Importing models into GameGuru

Contents

What you need

For a model to work correctly inside GameGuru you will need **six** things:

1. The **.x file** of the model
2. A **Diffuse** image map
3. A **Normal** image map
4. A **Specular** image map
5. A **.BMP** image
6. An **FPE** file

Explanation

.x File

This is the actual DirectX model file that you want to import. General models should be kept around 500 polygons and characters around 6,000 polygons. You can check if an .x file has LOD variants by opening it in a text editor (like notepad) and looking for **LOD_1** and **LOD_2**.

Please note the following carefully to make sure the model imports correctly.

- The filename *should* be in lowercase and can have spaces
  - Example:
    ```
    my model.x
    ```
- The _D, _N and _S texture files *can* share the name as the .x file, but this is not required and in the case of reskins; not convenient. It should however be something relevant.
- The BMP, FPE and X file **must** all share the same name
  - Example
    ```
    My Model.bmp
    My Model.fpe
    my model.x
    ```

---oOo---

Diffuse Image map

A Diffuse map is a texture you use to define a surface's main colour.

In order to work well with a normal map and a specular map, a good diffuse texture should not have any directional lighting included, it should only have generic "ambient occlusion" - i.e. the surface gets darker in deep cracks and around embossed details. If you are generating your normal maps from high-poly geometry, you should bake a matching ambient occlusion pass from the geometry and multiply this on top of your diffuse texture to make sure that the lighting matches the normal map.

Please note the following carefully to make sure the model imports correctly.

- The diffuse image map **must** be in .dds format (not jpeg, bmp, png or gif)

- The name of the file **must** end in '_D.dds'
    - For example "`mymodel_D.dds`"
- The diffuse, normal and specular texture files **must** share the same name and **must** be in lowercase apart from the suffix.
    - Example:
      ```
      mymodel_D.dds
      mymodel_N.dds
      mymodel_S.dds
      ```

---oOo---

Normal Image Map

Normal mapping, or "*Dot3 bump mapping*", is a technique used for faking the lighting of bumps and dents - an implementation of Bump mapping. It is used to add details without using more polygons. A common use of this technique is to greatly enhance the appearance and details of a low polygon model by generating a normal map from a high polygon model or height map.

Please note the following carefully to make sure the model imports correctly.

- The normal image map **must** be in .dds format (not jpeg, bmp, png or gif)
- The name of the file **must** end in '_N.dds'
    - For example "`mymodel_N.dds`"
- The diffuse, normal and specular texture files **must** share the same name and **must** be in lowercase apart from the suffix
    - Example:
      ```
      mymodel_D.dds
      mymodel_N.dds
      mymodel_S.dds
      ```

---oOo---

Specular Image Map

Specular maps are the maps you use to define a surface's shininess and highlight colour.

The higher the value of a pixel (from black to white), the shinier the surface will appear in-game. Therefore, surfaces such as dry stone or cotton fabric would tend to have a very dark specular map, while surfaces like polished chrome or plastic would tend to have lighter specular maps.

The colour of a pixel is also used, to calculate the resulting colour of the surface. A very saturated specular map will have a very different visual effect than a grey specular map. If you need a more "neutral" highlight on a surface, your specular map should use the inverse of the diffuse map's colour. Using the same colour on the specular as on the diffuse will result in a more saturated highlight when viewed in the game.

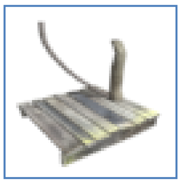
You can use contrasts in specular to make a surface appear more visually interesting in the game - for example, this door has a very dark specular for the wood while the metal parts are much lighter, which will make the metal stand out more as a shinier surface when light hits it. This sort of contrast can help make surfaces in the game appear more realistic too.

Please note the following carefully to make sure the model imports correctly.

- The specular image map **must** be in .dds format (not jpeg, bmp, png or gif)
- The name of the file **must** end in '_S.dds'
    - For example "`mymodel_S.dds`"
- The diffuse, normal and specular texture files **must** share the same name and **must** be in lowercase apart from the suffix
    - Example:
      ```
      mymodel_D.dds
      mymodel_N.dds
      mymodel_S.dds
      ```

---oOo---

.BMP image



This will be the thumbnail image that represents the model that users will see in the Library. It **must** be 64 x 64 pixels, a **bitmap** (**bmp**) file and it is important that the background **must** be white (RGB(255,255,255)) like the example on the left.

---oOo---

FPE file

This is the datafile for the model [entity] and describes how the entity should appear in game. Here's an example:

```
;header
desc          = Broken Cinderblocks (Large)

;visualinfo
textured      = cinderblocks_D.dds
effect        = effectbank\gameguru\entity_basic.fx
castshadow    = 0
collisionmode = 1

;orientation
model         = broken cinderblocks (large).X
offx          = 0
offy          = 0
offz          = 0
rotx          = 0
roty          = 0
rotz          = 0
scale         = 100
defaultstatic = 0
materialindex = 3
```

```
;statistics
strength       = 25
explodable     = 0
debrisshape    = 1

;ai
aiinit           = appear1.fpi
aimain           = default.fpi
aidestroy        = disappear1.fpi

;spawn
spawnmax       = 0
spawndelay     = 0
spawnqty       = 0
```

---oOo---

Steps to Make Model GameGur Ready

- Initial Steps

1. Rename the .x file to something appropriate. For example:

   ```
   mymodel.x
   ```

2. Rename the Normal (_N), Diffuse (_D) & Specular (_S) to something appropriate. For Example:

   ```
   mymodel _N.dds
   mymodel _D.dds
   mymodel _S.dds
   ```

- BMP and FPE

1. In your art package, create and save a BITMAP (.BMP) that is 64 pixels by 64 pixels. This will act as your THUMBNAIL in the Selection Editor and so should represent your model. It is common practice to use a Pre-Render from your 3D Art Package.
2. Rename this file according to your naming conventions. For example:

   ```
   MyModel.BMP
   ```

3. Using Notepad or some other plain text editing software either make a copy of the provided **Cinderblock** FPE file here in this document, or copy an FPE file from another entity or from a provided empty template of an FPE file ready for editing.
4. Rename this according to your naming convention. For example:

   ```
   MyModel.FPE
   ```

- Customising the FPE - Description and Association

1. Open the FPE file you've just copied in a text editor like NOTEPAD or WORDPAD
2. Edit the Following (using the Cinderblock FPE as an example):

   ```
   desc = Broken Cinderblocks (Large)
   ```

   Change this to a short description of your model. For example:

   ```
   desc = MyModel
   ```

3. Edit the Following (using the Cinderblock FPE as an example):

   ```
   model = broken cinderblocks (large).X
   ```

   Change this to the name of your own .x file. For example:

   ```
   Model = MyModel.x
   ```

4. Edit the Following (using the Cinderblock FPE as an example):

   ```
   textured = cinderblocks_D.dds
   ```

   Change this to the name of your own texture file. For example:

   ```
   textured = MyModel_D.dds
   ```

- Customising the FPE - Dynamic Verses Static

There are two types of entities in the world, DYNAMIC and STATIC. A DYNAMIC object can be moved around, like barrels, crates, keys and guns, whilst STATIC objects are immovable and are not affected by player interactions, gravity or physics, for example houses, rocks or trees.

If your object is a STATIC object (in as much as it won't move or be interacted with) then change *defaultstatic* to 1:

```
defaultstatic = 1
```

otherwise if it can be moved and / or interacted with:

```
defaultstatic = 0
```

- Customising the FPE - Collision Modes

Depending on the type of item, the way in which collisions are handled can be specified by use of the COLLISIONMODE field.

If you have a basic model that does not require precise collision such as a crate, step or wall, then a BOX collision would be a good option. This is the fastest collision method available. This is achieved by entering:

```
collisionmode = 0
```

If you have a more complicated model that is not as angular such as a vaulted arch, rock formation or is otherwise unusually shaped; POLYGON collision would be a good option. This is the slowest collision method available but offers a greater degree of accuracy. This is achieved by entering:

```
collisionmode = 1
```

You may have objects that you simply do not want to collide with anything. Background items that having collision detection would be impractical. Objects such as ground plants, debris or wires on the floor - there only to add colour to the scene but not to hinder the player. Collision for these types of items can be turned off completely by entering:

```
collisionmode = 11
```

For very complicated items, you may only wish to have partial collision. A Tree for example. POLYGON collision would be very slow due to the volume of the leaves and BOX collision would be too imprecise as the player would not be able to get close to the tree. For these types of items, limb collision can be used.

```
collisionmode = 100x
```

Replacing x with the limb number will cause that limb only to receive BOX collision and the rest of the object will have no collision at all. Following the above example this allows the trunk of the tree to have collision allowing the player to walk right up to it without colliding with the leaves.

For unusual or curved objects such as bent trees or collapsed pylons for example, POLYGON collision can be achieved by using:

```
collisionmode = 200x
```

- Testing Your Model

The six files you have created (the .x file, the diffuse, normal & specular image map, the BMP image and the .FPE file) can then be moved into the **ENTITYBANK** in the GameGuru installation folder, usually along the lines of

C:\SteamLibrary\SteamApps\common\Game Guru\Files\effectbank\reloaded

It's best to create a unique sub folder within the entity bank to hold your models, this avoids messy problems of multiple artists creating similar items that happen to share mesh or texture names, especially on the more generic items such as boxes, rocks or trees.

C:\SteamLibrary\SteamApps\common\Game Guru\Files\entitybank\My Models

Common Questions / Problems

Q> My object is showing up as a little white sphere / dot!

A: There is a problem with your **mesh** (.x) file somewhere. Make sure it is spelt correctly in the FPE file.

Q> My object is loading but is white, there's no texture.

A: There is a problem with your **texture** file. Make sure it's spelt correctly in the FPE file.

Q> My model does not appear at all (no white sphere / dot).

A: There could be a problem with the scale of your model. Try increasing the size by 1000% in your 3D modelling software and re-exporting. This should at least make the object visible in the editor and give you some indication of what the correct export scale should be.

You can also scale the model in the FPE file, use the field called **SCALE** with 100 being default, 50 being half size, and 200 = double sized.

Q> My object loads correctly but is not at the end of my cursor.

A: Your object may have been exported in an offset position. Ensure you export your model at 0,0,0 from you 3d modelling software.

Q> Why does the diffuse, normal & specular map have to have the exact same name?

A: The FPE only looks for the *diffuse* (_D) image map, GameGuru will automatically look for the *normal* (_N) and *specular* filenames based off the *diffuse* image map file name. This is why it's **important** that the names (apart from the suffix) be identical.

Q> I have a tree that I made but how do I know which limb is the trunk for selective collision?

A: First you need to ensure it was exported with the trunk and leaves as two separate objects (independently selectable within the 3D modelling tool).

When exporting your X file look for the option to export as a TEXT file.

View this in NOTEPAD/WORDPAD...at the end of the document, your limb names will be shown followed by a lot of data information. The order in which these are shown is the limb order. i.e. The first block of information is limb 0. The second block is limb 1, etc. Once you find out which limb you need, you can then re-export as a Binary/Compressed Binary.

ADDITIONAL NOTES

Scaled Exports

In order to ensure entities are handled properly in the engine, ensure that all scaling transforms in the model are removed for export. That is, any modifiers in the model creation stack which apply a scaling function to the geometry should be removed by baking those new scales into the raw geometry data. This will ensure collision, AI avoidance and shadowing are all handled properly in the engine. Failure to take this step could result in erroneous collisions and objects/shadows scaled out of proportion during the final render.