*Opis mojej części do dokumentacji (Piotrek)*

opis plików:

„StartMenu.cs” – skrypt startowy programu, rozpoczyna pracę od wyświetlenia menu wyboru mapy.

„Form1.cs” – skrypt zawiera klasę Form1 tworzącą menu wyboru mapy. Klasa zawiera wszelkie potrzebne funkcje do odczytu pożądanych danych z plików graficznych zawierających mapy wysokościowe terenu.

„MyTerrain.cs” – skrypt zawiera klasę MyTerrain odpowiedzialną za generację trójwymiarowego terenu w Unity na podstawie danych odczytanych z mapy wysokościowej terenu.

„PerlinNoise.cs” – skrypt zawiera zbiór algorytmów szumu wykorzystanych przy losowym wstawianiu drzew i innych detali na utworzonym terenie.

Najważniejsze klasy i funkcje programu:

## class StartMenu

Klasa jest przypisana do pustego obiektu *„Start Object”* na startowej scenie projektu. Program w fazie uruchomieniowej wywołuję metodę „*Start()”* tej klasy, w której jest tworzony nowy obiekt klasy Form1 implementujący menu wyboru mapy.

Klasa dodatkowo zawiera zmienną całkowitą „*m\_mapSize”* i tablicę „*m\_grayLevels”*, do których po wyborze mapy wysokościowej mają zostać zapisane rozmiar mapy i wysokości poszczególnych pixeli w postaci poziomu szarości w przedziale <0.0 ; 1.0>.

W metodzie „*Start()”* klasy dodatkowo jest wywoływana funkcja *„Object.DontDestroyOnLoad(this)”*, aby obiekt, do którego jest przypisany nie został usunięty, aby nie utracić odczytanych danych z mapy.

## class Form1

Klasa jest odpowiedzialna za tworzenie menu wyboru mapy. Zawiera wszelkie potrzebne funkcje do odczytu pożądanych danych z plików graficznych zawierających mapy wysokościowe terenu.

Klasa umożliwia odczyt danych z map wysokościowych w formatach grafiki rastrowej:

1. \*.jpg, \*.jpeg, \*.bmp, \*.png, \*.gif, \*.tiff
2. oraz surowym \*.raw

Mapy mogą być zarówno czarno-białe, jak i kolorowe, gdyż odczytane dane o kolorze piksela w modelu RGB przeliczamy na informację o jasności koloru (poziom szarości). Należy jednak wziąć pod uwagę, że nie będzie w takim przypadku rozróżniane przypisanie na mapie wysokości do różnych kolorów (zielony, czerwony, niebieski), a jedynie do jasności koloru. Wymogiem natomiast jest, aby wysokość i szerokość mapy były jednakowe i rozmiar ten był potęgą dwójki. Ograniczenia te zostały dodane, gdyż Unity takie ograniczenia narzuca, aby poprawnie zostały ustawione wysokości terenu na podstawie przekazanych danych.

Wybór mapy i odczyt danych następuje po wywołaniu metody *„int readHeightmap(string fileName)”*.

Do odczytania plików rastrowych wykorzystano klasy „*Bitmap”* i *„Pixel”* oraz ich metody udostępnione w bibliotekach języka C#. Do odczytu jasności pixela wykorzystano metodę *„GetBrightness()”*, która zwraca już gotową informację o jasności pixela.

Do plików w formacie surowym \*.raw trzeba było podejść w inny sposób. Format ten przechowuje w sposób bezpośredni bajt, po bajcie (a dokładniej 1,5 bajta po 1,5 bajcie) kolory pikseli, jednak nie posiada żadnego nagłówka z dodatkowymi informacjami i klasa Bitmap nie oferuje możliwości pobrania danych z obrazka w takim formacie. Dane z pliku \*.raw program przepisuje bezpośrednio do bufora bajtowego, a następnie jasność piksela jest przeliczana z jego składowych RGB przy pomocy wzoru zaczerpniętego z Internetu (na końcu dodatkowo dzielenie przez 255, aby uzyskać jasność w skali <0.0 ; 1.0>):

m\_grayLevels[x, y] = (float)((buffer[bufferIdx] \* 0.3) + (buffer[bufferIdx + 1] \* 0.59) + (buffer[bufferIdx + 2] \* 0.11)) / 255;

W obu przypadkach w rezultacie zwracane są takie same (a przynajmniej prawie takie same) dane dla dwóch takich samych obrazów zapisanych w innym formacie.

Odczytane dane można również zapisać do pliku tekstowego i z takiego pliku je później odczytać.

## class MyTerrain

Klasa jest odpowiedzialna za generację trójwymiarowego terenu w Unity na podstawie danych odczytanych wcześniej z mapy wysokościowej terenu.

Sporo miejsca by zajęło opisanie wszystkich funkcjonalności użytych do doprecyzowania generowanego terenu, ale warto nadmienić co najważniejsze. W Unity generacja terenu z poziomu skryptu składa się z dwóch części, skonfigurowanie pożądanego terenu z wykorzystaniem odpowiednich metod klasy *„TerrainData”* oraz utworzenie terenu przy pomocy klasy *„Terrain”* na podstawie skonfigurowanych danych w klasie *„TerrainData”*.

Aby ustawić pożądane wysokości terenu, potrzebne dane odczytane z mapy wysokościowej terenu są pobierane z obiektu *„Start Object”*, w którym wcześniej je zapisano i przekazywane do metody *„SetHeights(...)”* klasy *„TerrainData”*.