HELP GUIDE

MAX OBJECT Guide

Object Support for GameGuru MAX

This guide outlines the extent of the first version of Object support for GameGuru MAX.

Object Creation

Creating a new object for GameGuru MAX requires some understanding of the different files and their contents to create a game asset that is fully compatible. Some guidelines are essential and other are more cosmetic, but all help create a level of consistency the user expects when using out of the box objects.

How To Make an Object For GameGuru MAX

One of the best ways to create your first object is to clone an existing one that closely matches the object you want to provide, be that a characters, a piece of inanimate scenery, a door, a key, etc. This is because a lot of the properties have already been set up for this type, meaning you have less to do to create your object and a greater chance it will do what the user expects right away.

Clone An Existing Object

First copy all the files associated with an existing object, all identifiable as they will share a common name so the files group together when sorted alphabetically. We will take "entitybank\Max Collection\Health\Milk Bottle" as an example, that comprises five files:

- Milk Bottle.fpe
- Milk Bottle.dbo
- Milk Bottle_color.dds
- Milk Bottle_normal.dds
- Milk Bottle_surface.dds

The main file here is the FPE file which includes all the properties of the object, and references to all the other files. The DBO is the proprietary format used by GameGuru MAX and is the only model format allowed as the final destination for objects in the object library. The built-in model importer will take your FBX, GLTF or OBJ file formats and convert/export them as DBO model files. The three DDS files are texture files and note the very specific postfixes for *_color, _normal* and *_surface* which are required to give the object it's PBR capabilities.

All Objects Should Be Sealed

When you prepare your 3D model for importing into GameGuru MAX, you must ensure that all meshes are sealed, that it, that there are no holes that would allow the user to see inside the mesh of the model from any angle. This is required as both the collision system and the shadow rendering system requires that meshes always have a sealed outward facing polygon set.

Also, when you want to create thin meshes in your model, ensure they are at least 5 units thick across any dimension of chunkiness. Again, this allows the shadow system to work properly. You can have a thinner polygon, but it needs to be created as either a double sided polygon (4 polygons making the quad) or a regular one sided polygon set to double sided in the material properties, be set as transparent and be created as its own mesh, not connected to a larger polygon soup. In this way the game engine can have control over this special case, but avoid creating this wherever possible as there is a performance hit with such constructs.

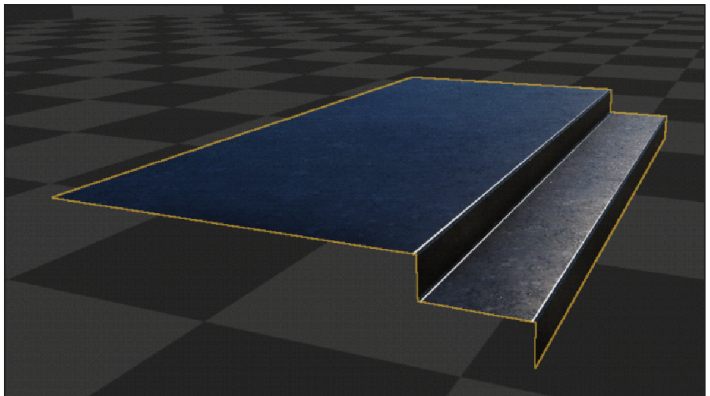

Figure 1 : An example of a mesh that is not sealed