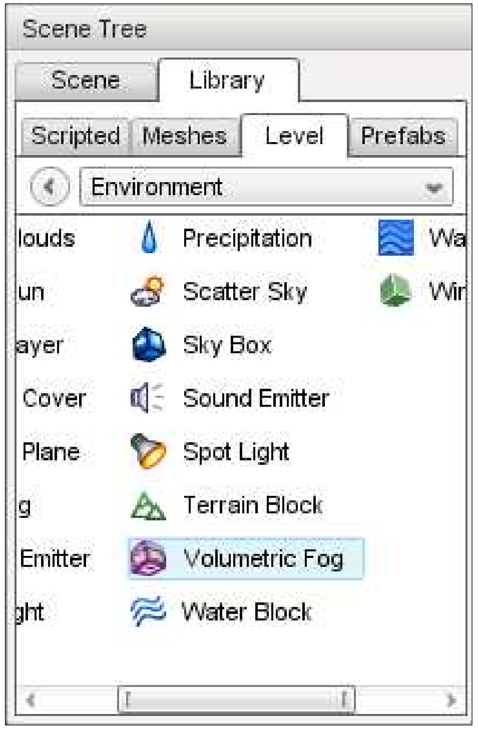
## Adding a VolumetricFog object to your mission.

If you have followed the instructions from the Installation Guide, you should be able to add VolumetricFog objects to your missions using the World-editor. You can find the VolumetricFog object in the Scene Tree under **Library->Level->Environment**.



Now double-click on the **Volumetric Fog** item to open a dialog, where you can choose a name, shape, scale and color for the new VolumetricFog object.

*Remarks:*
- *The shape can be a .dts or a .dae (COLLADA) model file. If there are different levels of detail (LOD) in the dae file, then this file must be accompanied with the TSShapeConstructor script file (as generated when importing a .dae file in the Shape editor), otherwise all LODs will be added to the same detail level.*
- *It is recommended to use models with only 1 material added to it, because the VolumetricFog object will only load the first mesh available per detail level.*
- *Remember to use colors with RGB-values from 0-255. The A(lpha)-value is ignored.*
- *There are, for your convenience, a few models added to the pack. They can be found in the art/environment folder. They are all single LOD models except for the LightVolume_Sphere model.*

If all has gone well, the new VolumetricFog object is added to your scene.

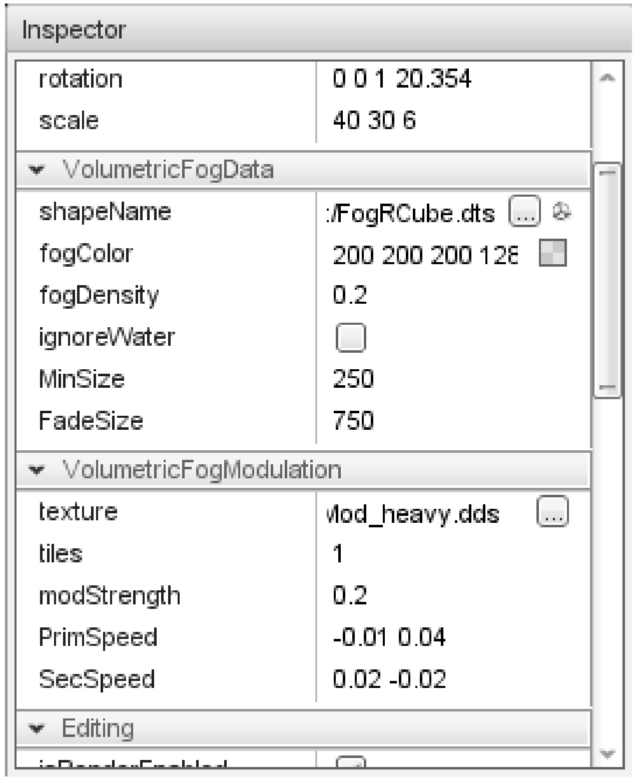
## Editing a VolumetricFog object in your scene.

Select a VolumetricFog object in the scene to view it in the Inspector. There are 2 parts that are specific for the VolumetricFog object, VolumetricFogData and VolumetricFogModulation. VolumetricFogData contains the basic data for rendering a VolumetricFog object, while the VolumetricFogModulation part contains the data for the modulation and animation of the VolumetricFog.

VolumetricFogData contains the following data:

- shapeName: the name of the .dts or .dae model file used as volume (see also previous remarks),
- fogColor: the overall color of the fog (see also previous remarks),
- fogDensity: the overall density of the fog (0 disables the rendering of the fog),
- ignoreWater: if set then the fog will also be rendered when submerged,
- MinSize: The smallest size (in pixels) where the fog will be rendered (LOD),

- FadeSize: Size in pixels from where the density and the modulation starts to fade (LOD).



Inspector

| | |
|---|---|
| rotation | 0 0 1 20.354 |
| scale | 40 30 6 |
| **VolumetricFogData** | |
| shapeName | :/FogRCube.dts |
| fogColor | 200 200 200 128 |
| fogDensity | 0.2 |
| ignoreWater | ☐ |
| MinSize | 250 |
| FadeSize | 750 |
| **VolumetricFogModulation** | |
| texture | Mod_heavy.dds |
| tiles | 1 |
| modStrength | 0.2 |
| PrimSpeed | -0.01 0.04 |
| SecSpeed | 0.02 -0.02 |
| **Editing** | |

VolumetricFogModulation contains the following data:

- texture: the texture used for modulation of the fog. The red channel modulates the fog density, while the green channel modulates the fog (an empty string disables the modulation),
- color by (1 – green value),
- tiles: the number of times the texture should be applied to the fog,
- modStrength: the strength of the modulation (0 disables the modulation),
- PrimSpeed: the speed in x and y direction (or u and v) of layer 1 of the texture modulation,
- SecSpeed: the speed in x and y direction (or u and v) of layer 2 of the texture modulation.

## Adding a VolumetricFog object in script.

To add a VolumetricFog object in script you can use the following template:

```
%fog_object = new VolumetricFog() {
      shapeName = "art/environment/Fog_Cube.dts";
      fogColor = "200 200 200 128";
      fogDensity = "0.2";
      ignoreWater = "0";
      MinSize = "250";
      FadeSize = "750";
      texture = "art/environment/FogMod_heavy.dds";
      tiles = "1";
      modStrength = "0.2";
      PrimSpeed = "-0.01 0.04";
      SecSpeed = "0.02 -0.02";
      position = "748.644 656.371 65.3506";
      rotation = "0 0 1 20.354";
      scale = "40 30 6";
   };
```

## Methods and Callbacks.

The VolumetricFog object has the following methods:

- **VolumetricFog::SetFogColorF(%this, %new_color)**: Sets the fog color to new_color. New_color RGB-values in the range of 0.0 – 1.0, A(lpha)-value is ignored.
  Example:
  My_FogObject.SetFogColorF(“0.7 0.7 0.7 1.0”);
- **VolumetricFog::SetFogColor(%this, %new_color)**: Same as SetFogColorF(new_color) except now the values are in the range of 0 – 255, again A-value is ignored.
  Example:
  My_FogObject.SetFogColor(“179 179 179 255”);
- **VolumetricFog::SetFogDensity(%this, %new_density)**: Sets the fog density to new_density.
  Example:
  My_FogObject.SetFogDensity(0.3);
- **VolumetricFog::SetFogModulation(%this, %new_strenght, %new_speed1,%new_speed2)**: Sets the fog modulation strength to new_strenght, the modulation speed of the first layer to new_speed1 and the modulation speed of the second layer to new_speed2. New_speed1 and new_speed2 are x y values (or u v if you like).
  Example:
  My_FogObject. SetFogModulation(0.5,”-0.2 0.5”,”0.02 -0.01”);
- **VolumetricFog::Dissolve(%this, %speed)**: Decreases the fog density every %speed milliseconds until it is dissolved (density = 0).
  Example:
  My_FogObject.Dissolve(500);
- **VolumetricFog::Thicken(%this, %speed, %end_density)**: Thickens the fog every %speed milliseconds until density is equal to %end_density.
  Example:
  My_FogObject.Thicken(1000,0.6);

Callbacks:

- **VolumetricFog::onEnterFog(%this, %obj)**: Triggered when the control object (Camera or Player) enters the VolumetricFog object. %obj contains the ID of the control object.

- **VolumetricFog::onLeaveFog(%this, %obj)**: Triggered when the control object (Camera or Player) leaves the VolumetricFog object. %obj contains the ID of the control object.

## Miscellaneous functions and variables.

There is 1 function and variable that deals with the size of the rendertargets.

- **SetFogVolumeQuality(%new_quality)**: This function changes the size of the rendertargets used by the shader to render the VolumetricFog. In normal operation these rendertargets are the same size as the window or screen when playing the game. You can however resize them using this function. A value of 1 means normal size, 2 means halfsize, 3 is 1/3 size, 4 is quarter sized and so on. The aim of this function is to reduce usage of video-memory, but has the disadvantage that it will introduce render artefacts whit smaller sizes. Calling this function also changes the variable $pref::VolumetricFog::Quality
  Example:
  SetFogVolumeQuality(3);
- **$pref::VolumetricFog::Quality**: Variable containing the current value of the rendertarget size. Will be saved in prefs.cs and used if present.
  Example:
  $pref::VolumetricFog::Quality = 3;

## Artwork.

The folder Game/art/environment contains some models to get you going and to serve as an example on how to model the shapes in your desired 3D-modeling program. You have to keep in mind that if you are creating LOD-ded models and you plan to use .dae as the filetype, you need the .cs file that is created by the TSShapeConstructor to recognize the different LOD-stages. It is advised to use models with only one material applied to it, although this material is not used by the renderer (hint: use just 1 simple material with a diffuse color and no textures for all your fog models).

Also there are 3 .dds files to be used as modulation textures. If you create them yourself use the red channel for the modulation of the density and the green channel to modulate the color (this will be modified as 1-green value). It is not needed to generate mipmaps.

## Ideas.

The VolumetricFog object can not only be used as a plain and simple volumetric fog. By adding a texture, strength and speeds to it you can make it more alive, or use it to simulate a smoke filled room. You can also use it to simulate some kind of volumetric lighting. Adding a VolumetricFog object with a spherical shape at the same position as a pointlight gives a nice effect, or use a conical shape for a spotlight. Make the color of the VolumetricFog the same as the color of the light.

Also using the 2 callbacks you can create some kind of poisonous gas cloud. Start applying damage to the player when entering the fog, stop when the player leaves.

Or you can use strange shapes to create the effects that you want, be aware though that it will load and display concave shapes they will not render correctly (due to limitations of the current rendertargets).

## Contact information.

| | |
|---|---|
| Website: | www.richardsgamestudio.com |
| e-mail: | support@richardsgamestudio.com |
| Twitter: | @RichardsGame |
| | https://twitter.com/RichardsGame/ |
| YouTube: | RichardsGameStudio |
| | https://www.youtube.com/user/RichardsGameStudio |