Generating sprite graphics in The Sims

Overview and background

Objects

Objects in the game are stored as files in the "Objects" directory. Each object has two resource files in this directory that contain all the information necessary to view and play with the object in the game.

The SPF file contains all of the drawing information for the object. The IFF file contains other resources crucial to loading and playing with the object. The separation has helped us in being able to exclusively edit different parts of the object.

If two objects have similar play, they may reside in the same file. This is a way of making it easier to track and revise and to save some ram. For example, we choose to put all of our bed objects into a single file pair, Beds.iff and Beds.spf.

Object Graphics

The graphics for each object consists of three parts. The first part is the palettes. These are all the colors used by an object and are referenced by the sprites when drawing in the game. The second part is the **sprites**. These are the component pieces of the art for an object. Two or more sprites may share the same palette. The second part is the **draw list** or **draw groups**. This is the information that specifies how the sprites are to be combined to create the full view for each state of the object (broken, doors opening, on/off, etc.).

Importing content

The sprite and draw list information is generated from source targa and bitmap files, together with a simple script that specifies how each source file is to be used. The script is called an OMK file and must be hand-authored by a programmer prior to importing an object.

Each object has two exported 32bitTGA files and one artist-generated 8bit BMP file per sprite. One TGA file will be named according to the naming field in the Sprite Exporter; this one has the z-depth information in its alpha channel. The other will be named according to the naming field in the Sprite Exporter as well, but will have "Alpha" appended to the name; this one has the alpha information in its alpha channel required to have the sprite antialiased in the game. The BMP bit depth is 256 colors for RAM and CPU speed reasons; this is the file that the game gets the actual pixel art from. Since palette selection is an art in itself, an 8-bit BMP file must be created for each TGA file either by hand in a tool like Photoshop, or scripted in Debabelizer.

To save space, palettes are shared by BMP files. There may be more than one palette per object if necessary, but we try to have the object use one palette throughout all sprites.

The columns in a source file correspond to the different views in the game. First are 4 rotations for close in view, then 4 medium and 4 small, reading from left to right.

The rows correspond to the component sprites per tile, and must be exactly in the order specified by the OMK file. Usually the exporter puts them in the right order automatically. If the source file is too small or if a group of source files have different sizes, the import will fail. To save export time, unused columns may be omitted if the sprite has symmetry.

Directory structure

The game and our import scripts expect certain directories to be in the right place. This should be taken care of automatically if using source safe or a zipped up copy of what is in source safe. There are two main directories for the purposes of generating sprites.

{root}/SimsBuild/Runtime/GameData The game data directory.
{root}/TDSContent/Sprites The source sprites directory.

The objects directory mentioned above is within the data directory. Other types of non-object sprites are stored in other places under the data directory.

The sprites directory contains a folder for each object in the game. The folder is named after the root name of the object file. For example the folder for Beds.iff and Beds.spf would be just "beds". Within each

of these directories is the OMK file as well as **all** of the source art work. This includes the TGA and BMP files mentioned above as well as the MAX model. For organizational reasons, if a directory is shared by more than one object, there are sub-directories for the art work of the different objects.

SprMaker

SprMaker is a tool that automates parts of the import process. Basically once all the source files have been generated and saved in the proper locations, SprMaker puts objects "in the game". SprMaker is a stand-alone MFC executable that may be run from any location.

Directories

In order to operate, SprMaker must be given the sprites directory and the data directory so that it can find all the source and target files for you. When it starts up, it reports the status on these two directories and the temporary directory. The temporary directory is just for temporarily saving the results for the import process.

To set your content directory, click "Open TDSContent". Use the file browser and go to your content directory and open the file "TDSContent.scnt". This is a placeholder that the app uses to avoid confusion about what is the right directory.

To set your data directory, click "Open Game Data". Use the file browser to locate your data directory and open the file "Scenario.tds". This is also a placeholder.

Once you have done this, you may choose to click the button "Remember in C:" so that SprMaker will automatically set up its directories the next time it is run.

Selecting the object to build

File List

The leftmost list of items in the SprMaker window contains all of the **writable** SPF files in the Objects directory. This is because we use a source control system that locks files to ensure that you cannot change a file that is not "checked out". The "Show All" checkbox allows you to see all object files, even if their import will fail due to the file being locked. You can also click "show all" twice to update the list after you have made changes to a file's properties to be writable from outside the program.

Object List

Once an object file is selected, the middle list will show all of the objects that are contained in that file. There should be one or more objects in each file. If nothing shows up here, something is wrong. SprMaker automatically selects the first one in the list.

Draw group list

The third list on the far right is for informational and debugging purposes only. It shows the internal id numbers used by all the draw lists for the selected object. Clicking in it has no effect, but if the list shows no items for a selected object, something is wrong.

Building an object

Once an object is selected, the build button changes to reflect its name. Clicking the build button will kick off the build process. While the build process is working, the three status bars reflect what it is currently doing.

The small status bar on the left shows whether the process is working or idle. While it is working, other items in the window are grayed out, and the status shows a text animation. The second one to the right shows the current sprite that it is working on. Each sprite requires two steps, the rendering of the TGA and BMP and the compression of the data into sprites. The lower status bar shows which of those two steps the build process most recently reported.

When the build is finished, the first status bar will return to "idle" and the report window below will show all of the text generated by the report. If there was an error that cause the build process to fail, it

shows up at the end of the report as a “fatal error”. So you should always glance at the report after building an object.

In-game viewing

When The Sims starts up, it loads all of the object’s files and creates a map of all resources and resource data. Editing these files in any way from outside the game while the game is running can cause a crash! Since shutting down and restarting the game take so long, there is a facility for suspending the game so that the files may be edited and the resource map rebuilt.

To generate sprites while the game is running, you must have a build with “edith”, which is our embedded content editing program. You also must be running the game in windowed mode. To run in windowed mode, you must set your video mode to 16 bits, and use a shortcut to the game which specifies “-w” in the target. Once the game is running, click on an occupied lot to bring up the house view. Then press the ‘e’ key to bring up edith. If nothing happens, you may not have an edith build and will need to obtain one from someone on the sims team.

Once the edith window is up, you can suspend file access by choosing “Suspend Files…” under the File menu. This will show a modal dialog and the game will not draw or play at all until the OK button is pressed. At this time, use SprMaker to build some sprites. When SprMaker has finished, return to the game and press OK. If the object has already been placed, you may need to zoom in and back out to reload the graphic. The catalog thumbnails do not update at this time.

Zipfile naming convention

In order to keep a running record of all deliveries from New Pencil, as well as all deliveries from Maxis, a naming convention for zipfiles has been established to reduce confusion and version errors.

All .zip files will follow this convention <prefix>_<name>_<date>.zip, where:
<prefix> = “ToNP” if it is being sent to New Pencil from Maxis, or “ToMaxis” if it is being sent to Maxis from New Pencil.
<name> = some short name which is generally descriptive of the contents
<date> = the date the .zip file was created, expressed in 4 digits of the month and the day.

Example:
ToNp_Newbuild_0611.zip (A zip file containing a new build of the game, created on June 11th, and sent to New Pencil from Maxis.