

Hey Phil,

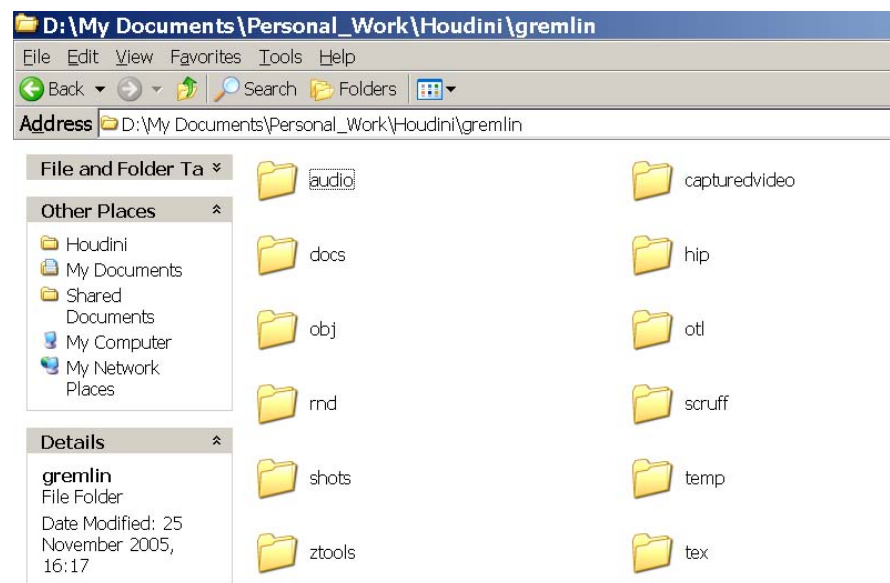
How's things?

We kind of concluded that the directory structure in Creating 3D effects was geared towards effects houses, so we used it as a template to devise a system that would be more useful for sharing between a smaller scale project typical of the style of workflow we have at uni.

What I'm going to tell you isn't what we had set up originally for short lived, but kind of evolved from what went wrong with the directory structure we had then. We found ourselves needing to reorganise things and add tons more folders to get it working. When it finally came together it was such a help.

I've added attachments to guide you through a structure I have at the moment for a personal project I'm doing between Houdini and ZBrush (which rocks by the way) modelling a gremlin type thing.

01 - Directory overview



Everything is contained in the main project folder (gremlin).

This main folder holds everything that is needed for the project as a whole - (i.e. rnd folder for project research, docs folder for admin, scruff for misc junk) The important ones to note at this level are the hip folder, shot folder and capturedvideo folder.

The capturedvideo holds all the *unedited* raw captured video, so if any changes are made to the edit it is held there unaffected.

The hip folder holds all the hips that cover the whole project and are not shot specific - ie modelling, texturing hips for characters. These can then be finalised into obj's or otl's and put into appropriate folders also at this level. The tex folder also remains at this level and holds all the textures.

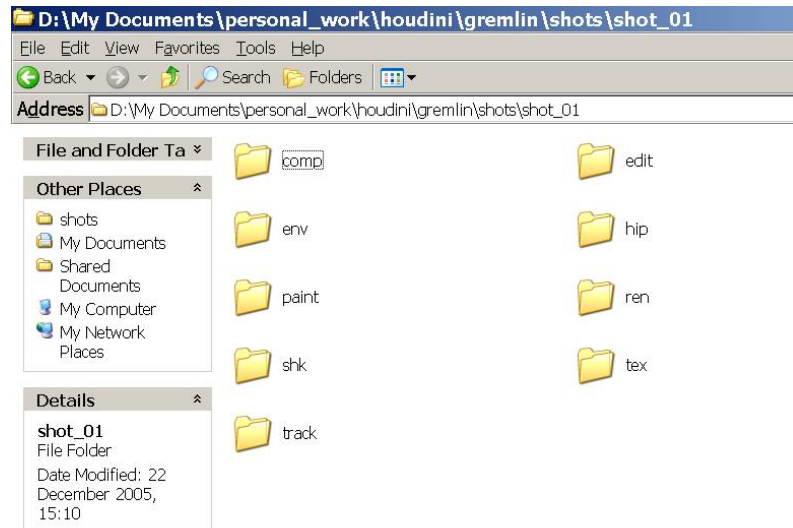
The! shot folder holds all the individual shots for the project. These are finalised once an edit is locked and put into sequential order.

02 - the shot folder

so the shot folder contains a sequence of folders that relate to the edit order, shot_01, shot_02, shot_03 etc. I also tend to keep in shots that dont have any effects done to them, to keep it in logical order to the final film.

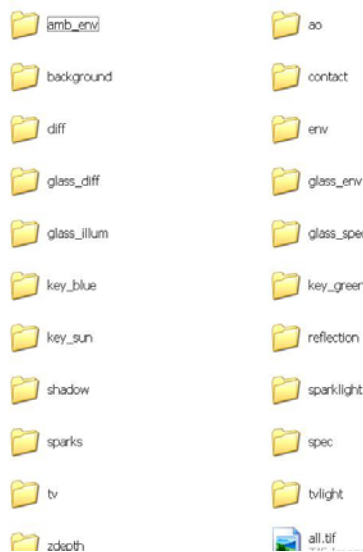
Better than naming it treeshot5 and getting confused.

Inside each shot folder is a directory of new folders named for each phase of the shot pipeline. so edit folder contains the cut of the edited shot, env holds environment maps for that specific shot location, track holds tracking data, paint holds mattepaintings or touchup bits.



The important ones here are hip, shk, comp and ren. hip is for shot specific hip! files, ie animation. shk is for all shake compositing files and comp is for the final composited rendered out sequences to go. Ren folder is for all the elements that make up the composite, and are typically rendered out of houdini. This can fill up with even more folders but is essential to keep each element separate. Folders in here are called things like spec, diff, ambEnv to reflect the type of pass.

I have attached an example of the ren folder from short lived demonstrating all the passes.



03 Naming conventions.

ok, we used a standard method so that it would be logical at any time what any file was, and from what sequence. This is basically **name_function_v1r1.0001.tif** **v = version r=revision and padding is only necessary if it is a sequence**

(! so to save out of houdini it would be: **name_function_v1r1.F4.tif**
out of shake it would be: **name_function_v1r1.#.tif** ... i know you know this, just pointing out in case anyone else reads it)

In the main folder, saved files are named as logically as possible, but are not shot specific - so for the model of the gremlin in the hip folder **gremlin_model_v1r1.hip** is satisfactory, and as it progresses in saves, the version and revision number are updated. (we took a version as being a complete rehaul and new idea, and revision to be a progressive modification)

In the shots folders, everything has become specific to the shots - so they need to convey that. So, each file, no matter what it is, starts with its relevant shot name.
for example: **shot01_cmp_v1r1.shk** indicates that this is a shake file composite for shot 01. **shot03_grade_v2r1.shk** would be a grade for shot03.

Other example: **shot08_spec_v1r3.0001.tif** indicates that this is frame one of the spec pass sequence - version 1 revision 3. If there were other revisions rendered out, i would know to pick 3 as the latest - this demonstrates how new ideas can be tested whilst keeping older revisions and everything tidy.

I hope that i havent waffled to much and that this makes sense.

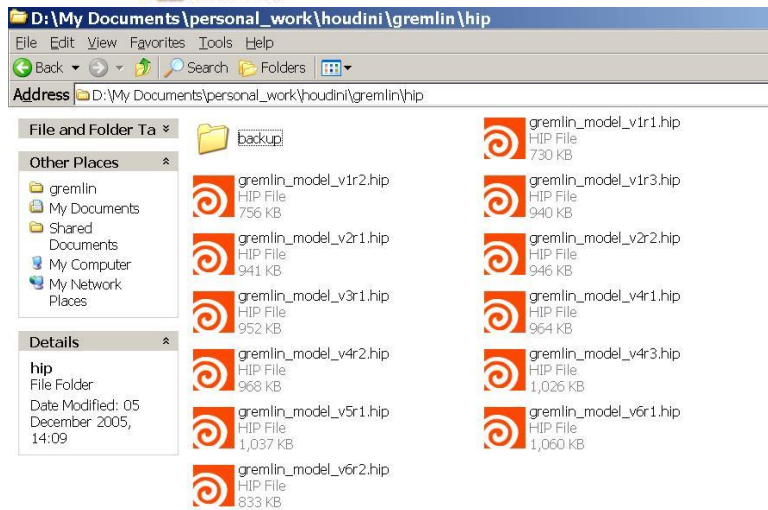
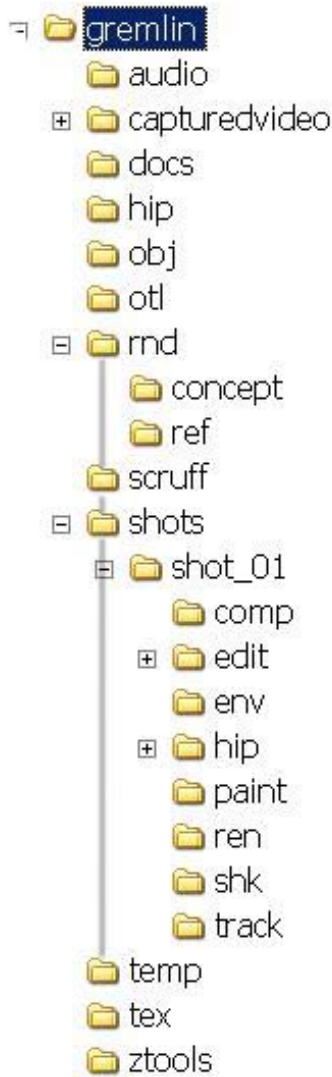
Obviously the structure is flexible - if u prefer to call folders by different names thats fine, as long as everyone else in the group does exactly the same, or it all falls apart very quickly. The cool thing is, once everyone is set up this way its impossible to get lost if everyone keeps to the rules - really really helps for a group project, which i believe is coming u! p.

I hope that i have covered what you wanted to know, if any of this makes no sense what so ever, let me know and ill try and explain it better. Hopefully the pics attached will make it more logical.

oh and one final thing - EVERYTHING should be lowercase - had way too many nightmares caused by not doing that.

All the best, and have a very merry christmas!

John



To help you stay safe and secure online, we've developed the all new [Yahoo! Security Centre](#).

Hey Phil,

Glad I could help! I think what is really great about this system, is that once it is understood and set up it allows the user to work on a project and not worry so much about where to put things. Also with \$JOB variable set it makes it all so fast and easy to switch between machines.

I would agree about setting the main gremlin folder to \$JOB, and then whenever a new shot it worked on, that is assigned to the relevant \$HIP.

As for shake, you can set an environment variable for the project, and call it whatever you want, This has to be done in the shell first, then a .h file has to be made and referenced to. Ill copy and paste a document i have on it to the bottom of this email.

As for uni, great to hear about your promotion. Well deserved, and it must be great to have control over the direction of the course. I agree th! at it would be fully possible for a photoreal project to come out of term 1. Things like the directory structure really helps, as does an understanding of the multipass stuff.

I spoke to Melania briefly a couple of days ago, and suggested coming in to do a small masterclass on how we did the shortlived multipass setups, from Houdini through to Shake. It would give students a good idea about what to do with onset data. Once that kind of thing is laid down as groundwork, it opens up a world of creative possibility. I did send the shake files to melania but the disk didnt work, so what I might do is send a version again - maybe of the gremlin project so it includes tracking and better set up of mulitpass in houdini (the new tutorial on sidefx.com answered so many mistakes we made).

I've also managed to get hold of some more gnomon dvd's i could cop! y for you. Most are a bit naff - but the really good ones are by Eric Hanson, and Digital Sets 4 deal with some amazing 2.5d stuff. (i.e, taking full 360 panorama's, gathering GPS data from your location and downloading free Digital Elevation models from the net of your location and converting them to a displacement map. Some incredible results - jaw drops)

Anyway,

merry christmas!

John

PS thanks loads for the hip files, they are very cool - do you have much on fur/hair? that will be my next personal research when i get some time. !!

Shake:

To set an environment variable in the Terminal, open either your *.tcshrc*, your *.aliases*, or another file that gets read when you open a shell. Enter a line something like this:

setenv myproj /Documents/shot1

Once you have saved the file, type

source .tcshrc

and you have now set your environment variable. All shell windows that you have already created, and any open applications must be relaunched to read the new variable. For variables you've set using the *environment.plist* file, you have to log out and back in again.

To Test Your Environment Variable

There is a simple way to test if your environment variable exists. In a Terminal, type *echo \$myproj* and it should return the proper value.

Now for the Shake part.

- Go to your *\$HOME/nreal/include/startup/ui* directory and create a text file called *paths.h*. Enter the following lines. Note the / at the end of the second and third entries:

```
nuiFileBrowserAddFavorite("$myproj");  
gui.fileBrowser.lastImageDir= "$myproj/" ;  
gui.fileBrowser.lastScriptDir= "$myproj/" ;
```

- Create a directory that *\$myproj* points to, that is, if you set it to */Documents/shot1* then create */Documents/shot1*.
- When you relaunch Shake, it should launch to where *\$myproj* is. If you pull down the *Directories* menu, you also see *\$myproj* listed.

- When you read an image from here, Shake keeps the environment variable (*\$myproj*) in the path. Therefore, whenever you move the entire project, reset the environment variable to point to the new path.

- If you batch-render your project, the background machines must also understand the environment variable.

You can take further advantage of environment variables and projects by adding your own *startup* directory into the project area. Specify a shot-specific cache directory, assuming you have disk space to burn, and, golly, with drive prices so low, who doesn't? This! is only useful if you are working on several shots at once, as it keeps cached files around on a per-shot basis. Be *extremely* careful with this, however, because you can end up with Gigs of data that you don't need if you don't clean up after yourself. Kind of like real life.

To set per-project settings for Shake:

- As an example, in your project directory, create *startup/ui* directories, that is,

```
/usr/shot1/startup/ui
```

- In your *startup* directory, place a file to relocate your cache. Create a text file called *cache.h* and add these lines, obviously changing *MyShotName*:

```
diskCache.cacheLocation="/var/tmp/Shake/MyShotName";  
diskCache.cacheSize = 500;
```

The second line indicates the size in MB of your cache. Set it to something appropriate ! according to how much disk space you have to spare. Shake automatically creates that directory the first time it needs it. Don't forget to remove this cache directory when cleaning up your project.

- To have per-shot macros or settings like Formats, add them into the *startup* and *ui* directories. See the 769 tutorials and pages on this process under *Customization, Scripts* and *Advanced Tutorials*.
- Return to set another environment variable, following the steps outlined above. You want to set *NR_INCLUDE_PATH* to your project directory:

```
setenv NR_INCLUDE_PATH $myproj
```

To help you stay safe and secure online, we've developed the all new [Yahoo! Security](#)