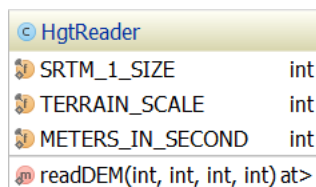


UML Documentation

The code of the system is organized in three packages: demtools, renderengine, comoflyer.

1. Demtools

This packages is dedicated to the tools working with DEM (Digital Elevation Model). The package contains only one class HgtReader which facilitates access to and manipulations with .hgt SRTM DEM files. The class is essentially a container for static methods since it has no non-static fields and methods and no constructors.



HgtReader	
SRTM_1_SIZE	int
TERRAIN_SCALE	int
METERS_IN_SECOND	int
readDEM(int, int, int, int)	at>

Figure 1: HgtReader class structure.

The class has 3 public fields intended to be used outside of the class:

- **SRTM_1_SIZE** integer equals to 3601 – the resolution of SRTM DEM file is 3601 values per one degree.
- **METERS_IN_SECOND** integer equals to 30 – a derivative from the previous values. Defines number of meters in one arc-second.
- **TERRAIN_SCALE** equals to 5 – a coefficient used to scale the terrain when it is loaded to the 3D scene. It makes no sense in changing it since the whole scene depends on it and rescales automatically. Although this scaling increases performance a bit.

The class has only one public method **readDEM** to be used outside of the class. The method receives geographic coordinates (latitude integral part, latitude fractional part in arc-seconds, longitude integral part, longitude integral part in arc-seconds) of the desired centre of the terrain and returns a pair of values: float array of heights representing the loaded terrain and one float value which represents the height of the centre of the terrain.

2. Renderengine

This package contains three classes: ReadableDepthRenderer, SuperRenderer, OffscreenSceneProcessor and TextureUtil. The class-hierarchy of the ReadableDepthRenderer and SuperRenderer can be seen on the Figure 2.

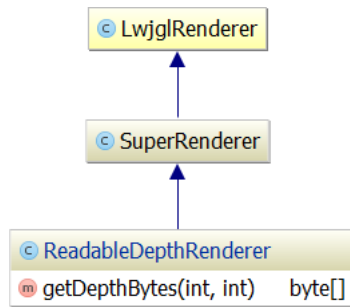


Figure 2: ReadableDepthRenderer and SuperRenderer class hierarchy.

The SuperRendered is a complete copy of the `com.jme3.renderer.lwjgl.LwjglRenderer` from the `jme3` library which it extends. The only difference and the only reason it exists is the `private int maxFBOAttachs = 1;` set explicitly at the beginning of the file and `//maxFBOAttachs = intBuf16.get(0);` commented at the middle of the file. This overcomes the “Framebuffer has more color attachments than are supported by the video hardware!” exception thrown at the off-screen rendering mode.

The `ReadableDepthRenderer` extends `SuperRenderer` with the **getDepthBytes** method. The method accepts the height and width of the screen as in input and returns the byte array which represents the depth content of the z-buffer.

The `TextureUtil` is a complete copy of `com.jme3.renderer.lwjgl.TextureUtil` with some methods made public to be called by `SuperRenderer`.

The `OffscreenSceneProcessor` implements `com.jme3.post.SceneProcessor` interface. It contains a field of type `OffscreenComoFlyer`. The class implements all methods with empty ones except the **postFrame** where it calls **updateImageContents()** method of the `OffscreenComoFlyer` to update the image content after the scene has been rendered to the GPU’s framebuffer.

3. Comoflyer

This package contains the main functionality within 6 classes: `GeoApplication`, `OnscreenComoFlyer`, `OffscreenComoFlyer`, `StaticHelpers`, `StaticDepthHelpers`, `Main`.

The class-hierarchy of the `GeoApplication`, `OnscreenComoFlyer` and `OffscreenComoFlyer` can be seen on the Figure 3.

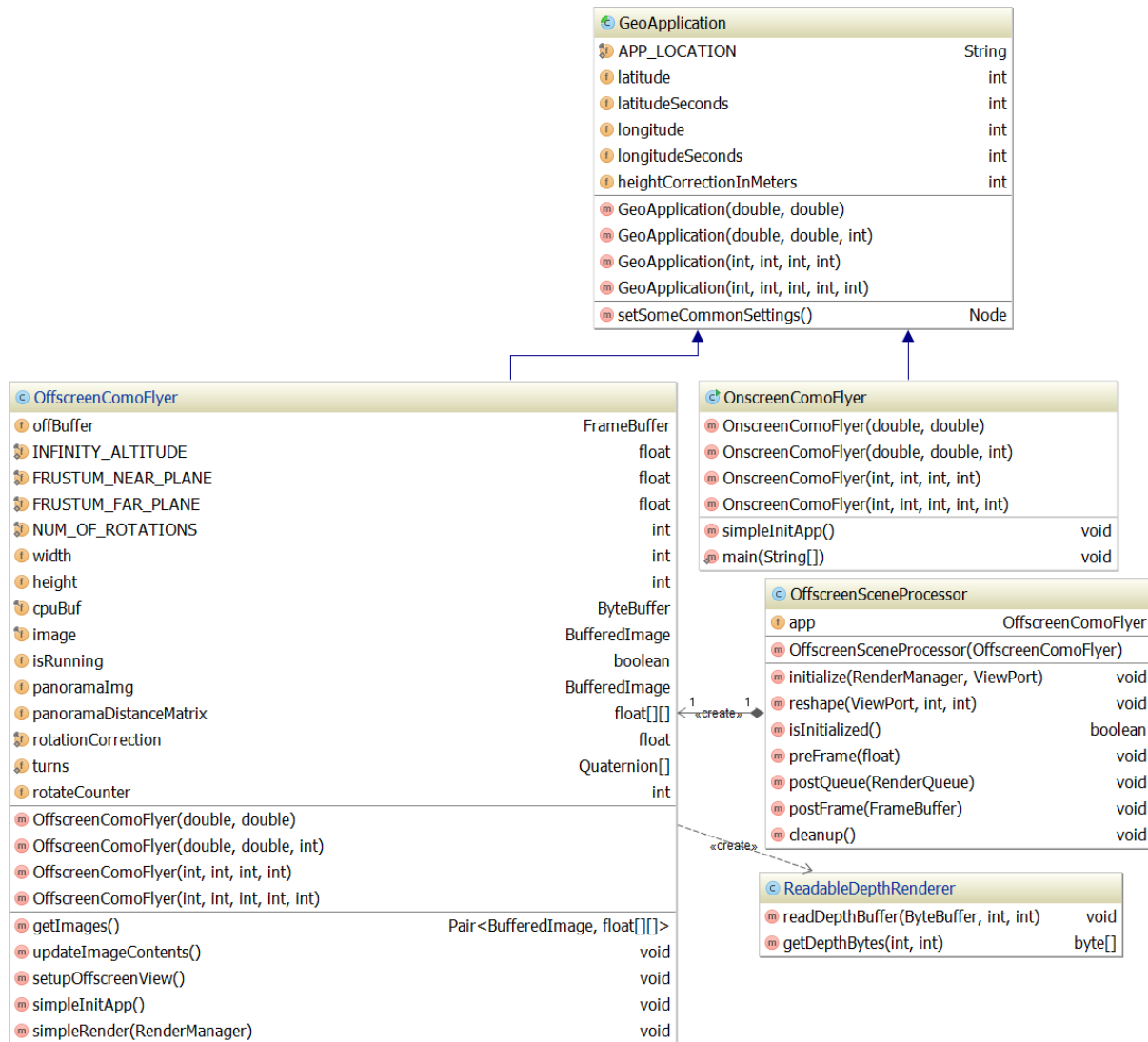


Figure 3: GeoApplication, OnscreenComoFlyer and OffscreenComoFlyer class hierarchy.

The GeoApplication abstract class extends `com.jme3.app.SimpleApplication` by adding constructors with geographical coordinates common for all successors.

The OnscreenComoFlyer class is the simplest successor which implements the application that performs rendering to the physical screen of the computer and allows flying over the terrain using mouse and keyboard as controls.

The OffscreenComoFlyer class performs off-screen rendering to the image of the 360° panorama. There are two public methods exposed:

The **updateImageContents** method is intended to be called from the **OffscreenSceneProcessor.postFrame** method to notify the OffscreenComoFlyer entity that the scene has been rendered and the content is available in the framebuffer. 360° panorama is created in a number of rotations. After each rotation OffscreenComoFlyer is notified and performs copying of the bytes from framebuffer to later process and add them to the whole panorama image.

The **getImages** method returns a pair of BufferedImage and float[][] matrix. BufferedImage is an 360° panorama image. float[][] matrix contains the distance values in meters from the virtual camera to every pixel of the panorama.

StaticHelpers		StaticDepthHelpers	
getSky(AssetManager)	Spatial	flip(BufferedImage)	void
getSun()	DirectionalLight	writeOneMatrixIntoAnother(float[][], float[][], int, int)	void
getLight()	DirectionalLight	getDepthImage(float[][])	BufferedImage
createTerrain(AssetManager, int, int, int, int) >		arrayToIntArrayWithInfinity(float[][])	int[][]
heightFromDistance(float[][], double, int) at[][]		intArrayToImage(int[][])	BufferedImage
computeAnglesFrustumUpperPart(int, double)		normalize(float, float, float, float, float)	float
		normalize(double, double, double, double, double)	double
		normalize(float[][], float, float, float, float)	float[][]
		linearize(float, float, float)	float
		linearize(float[][], float, float)	float[][]
		saveToCsv(float[][], String)	void
		toFloatArr(byte[])	float[]
		toMatrix(float[], int)	float[][]
		flip(float[][])	float[][]
		getMinMax(double[])	double[]
		getMinMax(float[][])	float[]

Figure 4: StaticHelpers and StaticDepthHelpers methods.

StaticHelpers and StaticDepthHelpers classes are simply containers of static methods. Most methods are simple and self-describing. The most important and unclear ones are described below:

The **heightFromDistance** method converts distance mask into the height mask taking into account current vertical view angle of the virtual camera and the altitude of the camera.

The **normalize** method rescales the value considering the new min and max values provided as input.

The linearize method linearizes the depth value from z-buffer (initially it's not linear).

The Main class is used to illustrate the way of usage of the OffscreenComoFlyer to create and save the panorama together with depth mask and height mask.