3D Map Visualization - documentation

1 BuilderScript.cs

1.1 Blok

```
public Vector2[] bresenhamLine(Vector3 a, Vector3 b)
```

Takes two points returns (x,y) and returns rasterized line between them(array of points). Uses Bresenham line rasterization algorithm.

1.1.1 Riadok

```
bool steep = false;
```

iterate x, calculate y

1.1.2 Riadok

```
if (dy > dx)
```

iterate y, calculate x

1.2 Blok

```
private int[] swapInts(int a, int b)
```

Takes two integers and returns an array with them swapped in order.

1.3 Blok

```
public int[][][] drawContours(Vector2[][] contours, int height,
    int width)
```

Takes and array of rasterized contours, total width and height of contour map and marks them into a 3 dimensional array. First two dimensions represent y and x coordinates of points and the third serves for storing multiple overlapping contours. Returns this array.

1.3.1 Riadok

```
int[][][] ret = new int[height][][];
```

3 contours can overlap

1.4 Blok

```
 y++)
```

Horizontal pass

Jakub Motýľ Martin Miklis, Timotej Jurášek

1.5 Blok

```
 x++)
```

```
 Horizontal pass
```

1.6 Blok

```
    public int[][] scanline(int[][][] contours)
```

```
 Scans through drawn contour array horizontally and vertically, averaging
the values, and marks relative height level for every field according
to number of contours crossed on the way. 0 marks lowest level.
```

1.7 Blok

```
if (vstup.Length < y && vstup[0].Length < x)
```

```
pre pripad , ze je mensie pole nez vystup
```

1.8 Blok

```
if (a * interval + i < vstup.Length && b * interval + j < vstup
    [0].Length)
```

```
ak mi nestacia policka povodnych, ratam ich ako nuly
```

1.9 Blok

```
 a++)
```

```
naplnim vsetky policka
```

1.10 Blok

```
public float[][] sampleQuantization(int[][] vstup, int x, int y)
```

```
inicializacia
```

1.10.1 Riadok

```
int maxsize = Math.Max(vstup[0].Length, vstup.Length);
```

```
ak mame velke pole a potrebujeme zmensit
```

1.10.2 Riadok

```
return vystup;
```

```
vratim vyplnenu tabulku
```

2 MyTerrain.cs

2.1 Blok

```
public void load(List<Vector3[]> contours)
```

Loads cubic bezier representation of all contours in the map

2.2 Blok

```
public Vector3[][] getApproximatedContours(int detail)
```

Returns polygonal path representation of all contours in the map approximated with (detail-1) being number of approximated points on each curve. Uses Bernstein basis polynomial explicit definition as approximation strategy by setting t values

2.3 Blok

```
private int combinatorialNumber(int n, int k)
```

Returns value of combinatorial number n over k

2.4 Blok

```
private int factorial(int n)
```

Returns value of factorial n

3 Parser.cs

3.1 Blok

```
j++)
```

from parent element parameter count we know number of coord elements

3.1.1 Riadok

```
ret[0] = Mathf.Min(ret[0], x);
```

min x

3.1.2 Riadok

```
ret[2] = Mathf.Min(ret[2], y);
```

min y

3.1.3 Riadok

```
ret[1] = Mathf.Max(ret[1], x);
```

max x

3.1.4 Riadok

```
ret[3] = Mathf.Max(ret[3], y);
```

max y

3.1.5 Riadok

```
contour.Add(new Vector3((float)x,(float)y,0));
```

add x and y values to contour

3.2 Blok

```
i++)
```

from previous parent objects parameter count we know number of object elements

3.2.1 Riadok

```
ix = findFrom("</coords>", ix);
```

close element coords

3.2.2 Riadok

```
ix = findFrom("</object>", ix);
```

close element object

3.2.3 Riadok

```
clist.Add(contour.ToArray());
```

add contour from last object's coords to array of contours

3.3 Blok

```
if (ptext[i + j] != keyword[j])
```

if chars are not equal, then test is false

3.4 Blok

```
j++)
```

compare actual position with keyword

3.5 Blok

```
if (test)
```

if test is not false, then return index after keyword

3.6 Blok

```
public int findFrom(string keyword, int fromIx)
```

Finds keyword from index in string ptext and returns index after last char in keyword from ptext.

3.6.1 Riadok

```
return -1;
```

if keyword is not found, then return -1

3.7 Blok

```
while (fromIx + i < ptext.Length && ints.IndexOf(ptext[fromIx +
    i]) != -1)
```

while char on index i is from valid chars do concatenation

3.8 Blok

```
public int readInt(int fromIx)
```

Reads integer from index in ptext string and returns it as int.

3.8.1 Riadok

```
string ints = "-0123456789";
```

Valid chars.

3.8.2 Riadok

```
return Convert.ToInt32(strInt);
```

return int from concatenation of valid chars

4 TerrainFill.cs

4.1 Blok

```
public void FillTerrain(float[][] input, TerrainData tData,
    Terrain myTerrain, int xBase, int yBase)
```

Fills terrain with values from scanline and quantization, but it must be divided by large number. Terrain needs float values between 0 and 1.

5 UIManager.cs

5.1 Blok

```
void Start ()
```

Initialization of type.

5.2 Blok

```
void OnLevelWasLoaded()
```

When main scene is loaded this will get path value.

5.3 Blok

```
public void LoadMapButton()
```

Function for load map button. Changes scene to filemanager scene.

5.4 Blok

```
public void LoadTrackButton()
```

Function for load track button. Changes scene to filemanager scene.

5.5 Blok

```
public void LoadMap(string path)
```

Integration function. When file was selected, this function is called and it runs all methods from project.

5.5.1 Riadok

```
parser.loadStringFromFile(path);
```

Calling parser with path.

5.5.2 Riadok

```
int[][][] drawnContours = builder.drawContours(rasterized,
    height, width);
```

Draw rasterized contours to two-dimensional array.

5.5.3 Riadok

```
int[][] scanlined = builder.scanline(drawnContours);
```

Run scanline and check height between contours.

5.5.4 Riadok

```
float[][] res = builder.sampleQuantization(scanlined, 65, 65);
```

Quantize huge two-dimensional array into smaller 65x65 for terrain input

5.5.5 Riadok

```
terrain.FillTerrain(res, tData, myTerrain, xBase, yBase);
```

Fill terrain with input.

6 ViewController.cs

6.1 Blok

```
if (pos.magnitude > 0.0001f)
```

interpolate camera position, if needed

6.2 Blok

```
if ((pppos - gotoPosition).magnitude > 0.0001f)
```

interpolate camera position, if needed

6.3 Blok

```
if (Input.GetMouseButton(0))
```

if left mouse button, set new pivot position

6.3.1 Riadok

```
gotoPosition += worldMouseVector * MoveSpeed * distanceFactor *
    -1;
```

inverted axees

6.4 Blok

```
 else if (Input.GetMouseButton(1))
```

if right mouse button, adjust target rotation of pivot

6.5 Blok

```
if (pos.magnitude >= ZoomCap || mouseWheel < 0f)
```

dont reverse the objects, cap max zoom

6.6 Blok

```
 else if (Mathf.Abs(mouseWheel) > 0.0001f)
```

if mouse wheel is scrolling, adjust camera-pivot zoom

6.7 Blok

```
void FixedUpdate()
```

On every physics calculation, take input and readjust camera and its pivot

6.7.1 Riadok

```
Vector3 mouseVector = new Vector3(Input.GetAxis("Mouse X"),
    Input.GetAxis("Mouse Y"));
```

get screen mouse vector

6.7.2 Riadok

```
Vector3 worldMouseVector = pivotPoint.transform.
    TransformDirection(mouseVector);
```

convert into world coordinates

6.7.3 Riadok

```
float mouseWheel = Input.mouseScrollDelta.y;
```

get mouse scroll input

6.7.4 Riadok

```
float rot = transform.rotation.eulerAngles.z;
```

stabilize z axis rotation

6.8 Blok

```
public class ViewController : MonoBehaviour
```

MoveSpeed, RotateSpeed and ZoomSpeed can be floats in range (0..infinity>, ZoomFactor can be float in range (0..1)

7 RunTestsScript.cs

7.1 Blok

```
void Start()
```

Use this for initialization some tests may need to have readjusted protection levels for methods in order to run