости, а значит образуют треугольник). Нужно пройти в цикле все индексы, от нуля до количества байтов минус корень. Далее соединять каждую i-ную, i+1, i+корень точку и i+1, i+корень, i+корень+1 в треугольники с использованием класса Mesh и компонента MeshFilter, при этом исключая варианты, где i + 1 делится на корень нацело, и все, осталось только подобрать правильное положение для камеры и нажать на Build and Run, предварительно установив “режим альбома”, чтобы экран был зафиксирован и не переворачивался.

Заключение

Существует множество причин, по которым я выбрал именно эту тему для проекта, но, скорее всего, главной из них является то, что я всегда желал посмотреть на мир, который скрывается под абстракцией современного программирования. Из-за отсутствия проблем с производительностью сейчас спокойно можно тренировать нейронные сети на таких языках, как Python, низкоуровневые ЯП-ы очень быстро теряют свою популярность, сайты создаются одним кликом правой кнопки мыши, почти все игры пишутся на двух игровых движках, поэтому что может помочь удовлетворить интерес к аскетичной разработке лучше, чем взгляд на архитектурные решения, которые предлагали ведущие геймдизайнеры, допустим, 20 лет назад

Рецензия на проектную работу ученика

10 класса ЧОУ «Первая гимназия Максимум»

Шияна Артура

на тему: «

Создание приложения для просмотра файлов карт игры “Warhammer 40000 Dawn of War Soulstorm” на мобильные устройства

»

С актуальностью данной работы м поспорить, ведь не все играют в компьютерные игры, но суть работы совсем не в этом. А в том, что ученик 10 класса программирует на достойном уровне. А так же имеет свою позицию в познании языков и IT сферы целиком

Материал, который ученик использует в проекте, выходит далеко за рамки школьного, что является следствием интереса и самостоятельного изучения всевозможной литературы по данной теме.

Оценивая работу в целом, стоит отметить хорошую структуру проекта, работа соответствует плану и поставленным задачам, логически продумана. Оформление работы и культура письма обеспечивают понимание изложенного материала.

Оформление работы: работа оформлена в полном соответствии с предъявляемыми требованиями.

Работа, представленная для рецензирования, является самостоятельным проектом, выполнена на высоком теоретическом и практическом уровне, заслуживает положительной оценки.

29.12.2024 г. Фахретдинова Л,А.