
helga_pipeline_documentation

Documentation

Release 0.1.5

TimmWagener

March 25, 2015

| | | |
|----------|-------------------------------------|-----------|
| 1 | Organization | 3 |
| 1.1 | Quick Guide | 3 |
| 1.2 | Naming Convention | 3 |
| 1.3 | Work and Publish | 4 |
| 1.4 | Pipeline | 5 |
| 1.5 | Fur workflow | 6 |
| 2 | Standards | 9 |
| 2.1 | Formats and Standards | 9 |
| 2.2 | Houdini Variables | 10 |
| 3 | Checklists | 13 |
| 3.1 | Mesh checklist | 13 |
| 3.2 | Animation checklist | 15 |
| 3.3 | Photogrammetry checklist | 16 |
| 3.4 | Asset checklist | 17 |
| 4 | I am a | 21 |
| 4.1 | I am a | 21 |
| 5 | Pipeline | 25 |
| 5.1 | API Overview | 25 |
| 5.2 | Coding Convention | 26 |
| 5.3 | Running on a different OS | 27 |
| 6 | API Documentation | 31 |
| 6.1 | helga.houdini package | 31 |
| 6.2 | Packages | 31 |
| 6.3 | helga.maya package | 31 |
| 6.4 | Packages | 32 |
| 6.5 | helga.nuke package | 50 |
| 6.6 | Packages | 50 |
| 6.7 | helga.general package | 58 |
| 6.8 | Packages | 59 |
| 7 | Indices and tables | 67 |
| | Python Module Index | 69 |



Welcome to the Helga pipeline documentation. Here you can find almost all answers to questions about how we do things within the Helga pipeline. It is the first place to look for answers if you have a questions. The second option is to ask me (Timm Wagener), you are very welcome to do that as often as possible, or write me an [email](#).

Note: Please keep one thing in mind: A pipeline, or at least our pipeline, is not at first place a bunch of scripts that a TD hacks together that will automate this and that. A pipeline really is an **agreement** between all project workers, to meet certain standards that make life easier for everybody. All scripting and automatization comes **after** that and just builds on top of that!

Organization

1.1 Quick Guide

Note: Here is a quick cheat sheet of how we be rollin. In 95% of all cases you do the right thing when you follow these simple rules.

1. **Communicate!** If you are unsure, dont do it. Talk to the person that knows!!!
2. **If you dont know, and nobody is there to tell you: Use your rnd folder.**
3. **Use the naming convention** To let people will know who is responsible, what is the latest version etc. Read about it [here](#)
4. **Your work is not done when your work is done!** Your work is done when you correctly gave it to the coworker who uses it next (Publishing/Copying/Overwriting).
5. **Workspace and Publishing** This is the system that builds upon the former rule. Work in your workspace and give it to your coworkers when its ready (publish). Read more about it [here](#). Much of the work your coworkers do depends on you being clean and organized (rig references for animators, texture references for shading artists, matte painting references for comp artists).
6. **Work clean and dont rush! If you dont work clean, you cause somebody else more work.**
7. **Never reference/use something from a work directory**

1.2 Naming Convention

Here you can find a quick reference for the naming convention we use.

Note: This naming convention holds true for most of our pipeline. Please try to match it as close as possible. Small errors count! Think of it as right **not** if a human can understand your versioning but if an algorithme could successfully parse your files and do something meaningfull without errors.

For work files it usually goes like :

```
ulfbert_model_a_0001_mh.mb  
ulfbert_sculpt_d_0012_mh.ztl  
helga_rig_base_c_0023_tw.mb  
snorri_concept_f_0012_ms.psd
```

1.2.1 Quick explanation

- **ulfbert_model:** Name that explains what file we are dealing with. Could also be *snorri_base_model* or *king_larry_the_third_model_from_hell*, just make sure people know the item that is versioned here.
- **a:** Wide iteration letter that is for **large** changes and is a landmark for you. You usually know (at least when you open a file) why you upversioned the letter
- **0001:** Version number. Always has four digits.
- **mh:** Modifier initials of the person that saved this file.

Chain them all together and you have a beautifully explanatory naming convention that helps everybody understand what he is dealing with.

1.2.2 Published files

Published files are a little bit different in that they just omit the versioning suffixes. This makes sense when you think about them as the current state of your work you made available to everybody.

Here is an example for the ulfbert rig :

Work folder

Here is the file how the rigger names and versiones it up

```
ulfbert_rig_body_c_0023_ah.mb  
ulfbert_rig_body_c_0024_ah.mb  
ulfbert_rig_body_d_0025_ah.mb  
ulfbert_rig_body_e_0026_ah.mb
```

Publish folder

Here is the published file. This file is used by all animators as a reference to do the animation so by overriding it, the rigger automatically updates the whole production.

```
ulfbert.mb
```

Read more about our work and publish system [here](#)

1.3 Work and Publish

Our system of **work** and **publish** explained. It is **really** as simple as copying and overwriting your work at the right locations, so dont be afraid ;)

1.3.1 Separate work and publish locations

This is just the simple principle that you manage what you give to production. Fellow coworkers never see your *messy* workspaces and only grab the content that you have published for them to use.

Important: If the object you want to use is **not** published, then **dont** just use a version from the responsible persons workspace. Instead **ask** the person for the content.

In many locations you will find a work folder which might contain the same folderstructure where itself is in for example :

```
3d/maya/scenes/assets/ <-- This is where people USE your work
    chars
    props
    cameras
    ...
    **work**
```

```
3d/maya/scenes/assets/work <-- This is where people work
    chars
    props
    cameras
    ...
```

Here is an example for concept artists, to show that this principle is not limited to 3d :

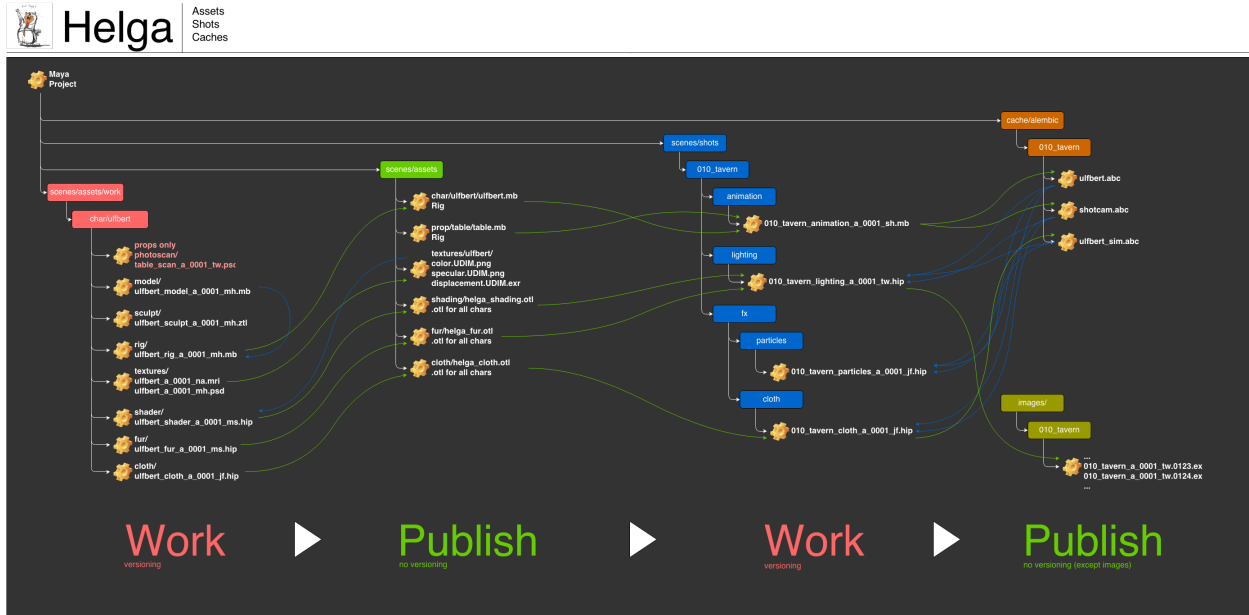
```
2d/concept_art <-- This is where people watch work
    chars/
        ulfbert/
            ulfbert_concept_hair <-- People go here to WATCH the latest ulfbert hair con
        props
        ...
        **work**

2d/concept_art/work <-- This is where people work
    chars
        ulfbert/
            ulfbert_concept_hair_a_0001_mh.psd <-- People go here to CREATE the latest u
            ulfbert_concept_hair_b_0002_mh.psd
            ...
        props
        ...
```

1.4 Pipeline

Here you can find some infographics to explain our pipeline visually.

1.4.1 Assets and shots pipeline



1.4.2 Photoscan pipeline

1.5 Fur workflow

1.5.1 Caching

1. Load Alembic from animation or simulation (both are identical) and use the character_fur HDA to create, simulate and tweak the fur.
2. Cache the complete fur to a separate file. This is the distilled and only output for fur. (All of the creative part of the fur process happens in a separate fur hip file, analog to the cloth or other effects files).

1.5.2 Rendering

1. Load the character_fur_render HDA and point it to the cache. (Or to the cache directory if several caches are needed for several hair systems). If not all needed caches exist, the fur rendering will fail. In render files will be NO tweaking on the fur or artistry on simulation/animation, only shaders will be tweaked there.
2. Load character_fur_shading HDA and plug it in material slot of character_fur_render HDA.

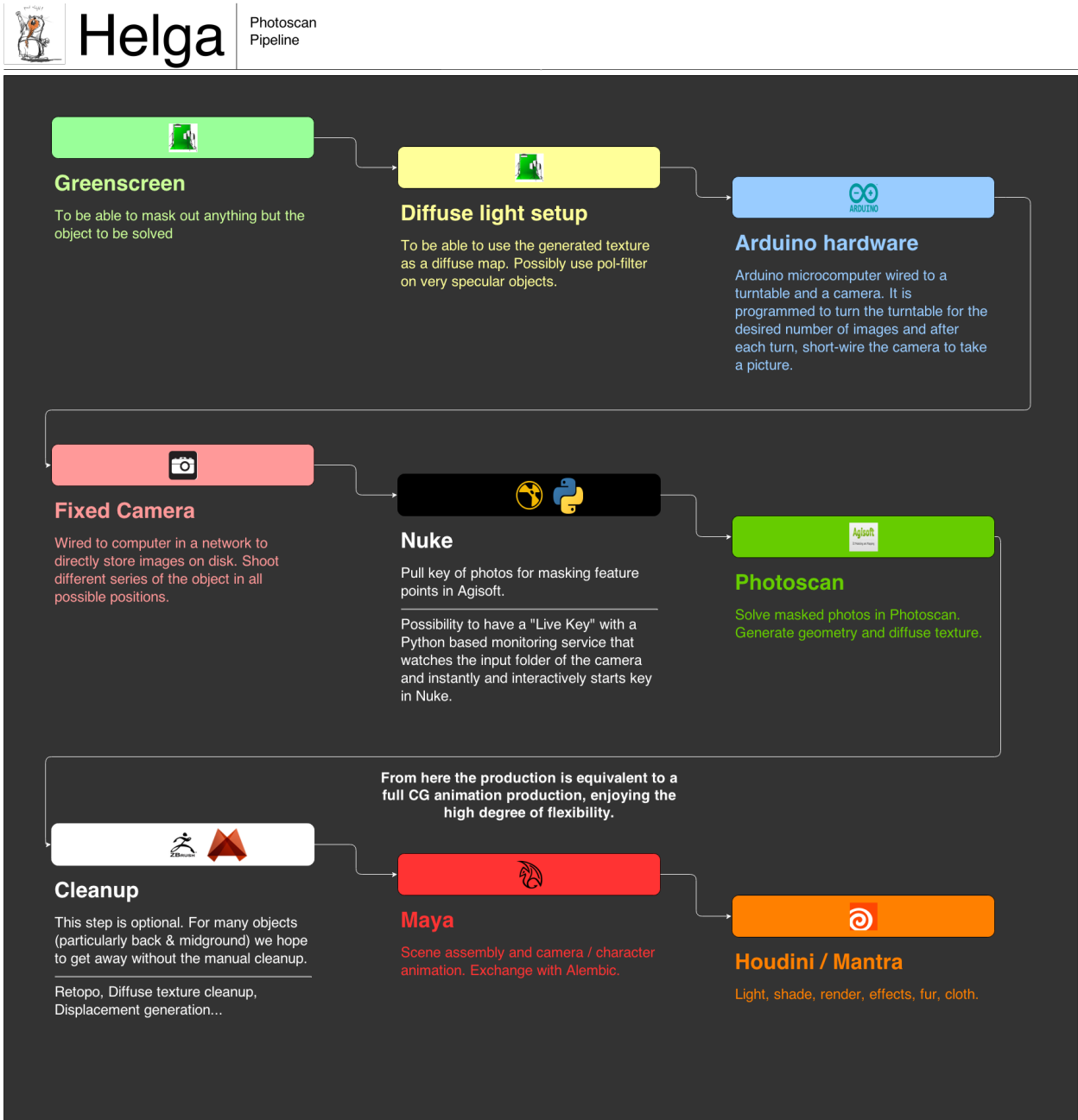
1.5.3 Required Digital Assets

1. character_fur HDA

Provides all the functionality to create/simulate/tweak and cache the fur. This is the motor.

Note: Needed functionality:

1. Read standardized Alembic from Animation/Simulation and create fur on it.
2. Ability to art direct the fur based on directors opinion



3. Simulate fur in order to account for intersection with cloth
4. Cache the fur.

All of this happens in a separate fur file for each shot done by the fur artist.

2. **character_fur_shading** HDA

Analog to usual shading assets. An HDA that just contains one or more materials to be plugged in the material slot(s) of the character_fur_render HDA.

3. **character_fur_render** HDA

HDA that wraps the character_fur HDA, hides all the complexity of the fur itself and exposes all the attributes needed for rendering.

Note: Needed functionality:

1. Material slots which allow to pick the materials from the character_fur_shading HDA or any other hair materials
2. Pick the cache done by the fur artist
3. Enable visibility for rendering and viewport

The character_fur_render HDA wraps the character_fur HDA for DRY reasons and to provide a stripped interface that hides the complexity for shading artists. If there is no cache, the fur cannot be rendered.

Standards

2.1 Formats and Standards

Note: Here you can find a list of formats and standards that we use in our production.

2.1.1 Textures

| | |
|---|-------------|
| Painted textures (color, specular, reflection, fur maps etc..... almost everything) | .tga |
| Photoscan textures (easier because .jpg doesnt have an alpha channel) | .jpg |
| Data maps, 32bit textures (displacement, etc.) | .exr |
| Textures for rendering (32bit and 8bit.....basically all) | .rat |

2.1.2 Rendering

| | |
|-------------|--|
| .exr | Half-float (16bit), zip-scanline, multichannel, Auto-data window, linear (Gamma 1.0) |
|-------------|--|

2.1.3 3d

| | |
|------|------------|
| Maya | .mb |
|------|------------|

2.1.4 Cache

| | |
|---|--------------|
| Geometry (Rigs, props etc.) | .abc |
| Cameras (Shotcam....) | .abc |
| Locator | .abc |
| Fur (<i>Guide curves are always cached before rendering</i>). | .bgeo |

2.2 Houdini Variables

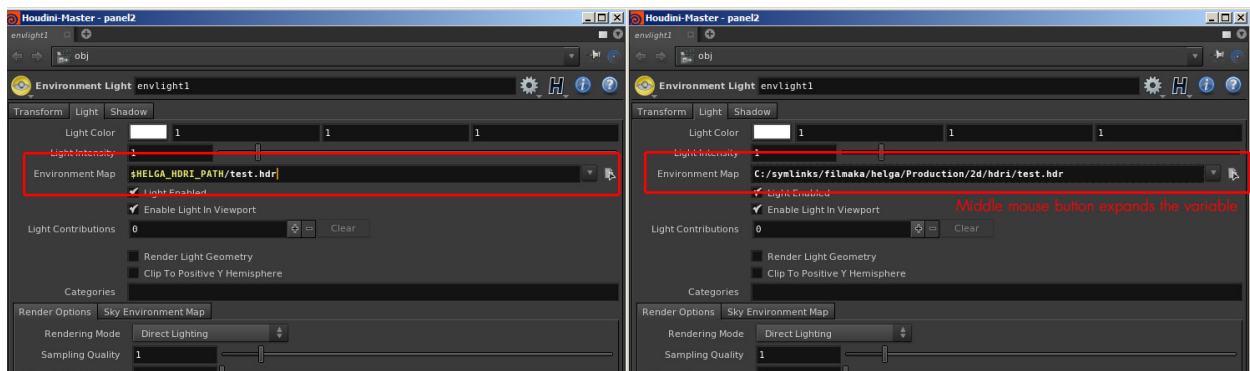
Note: Here is a quick overview of all the specific variables that can be used in Houdini. They **should** always be used instead of a hard wired path. They will expand before an .ifd is generated so they should be working with the farm properly.

All helga variables are prefixed with HELGA, if they aren't, they are not specific to the helga project.

Note: This list is subject to constant updates.

2.2.1 How To?

Here's how its used in Houdini. Its simple and comfortable.



2.2.2 Shots

| | | |
|-------------------|--------------------------|---------------------------------------|
| \$HELGA_SHOT_NAME | Name of the current shot | 030_establishertavern, 240_kick, etc. |
|-------------------|--------------------------|---------------------------------------|

This variable is added by the pipeline on startup if it doesn't exist. However, it is **NOT** given a value. The correct value for the current shot has to be set by the user. Please see below for examples.

Warning:

When importing Alembics (**characters, shots, cameras**) please always have your path starting with:

\$HELGA_ALEMBIC_PATH/\$HELGA_SHOT_NAME/

For example:

\$HELGA_ALEMBIC_PATH/\$HELGA_SHOT_NAME/props/

\$HELGA_ALEMBIC_PATH/\$HELGA_SHOT_NAME/cameras/shot_cam.abc

\$HELGA_ALEMBIC_PATH/\$HELGA_SHOT_NAME/chars/ulfbert_rendergeo.abc

2.2.3 Applications

| | | |
|---------------------|--------------------------|--|
| \$HELGA_HOUDINI_EXE | Path to pipeline Houdini | C:/symlinks/houdini/13.0.260/hfs/bin/houdinifx.exe |
| \$HELGA_MAYA_EXE | Path to pipeline Maya | C:/symlinks/maya/maya2014x64/bin/maya.exe |
| \$HELGA_NUKE_EXE | Path to pipeline Nuke | D:/tools/Nuke8.0v1/Nuke8.0.exe |

2.2.4 Pipeline

| | | |
|----------------------------|--|-----------------------|
| \$HELGA_PIPELINE_BASE_PATH | Root path of server | //big-foot/grimmhelga |
| \$HELGA_PIPELINE_FLAVOUR | Currently active pipeline flavour. Could be deploy or sandbox etc. | deploy, sandbox etc. |

2.2.5 Houdini

| | | |
|---------------------------|--|---|
| \$HELGA_SCRIPTS_BASE_PATH | Package base path. From here you can import helga. | //big-foot/grimmhelga/Production/scripts/deploy/helga |
| \$HELGA_HDRI_PATH | Path for all HDRIs | \$HELGA_PIPELINE_BASE_PATH + /Production/2d/hdri |
| \$HELGA_OTL_PATH | Default path for helga otl | \$HELGA_PIPELINE_BASE_PATH + /Production/3d/maya/scenes/assets/otls |
| \$HELGA_FUR_PATH | Path for fur caches | \$HELGA_PIPELINE_BASE_PATH + /Production/3d/maya/cache/fur |
| \$HELGA_ALEMBIC_PATH | Path for all abc caches used in lighting, fur and sim. | \$HELGA_PIPELINE_BASE_PATH + /Production/3d/maya/cache/alembic |
| \$HELGA_TEXTURES_PATH | Path for all textures used in the helga project. | \$HELGA_PIPELINE_BASE_PATH + /Production/3d/maya/scenes/assets/textures |
| \$HELGA_RENDER_PATH | Path where all images are rendered to from 3d. | \$HELGA_PIPELINE_BASE_PATH + /Production/3d/maya/images |

Checklists

3.1 Mesh checklist

Note: Don't let Marco H. trick you.... here's what to look out for ;)

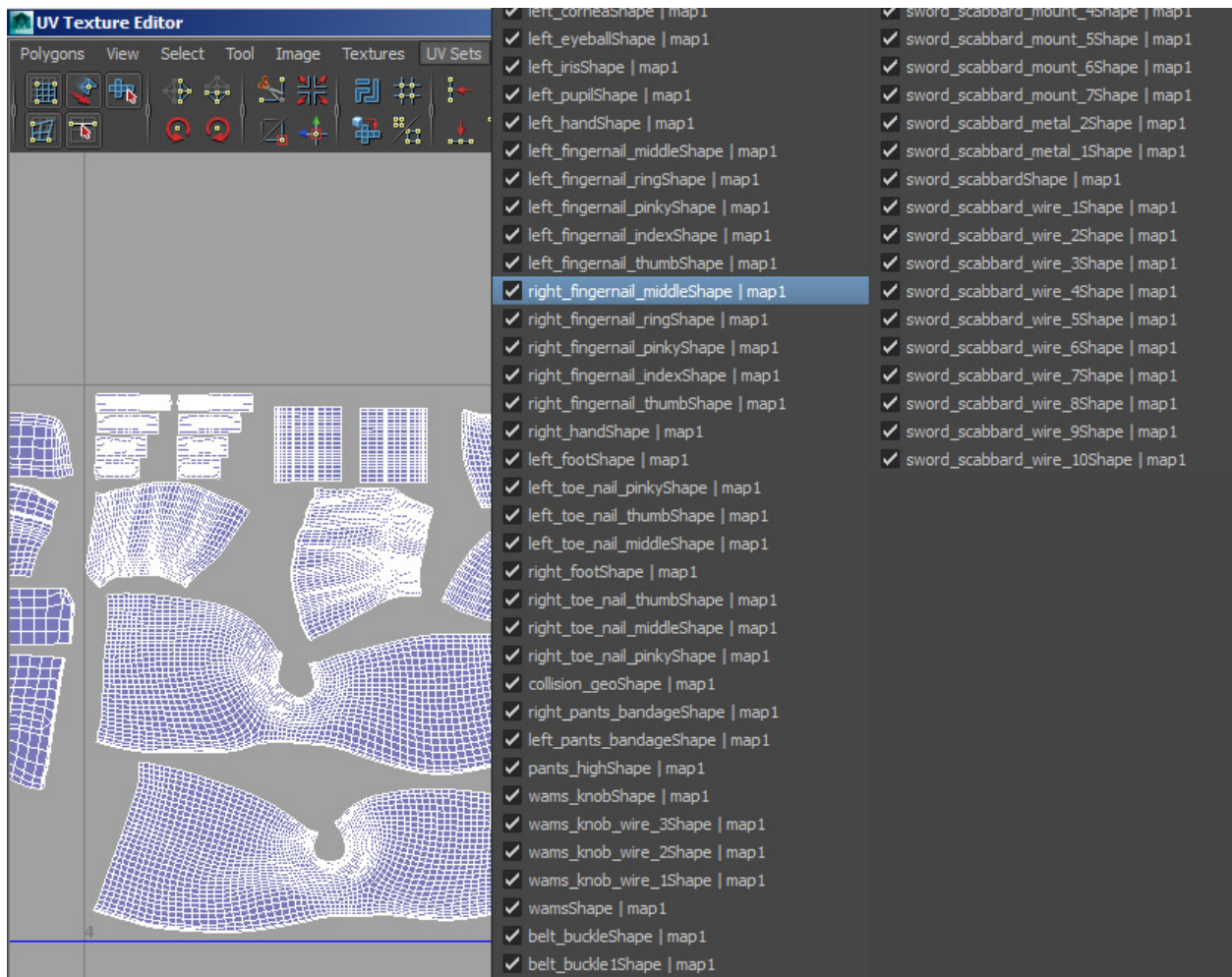
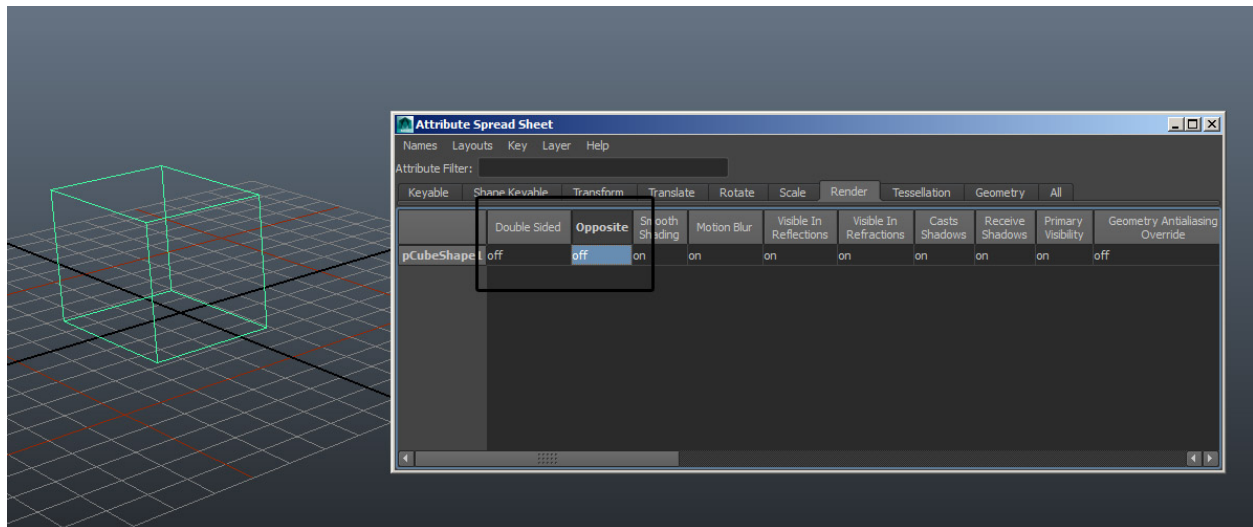
1. **Normals facing correct ?** Probably by default do Normals>Conform and Normals>Set to Face on all geo in Maya.
 2. **Is the geo frozen (Translate, Rotate, Scale) and the pivots of all objects are at (0,0,0) in worldspace?** This ensures all Alembics will be at the right position under the world later on and its also cleaner when rigging. In Maya you can run this:

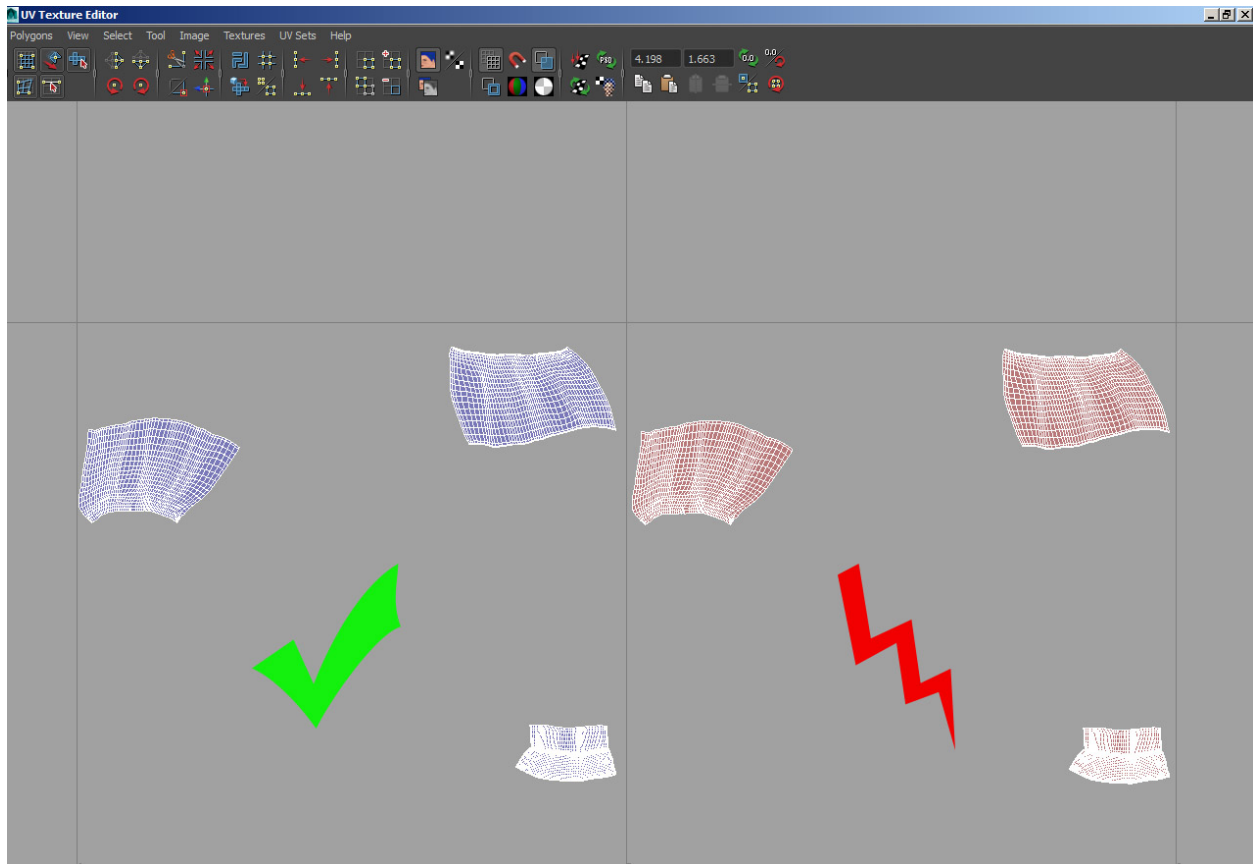

```
import pymel.core as pm

selected_nodes = pm.ls(sl = True, fl = True)

for selected_node in selected_nodes:

    #freeze
    pm.makeIdentity(selected_node, a = True)
    #reset pivot
    pm.xform(selected_node, piv = (0,0,0))
```
 3. **Doublesided and Opposite turned off?** This ensures that all faces point to the desired direction for all renderers without custom correction. In Maya you can do this in the Attribute Spreadsheet under the **Render** tab. (*The first two tabs*)
 4. **No empty UV Sets floating around and all primary UV Sets are enabled?** In case there are empty UV sets it makes sense to delete them. Also for some reason Maya likes to have its primary UV set named **map1**. While this is not always necessary, its good to follow this convention if possible.
 5. **No overlapping UVs on all the primary UV sets?** Maybe sometimes there might be reasons to do otherwise but in general its a good idea to have UV sets not overlapping, so that follicles or other rigging nodes that are placed on the surface based on UVs work correct.
 6. **UVs have the correct winding order? (Are the UV shells blue instead of red?)** This happens through flipping and freezing of geometry for example. Can be fixed with Edit UVs>Flip
 7. **Do objects have the right scale?** Default units in Maya are cm.
 8. **Only geo nodes in scene?** No history, random materials (for example from .obj import) or other nodes should pollute the scene. All geo should have the default Lambert material assigned.
-





9. Geometry usefully grouped?

`grp_geo`

`grp_body` `grp_left_leg...`

10. All geometry that is actively simulated (cloth etc.) needs to be all quads and tris! Collision geometry is fine with NGons. When in doubt ask Johannes.

3.2 Animation checklist

Note: Here are some important things for the animators amongst us. Since your output is used in simulation, lighting, shading etc. please pay attention to these points, to ensure everything is running smoothly through the pipe.

If you have questions just ask [Johannes](#), [Manuel](#) or [Timm](#).

3.2.1 Simulation Preroll

In order for the extensive cloth sim, that we want to put on top of the helga characters, there is some pre-roll time necessary. Trust me, it's not rocket science, here's how we wanna do it.

There are 2 steps that come before the character animation that are needed for the sim dudes.

1. **Frame: 901 - 951** Character is in T-Pose. The cloth settles on the T-Pose character.
2. **Frame: 951 - 1001** On frame **951** there is a keyframe of the T-Pose. From here the character morphs linearly into the start pose. Here the cloth gets ready for the real action.
3. **Frame: 1001** This is the **first** keyframe of the character animation.

We choose frame **1001** as the first keyframe of animation because it's a round **1000** frames offset from the usual frame 1.

3.3 Photogrammetry checklist

Note: Here i note the settings which seem to provide a good quality/time ratio. (For final solves). They are based on the first days of testing and finding out what works/doesnt work.

If you have questions ask [Johannes](#) or [Timm](#).

3.3.1 Solve settings (final)

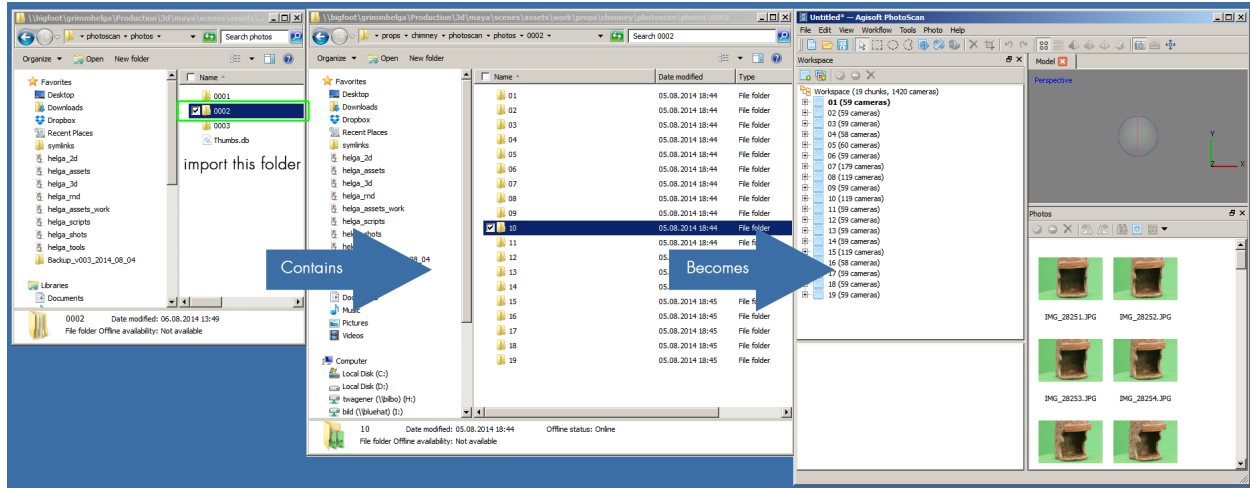
Set the settings in the **batch process dialog** to the following. Before the aligning of chunks (**second step**) it makes sense to check the solved chunks and disable those that dont work before aligning and merging chunks.

Note:

1. Import as multiple chunks
 2. **Align Photos:** High
 3. **Align Chunks:**
 - chunks count < 10: *High*
 - chunks count > 10: *Medium*
 4. **Build dense cloud:**
 - chunks count < 10: *High*
 - chunks count > 10: *Medium*
 5. **Mesh:**
 - Face count: Custom (**2.000.000** polies).
 6. **Texture:**
 - Size: 8192²
 - Format: **.jpg**
-

3.3.2 Folders to chunks

How folders in the **prop_x/photoscan/photos** dir. map to chunks in photoscan.



3.3.3 Machine list

The following machines have a Photoscan license:

- **Pegasus** (Manuel)
- **Gnocchi** (Timm)
- **Maccheroni** (Timm)
- **Adonis** (Johannes)
- **Fusilli** (Silke)
- **Gemelli** (Nicole)

3.4 Asset checklist

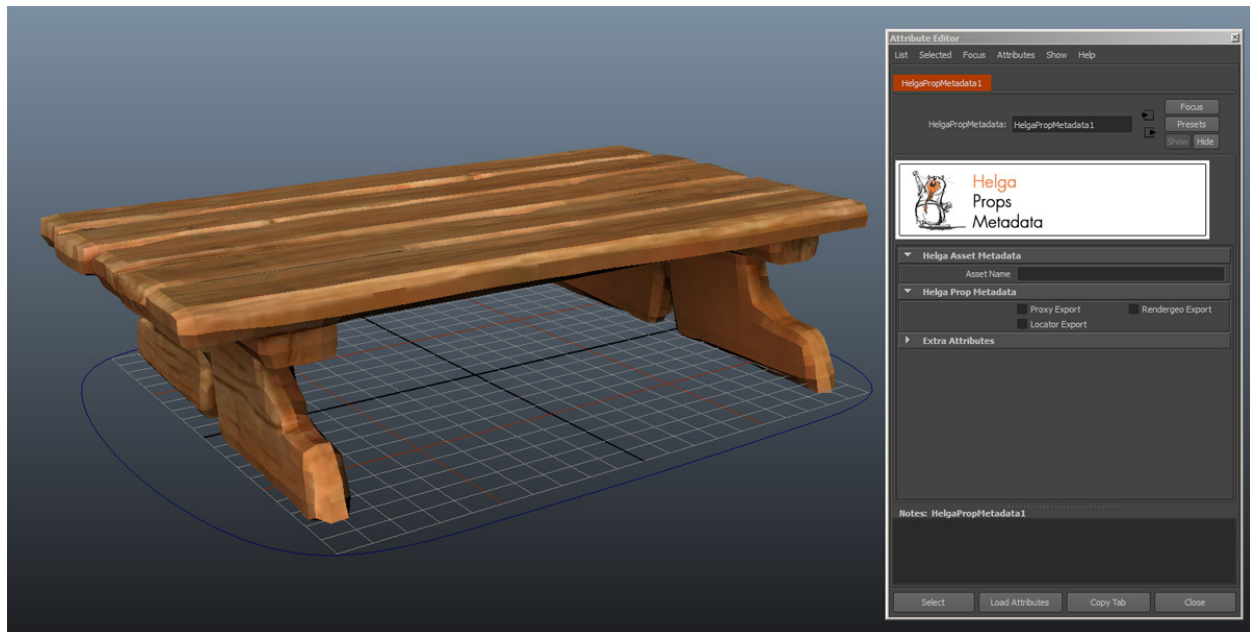
Note: In order for the smooth, customized export of assets from animation to lighting/shading, sim and everybody else, the assets need a little treatment before going in production. Here's a little list of things to take care of. It is advisable (although not needed) to use the **AssetManager** to automate a lot of these steps in Maya.

1. **Asset has blank prop or char metadata node ?** This tells the asset manager which categories you want to export. The node options are left blank, because they need to be filled in the scene as a local reference edit.

(Shot metadata nodes are embedded in the shot files, and never belong into an asset).

2. **Export Attributes attached (proxy, rendergeo, locator...)?** This tells the asset manager what the content of the categories is. Do all the export objects have the correct attributes assigned. The following attributes are typical:

- (a) helga_proxy
- (b) helga_rendergeo
- (c) helga_locator
- (d) helga_highpoly_rendergeo ← Points to the highpoly render version of the asset.

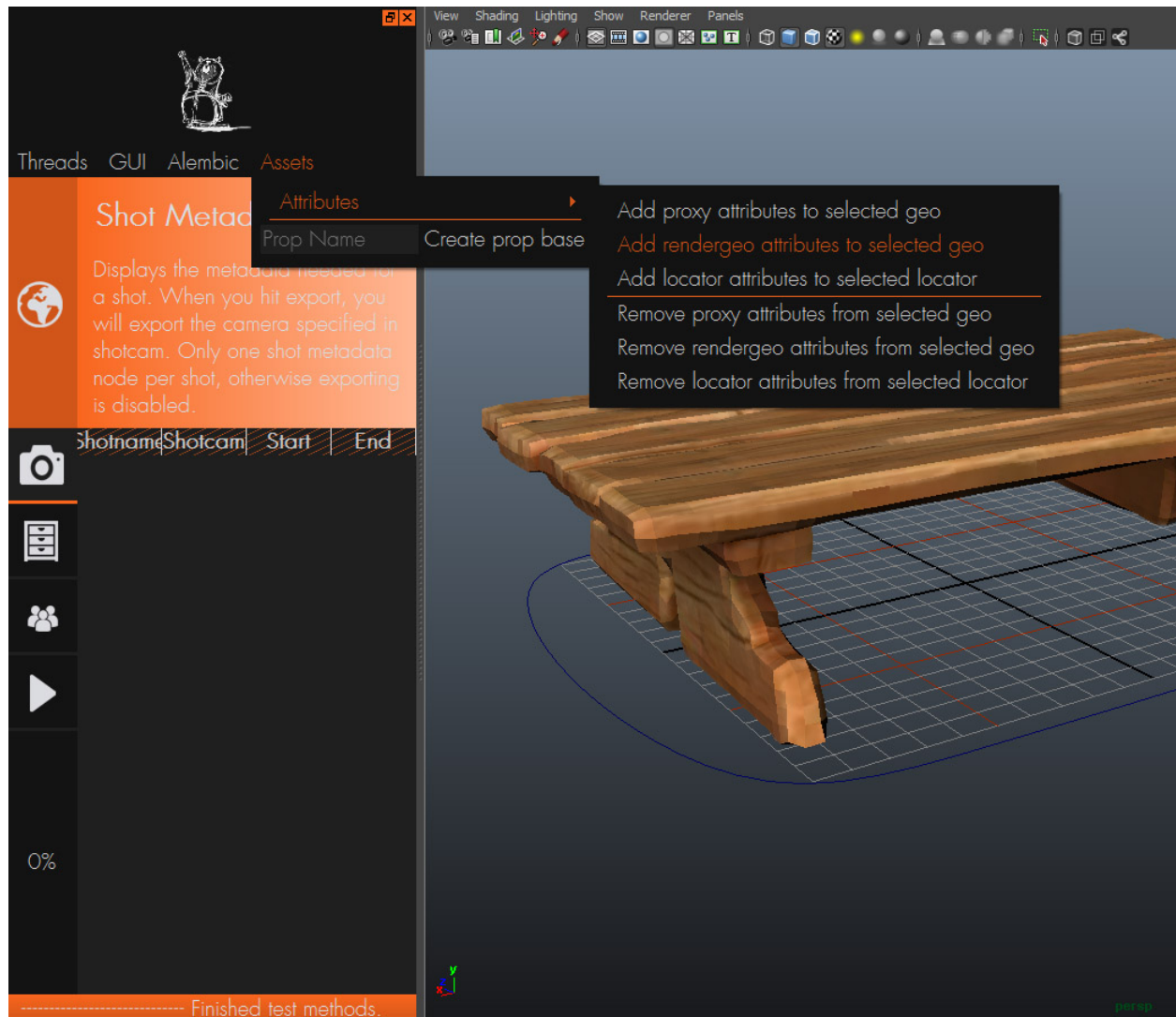


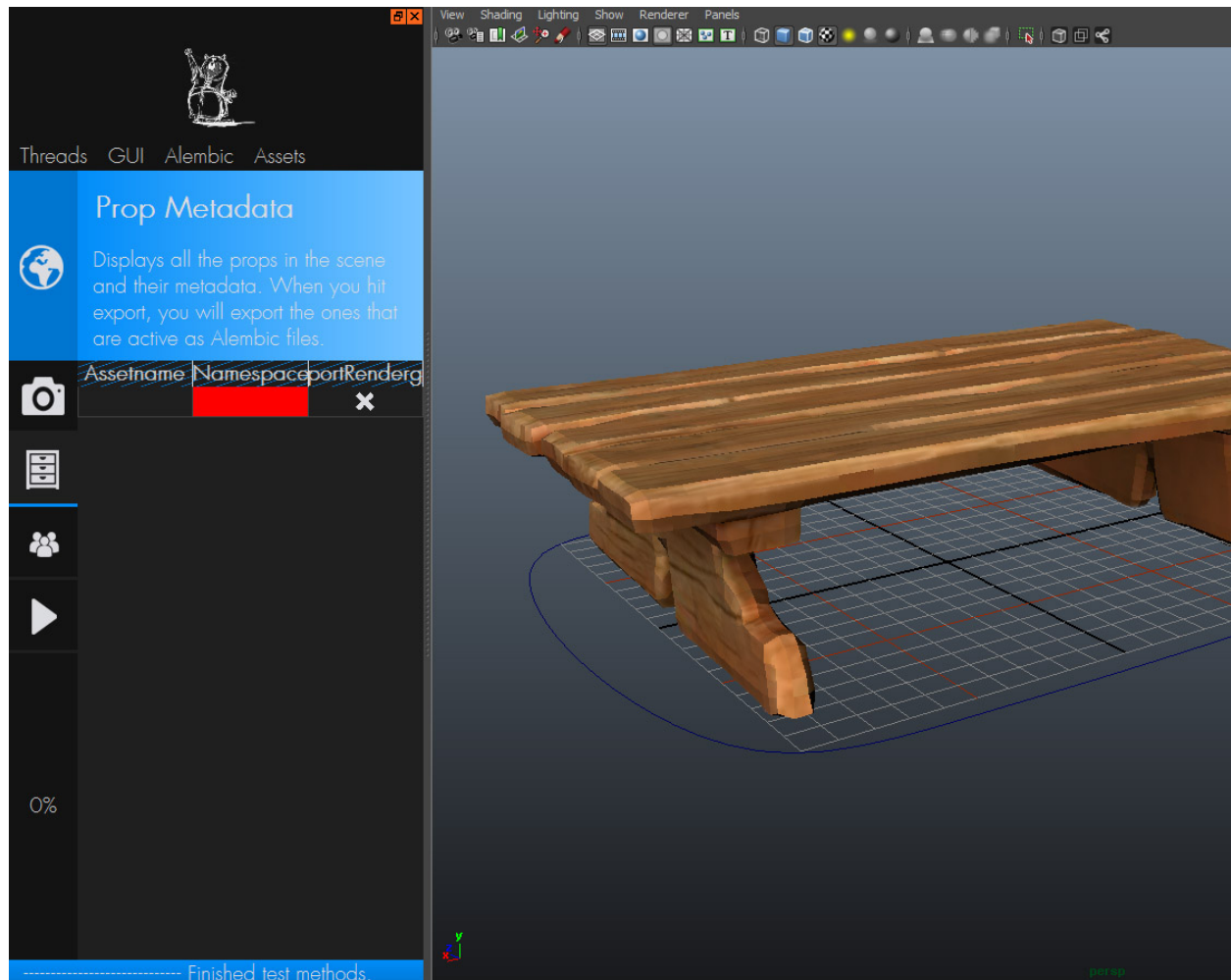
These attributes are always assigned on the **transform** nodes of the objects in Maya. No attributes are put on shape nodes by default. (Although its perfectly fine by artists to do so).

The asset manager tool provides an interface to add or remove those attributes on selected nodes:

Only objects with export attributes will be considered for the export

3. **helga_highpoly_rendergeo attribute on locator set?** The helga_highpoly_rendergeo attr. is filled out on the asset **before** referencing. It points to the highpoly render replacement geometry, usually an alembic file located in cache/alembic/highpoly_rendergeo.
4. **Does the asset have a material?** Make sure the textures are lightweight and published in the textures/props/prop_xy directory.
5. Does the asset have a rig, and are the joints and locator hidden?
6. **Did you check the asset with the AssetManager after publishing?** Always be the first to reference in your newly published asset and check if it performs with the AssetManager.





I am a

4.1 I am a ...

Here you can quickly find the important ins & outs sorted for your specific task.

4.1.1 I am a texture painter

Note:

Hi Marco, you radical machine!

Here is a quick summary of the important things for you...

Where to put your Mari, Mudbox projects or PS files?

Please work on your textures in this folder

Y:/Production/3d/maya/scenes/assets/work/chars/CHARACTERNAME/textures

Y:/Production/3d/maya/scenes/assets/work/props/PROPNAME/textures

Please version up like this:

ulfbert_textures_a_0001_mh.mra

ulfbert_textures_a_0002_mh.mra

ulfbert_textures_b_0003_mh.mra

...

chimney_textures_e_0005_mh.mud

chimney_textures_f_0006_mh.mud

Where to put the textures for the shading artist to use?

Please put the textures you want Manuel (or me) to use in

Y:/Production/3d/maya/scenes/assets/textures/CHARACTERNAME

Please name them without version (this is called *publishing* in fancy TD speech):

bump.1001.tga

bump.1002.tga

...

color.1001.tga

color.1002.tga

...

etc.

It is just important, that you overwrite your older files and dont version here, so that the textures that are used in the shaders update themselves.

What formats for textures?

Simply put: Everything is **.tga** except displacement maps which are **.exr** and maps from photoscan which are **.jpg**

Here is a link to our formats and standards

Should you have any questions, you are totally welcome to email [Manuel](#) or [me](#) ;)

4.1.2 I am a rigger

Rigging artist information

4.1.3 I am a modeler

Modeler information

4.1.4 I am a pipeline programmer

Please look at *this* and *that* .

4.1.5 I am a concept artist

Concept Artist information

4.1.6 I am a animator

Hey animation crew, here is a quick overview of the what, where and how within the *helga* animation pipeline. Dont be scared, we try to make life for you as easy as possible! Please risk a quick look at these points, and in a spin you should be ready to animate :D



Who can help if there are problems?

Direction: Marco

Animation: Cindy

Rigs: Hanna and Arash

Pipeline: Timm

Dont be afraid to ask as often as possible! Remember: Some TDs look more grim than they really are ;D

Where do i get those rigs?

You can find them here:

Y:/Production/3d/maya/scenes/assets/chars

Important: If the rigs are not there or do not behave as expected please dont fix this yourself or grab them from another location.

Instead please ask our riggers Hanna and

Arash, since its their responsibility

to supply you with correct working material (and also they are sick experts in the biz of rigging...much more than u and me)

Where are the shots?

Y:/Production/3d/maya/scenes/shots

Why is everything an asset and not just a model or rig?

Assets are nothing special, they are just models and rigs that meet special criteria. An asset has metadata for example to guarantee that it will export correctly for the lighting dudes.

The general idea is, that everything you animate (Set, Characters and Props) is an asset so a smooth functioning is guaranteed.

I want an additional asset in my shot. How?

You need another beer bottle or an empty glass to get the perfect framing? Its totally fine to reference another asset in your shot, as long as you grab them from here:

Y:/Production/3d/maya/scenes/assets/props

Those assets have been messed with by a **TD** and are guaranteed to export correctly to lighting, simulation etc.

If the asset you need isnt there, just ask me and i will try to add it.

Preroll?

After animation is approved, the simulation department (**Johannes**) will do its thing.

In order for them to do a good job, you need to supply them with a so called **Resting Pose**. The resting pose basically is a T-Pose that blends linear into your first frame of animation. You can read more about it [here](#).

5.1 API Overview

5.1.1 Foreword

First let me say that **API** is probably a little misleading since the `helga` package is mostly a collection of custom tools written in Python, which are independent of each other (like `helga.nuke.reconstruction.renderReconstructVRay`). But there might be modules in the future that are ment to be exclusively building blocks for programmers.

5.1.2 Organization

Our pipeline is split up into several kinds of packages that import from a logical hierarchy. We can add packages at any depth in the hierarchy at any time.

DCC packages

- `helga.houdini`
- `helga.maya`
- `helga.nuke`

Category packages

- `helga.maya.rendering`
- `helga.maya.arash`
- `helga.maya.hanna`
- `helga.nuke.reconstruction`
- ...

Please note the packages named after contributors. Packages without a name are assumed to be completely written by me (Timm Wagener). If you want to contribute just ask and you will get your own category package.

Warning: Please never contribute outside of your own category package!

Tool packages

- `helga.maya.rendering.createUpdateRenderElements`
 - `helga.maya.arash.clean_save`
 - `helga.nuke.reconstruction.renderReconstructVRay`
 - ...
-

5.1.3 Work and Deploy

There are two Helga packages located under:

- **production/scripts/work:**

Here is where we work and edit our scripts. **You are expected to always work here and copy your stuff over to deploy when ready.** If you work and edit in deploy to save the copy step, you will sooner or later loose work, since i will often delete and recopy the whole helga package.

- **production/scripts/deploy:**

Here is where the DCCs use our scripts and their compilers do the byte compilation (create .pyc files)

This is basically source control for the poor man, but since the helga package was initially ment to be exclusively my own sandbox we have to deal with that inconvenience.

5.2 Coding Convention

If you want to contribute to our pipeline, please let me know. There is a place for these efforts and i will try to support you with that. In return you are expected to follow our **coding standards** in order for your code to integrate consistently with the codebase and make it easier for everybody.

Our coding convention in one sentence:

Do it as in **PEP 8** or **PEP 257** and name your objects as in the [Google Python Convention](#).

Warning: The current codebase contains large portions of legacy code. While still functional it was created before the coding conventions where active. This will be refactored over time.

5.2.1 Important PEPs

There are a lot of PEPs. Some are more important than others. **You are expected to read and use them.** Often times in this document, i will just refer to the appropriate place in some PEP.

- Python Coding Style: **PEP 8**
 - Docstring Conventions: **PEP 257**
-

5.2.2 Variables, Classes, Modules, Packages etc.

For naming of objects in Python we use the Google naming convention. ([PEP 8](#) does not recommend any naming).

```
module_name, package_name, ClassName, method_name, ExceptionName, function_name,  
GLOBAL_CONSTANT_NAME, global_var_name, instance_var_name, function_parameter_name,  
local_var_name.
```

You can find the full documentation [here](#).

5.2.3 Tabs or spaces

PEP 8



5.3 Running on a different OS

Note: The helga pipeline is developed/maintained foremost on the **windows** platform. However there might be the need sometimes to switch to a different operating system like Linux. For these cases you can find a quick guide [here](#) how to start the DCCs in the correct environment, but keep in mind, that support for operating systems other than windows at the moment is **very experimental**.

Warning: This guide and method are considered temporary and might become deprecated once there is time for a refactoring of the HelgaLauncher tool that is used under windows.

5.3.1 Tested on

- CentOS7
 - Houdini: Yes
 - Nuke: No
 - Maya: Untested
-

5.3.2 Step by step guide

Please note that this guide is geared towards TDs or people that are a little familiar with git or Python, since there is a little effort involved in making it run outside of how it is ment to be run. 6 little steps to take on your own risk :D

1. **Temporary clone the helga git repo** Clone the helga repo to a temporary location. Just navigate to some temp folder and execute this snippet in your git enabled shell:

```
git clone https://github.com/timmwagener/helga.git helga
```

This should create a folder named helga at your current path and clone the helga repository in it.

2. **Helga folder structure** Now in your temporary helga repo navigate to *helga/general/setup/direct_access/supplementary_material/folder_structure*. There you will find a zip file called **helga.zip**, that contains the helga folder structure. Extract this zip to where you want the helga pipeline and project to be located. **PS:** The helga folder structure is based on the Maya project structure which might be unfamiliar to non Maya users.

3. **The real clone - work repository** Now its time to do the real clone of the helga repo and fit it into the new folder structure. Navigate to *your_root/Production/scripts/work* and execute

```
git clone https://github.com/timmwagener/helga.git helga
```

there again. This is the repo you will keep up-to-date and maintain and always copy into sandbox or deploy when you want to publish changes to production.

4. **Start procedure for Houdini, Maya & Nuke** Now its time to adjust the startup module for your dccc (Supported are Maya 2014, Nuke 8.x and Houdini 13.0.xxx). Please go to *Production/scripts/sandbox/helga/helga/general/setup/direct_access/yaml* where you will find a file called **pipeline_base_data.yaml**. Open it with a text editor, for example sublime text or notepad and adjust the following values:

- (a) **PIPELINE_BASE_PATH:** “path/to/where/Production/and/Organization/folders/are/under”
- (b) **PIPELINE_FLAVOUR:** “deploy” <– Pick either **sandbox** or **deploy**. This is the folder name where the script environment you want to use is under.
- (c) **MAYA_EXE:** “path/to/maya.exe”
- (d) **NUKE_EXE:** “path/to/Nuke8.0.exe”
- (e) **HOUDINI_EXE:** “path/to/houdinifx.exe” (or houdini.exe)

Save and you are good.

5. **Pipeline repo production copies - sandbox and deploy** Those two fellas could also be named experimental and stable, beta or gold.....you get the idea. Just go and copy the whole helga folder, in which you just did modifications, and `__init__.py` file from work into sandbox and deploy.

6. **Finally** Test if the application runs correctly. Navigate to *deploy|sandbox/helga/helga/general/setup/direct_access* and type the following in a python enabled shell:

```
#Houdini:
python -c 'import direct_access;reload(direct_access);direct_access.run("houdini")'
#Maya:
python -c 'import direct_access;reload(direct_access);direct_access.run("maya")'
#Nuke:
python -c 'import direct_access;reload(direct_access);direct_access.run("nuke")'
```

If everything went well, your DCC **should** start within the helga pipeline environment (either sandbox or deploy)

Hopefully that was not too confusing ;) Please remember that you now have the pathes and setup in place, the structure if you will, but you still need a lot of content like .otls for example. In your own production you would of course populate the pipeline with your own scripts, plugins, digital assets, gizmos etc.

Good luck :D

API Documentation

| | |
|----------------------------|-----------------------|
| <code>helga.houdini</code> | helga.houdini package |
| <code>helga.maya</code> | helga.maya package |
| <code>helga.nuke</code> | helga.nuke package |
| <code>helga.general</code> | helga.general package |

6.1 helga.houdini package

Helga script environment for Houdini. Here you can find scripts, tools and plugins used with Houdini.

6.2 Packages

| | |
|-------------------------------------|-----------------------------|
| <code>helga.houdini.arash</code> | |
| <code>helga.houdini.hanna</code> | |
| <code>helga.houdini.johannes</code> | |
| <code>helga.houdini.manuel</code> | |
| <code>helga.houdini.setup</code> | helga.houdini.setup package |

6.2.1 helga.houdini.setup package

Module contents

helga.houdini.setup package

This package is not necessarily a traditional package. In here is stuff like the central pipeline third party *scripts* and *plugins* folders. Also in here are scripts and icons used to setup our pipeline houdini.

Author: Timm Wagener

6.3 helga.maya package

Helga script environment for Maya. Here you can find scripts, tools and plugins used with Maya.

6.4 Packages

| | |
|-----------------------------------|------------------------------|
| <code>helga.maya.animation</code> | helga.maya.animation package |
| <code>helga.maya.arash</code> | helga.maya.arash package |
| <code>helga.maya.hanna</code> | helga.maya.hanna package |
| <code>helga.maya.johannes</code> | helga.maya.johannes package |
| <code>helga.maya.manuel</code> | helga.maya.manuel package |
| <code>helga.maya.rendering</code> | helga.maya.rendering package |
| <code>helga.maya.rigging</code> | helga.maya.rigging package |
| <code>helga.maya.setup</code> | helga.maya.setup package |

6.4.1 helga.maya.animation package

All packages and modules belonging to animation in Maya can be found here.

Author: Timm Wagener

6.4.2 Packages

| | |
|---|------------------------------|
| <code>helga.maya.animation.bakeAnimation</code> | Bake animation tools Package |
|---|------------------------------|

Bake animation tools Package

Bake animation for objects in Maya.

Author: Timm Wagener *Version:* 0.1

Submodules

bakeAnimation

Module with methods to bake the movement/channels of one object to another. Transform baking can be either in local or world space. This can be usefull to bake camera movement resulting from hierarchies into camera movement without hierarchies etc.

Usage

```
from helga.maya.animation.bakeAnimation import bakeAnimation
reload(bakeAnimation)

#Create instance
bakeAnimationInstance = bakeAnimation.BakeAnimation()
#Select object_a and object_b and execute bakeAnimationTransformOnly
#to bake the animation from object_a to object_b for the frames 1 - 100
bakeAnimationInstance.bakeAnimationTransformOnly(1, 100)
```

class `helga.maya.animation.bakeAnimation.bakeAnimation.BakeAnimation`

bakeAnimation (*animationStartTime=0, animationEndTime=0*)

Bake animation of two objects (or master - slave pairs). Bakes animation of all channels that are *keyable* and *not locked*. Also the channels need to be presents in both objects. The selection order is the same as with constraints:

- 1.Master: Source of Movement
- 2.Slave: Gets baked to movement of master

Parameters

- **animationStartTime** (*int*) – Beginning of sequence to bake
- **animationEndTime** (*int*) – End of sequence to bake

bakeAnimationTransformOnly (*animationStartTime=0, animationEndTime=0, localSpace=False*)

Bake animation of two objects but only bake channels that contribute to the transformation matrix (translate, rotate, scale). Shear is omitted since it has never been used in the history of animation. The selection order is the same as with constraints:

- 1.Master: Source of Movement
- 2.Slave: Gets baked to movement of master

Parameters

- **animationStartTime** (*int*) – Beginning of sequence to bake
- **animationEndTime** (*int*) – End of sequence to bake
- **localSpace** (*bool*) – If True, only the immediate values of the object are baked. If False bake complete transformation hierarchy (WorldSpace).

getSelection ()

setAnimationRange (*startTime, endTime, startRange, endRange*)

6.4.3 helga.maya.arash package

All packages and modules created by Arash Hosseini can be found here. For bugs, questions or help please ask :

Author: [Arash Hosseini](#) **Pipeline Lead:** [Timm Wagener](#)

6.4.4 Packages

| | |
|---|---------------------|
| <code>helga.maya.arash.clean_save</code> | Clean save Package |
| <code>helga.maya.arash.helga_login</code> | Helga login Package |

Clean save Package

Path checking and saving convenience for objects in Maya.

Author: [Arash Hosseini](#) *Version:* 0.1

Submodules

clean_save

Tool to support our naming convention by checking and convenience saving functionality.

Usage

```
from helga.maya.arash.clean_save import clean_save
reload(clean_save)
```

```
#run
clean_save.run()
```

Author: Arash Hosseini

```
class helga.maya.arash.clean_save.clean_save.cleanSave_UI (windowsWidth=360, win-
                                dowsHeight=400)
```

```
    artistField (*args)
    changeArtistColor (*args)
    changeColors (*args)
    changeDepartmentColor (*args)
    changeFaktorColor (*args)
    changeNameColor (*args)
    changeVersionColor (*args)
    cleanTextFields (*args)
    debug (*args)
    deletewindow (*args)
    depField (*args)
    dontSave (*args)
    faktorField (*args)
    file_path (fileName, fileType)
    login_file (*args)
    mainUI (windowsWidth=355, windowsHeight=400)
    nameField (*args)
    overWriteFile ()
    saveLogFile (*args)
    saveScene (*args)
    saveSceneOverwrite (*args)
    sceneCheck (*args)
```

```
sendDebug (*args)
upversion (*args)
versionField (*args)
wrongChangeArtistColor (*args)
wrongChangeDepartmentColor (*args)
wrongChangeFaktorColor (*args)
wrongChangeNameColor (*args)
wrongChangeSaveColor (*args)
wrongChangeVersionColor (*args)
helga.maya.arash.clean_save.clean_save.run ()
    Default method for script execution.
```

Helga login Package

GUI for the auto-rigger for the Helga characters.

Author: [Arash Hosseini](#) *Version:* 0.1

Submodules

helga_login

GUI for the auto-rigger for the Helga characters.

Usage

```
from helga.maya.arash.helga_login import helga_login
reload(helga_login)

#run
helga_login.run()
```

Author: [Arash Hosseini](#)

```
class helga.maya.arash.helga_login.helga_login.Base
    Bases: object
    delete_dock_control (name)
    delete_window (name)
    fk2ik_ulfbert_character (name, *args)
    ik2fk_ulfbert_character (name, *args)
    quit_UI_window (name)
    saveValue (nodeName, attrName, object_node)
```

```
select_ulfbert_character_controller (name, *args)
select_ulfbert_hand_controller (name, *args)
setValue (nodeName, attrName, node)
set_name_save_pose (*args)
class helga.maya.arash.helga_login.helga_login.Color
    Bases: object
    blue_a = (0.3764705882352941, 0.7254901960784313, 0.8117647058823529)
    colorConstant = 255.0
    gray_a = (0.5411764705882353, 0.47058823529411764, 0.5019607843137255)
    green_a = (0.3764705882352941, 0.8117647058823529, 0.6431372549019608)
    orange_a = (1.0, 0.7490196078431373, 0.25098039215686274)
    red = (1.0, 0.0, 0.0)
class helga.maya.arash.helga_login.helga_login.Helga_cms_anim_UI
    Bases: helga.maya.arash.helga_login.helga_login.Base
    character_body_button (*args)
    character_choice (*args)
    character_hand_button (*args)
    delete_ulfbert_character_UIs (*args)
    delete_ulfbert_hand_UI (*args)
    set_image_sequenz (path)
    startSubThread ()
class helga.maya.arash.helga_login.helga_login.Helga_cms_login_UI (windows_width=350,
                                                                    win-
                                                                    dows_height=200)
    Bases: helga.maya.arash.helga_login.helga_login.Base
    login_UI (*args)
    login_file (*args)
    run_helga_cms_anim_UI (*args)
    run_helga_cms_td_UI (*args)
class helga.maya.arash.helga_login.helga_login.Helga_cms_td_UI
    Bases: helga.maya.arash.helga_login.helga_login.Base
    check_scene_jnt (scene_joints)
    check_scene_polygone (scene_polygone)
    color_Editor (name)
    create_check_module_arm (*args)
    create_check_module_biped (*args)
    create_check_module_head (*args)
    create_check_module_leg (*args)
```



```
create_check_module_spine (*args)
delete_dock_td_control (name)
delete_module (*args)
helga_cms_td_main_UI (child_width=295, parent_width=300)
job_reload_UI (*args)
populate_cms_jnt (*args)
populate_polygone (*args)
populate_scene_jnt (*args)
select_module (name)
show_helper (message)
skinning (*args)
tool_jelly (*args)
class helga.maya.arash.helga_login.helga_login.Helja_body_UI (helja_parent_width=500,
                                                             helja_parent_height=500,
                                                             helja_child_width=480,
                                                             helja_child_height=480)
    Bases: helga.maya.arash.helga_login.helga_login.Base
    delete_helja_character_body_UI (*args)
    helja_body_main_UI (*args)
class helga.maya.arash.helga_login.helga_login.Load_character_pose_UI (character_load_parent_width=320,
                                                                         character_load_
                                                                         ter_load_parent_height=660,
                                                                         character_load_
                                                                         ter_load_child_width=315,
                                                                         character_load_
                                                                         ter_load_child_height=465)
    Bases: helga.maya.arash.helga_login.helga_login.Base
    change_saved_image (*args)
    character_load_pose_main_UI (*args)
    character_load_pose_namespace (*args)
    laod_pose_open_directory (*args)
    load_pose (*args)
    load_saved_poses (*args)
    refresh_load_pose_namespace (*args)
    replace_current_image (*args)
class helga.maya.arash.helga_login.helga_login.Ritter_body_UI (ritter_parent_width=500,
                                                                rit-
                                                                ter_parent_height=500,
                                                                rit-
                                                                ter_child_width=480,
                                                                rit-
                                                                ter_child_height=480)
```

Bases: `helga.maya.arash.helga_login.helga_login.Base`

`delete_ritter_character_body_UI (*args)`

`ritter_body_main_UI (*args)`

`class helga.maya.arash.helga_login.helga_login.Save_character_pose_UI (character_parent_width=320, character_parent_height=700, character_child_width=505, character_child_height=295)`

Bases: `helga.maya.arash.helga_login.helga_login.Base`

`change_pose_animation_status (*args)`

`change_pose_time_value_status (*args)`

`character_save_pose_main_UI (*args)`

`check_perspective (*args)`

`refresh_save_pose_namespace (*args)`

`save_pose (*args)`

`save_pose_action (*args)`

`save_pose_action_b (*args)`

`save_pose_clean_list (*args)`

`save_pose_export_animation (*args)`

`save_pose_export_animation_action (*args)`

`save_pose_image_action (*args)`

`save_pose_image_sub_action (*args)`

`save_pose_radioButton_Current_namespace (*args)`

`save_pose_radioButton_custom_namespace (*args)`

`save_pose_screen_animation (*args)`

`save_pose_screen_save (*args)`

`select_character_L_arm (*args)`

`select_character_L_fingers (*args)`

`select_character_L_leg (*args)`

`select_character_R_arm (*args)`

`select_character_R_fingers (*args)`

`select_character_R_leg (*args)`

`select_character_main_body (*args)`

`select_character_selected_member (*args)`

```
class helga.maya.arash.helga_login.helga_login.Ulfbert_body_UI (ulfbert_parent_width=500,
                                                                ulf-
                                                                bert_parent_height=620,
                                                                ulf-
                                                                bert_child_width=495,
                                                                ulf-
                                                                bert_child_height=495)

Bases: helga.maya.arash.helga_login.helga_login.Base

call_load_pose (*args)
call_save_pose (*args)
delete_ulfbert_character_body_UI (*args)
display_all_member_lists (*args)
hide_all_bends (*args)
hide_all_controls (*args)
hide_left_arm (*args)
hide_left_fingers (*args)
hide_left_leg (*args)
hide_left_side (*args)
hide_right_arm (*args)
hide_right_fingers (*args)
hide_right_leg (*args)
hide_right_side (*args)
show_all_bends (*args)
show_all_controls (*args)
ulfbert_L2R_rotate (*args)
ulfbert_L2R_scale (*args)
ulfbert_L2R_translate (*args)
ulfbert_L_Leg_fkik (*args)
ulfbert_L_arm_fkik (*args)
ulfbert_R2L_rotate (*args)
ulfbert_R2L_scale (*args)
ulfbert_R2L_translate (*args)
ulfbert_R_Leg_fkik (*args)
ulfbert_R_arm_fkik (*args)
ulfbert_about (*args)
ulfbert_body_graph_editor (*args)
ulfbert_body_main_UI (*args)
ulfbert_change_namespace_text_color_green (*args)
ulfbert_change_namespace_text_color_red (*args)
```

```
ulfbert_create_reference (*args)
ulfbert_debug (*args)
ulfbert_get_rotate_data (name)
ulfbert_get_scale_data (name)
ulfbert_get_trans_data (name)
ulfbert_key_selected (*args)
ulfbert_mirror_L2R_action (*args)
ulfbert_mirror_R2L_action (*args)
ulfbert_namespace (*args)
ulfbert_reference_editor (*args)
ulfbert_reset_body_controller (*args)
ulfbert_reset_pose (*args)
    ulfbert_reset_ui
ulfbert_reset_rotate (*args)
ulfbert_reset_scale (*args)
ulfbert_reset_translate (*args)
ulfbert_select_spine (*args)
ulfbert_sendDebug (*args)
ulfbert_set_rotateX_data (name)
ulfbert_set_rotateY_data (name)
ulfbert_set_rotateZ_data (name)
ulfbert_set_scaleX_data (name)
ulfbert_set_scaleY_data (name)
ulfbert_set_scaleZ_data (name)
ulfbert_set_transX_data (name)
ulfbert_set_transY_data (name)
ulfbert_set_transZ_data (name)

class helga.maya.arash.helga_login.helga_login.Ulfbert_hand_UI (ulfbert_hand_parent_width=455,
                                                                ulf-
                                                                bert_hand_parent_height=320,
                                                                ulf-
                                                                bert_hand_child_width=450,
                                                                ulf-
                                                                bert_hand_child_height=300)

Bases: helga.maya.arash.helga_login.helga_login.Base
call_hand_load_pose (*args)
call_hand_save_pose (*args)
refresh_hand_pose_namespace (*args)
ulfbert_hand_body_main_UI (*args)
```

```
ulfbert_hand_graph_editor(*args)
ulfbert_hand_set_namespace(*args)
ulfbert_select_L_index(*args)
ulfbert_select_L_middle(*args)
ulfbert_select_L_pink(*args)
ulfbert_select_L_ring(*args)
ulfbert_select_L_thumb(*args)
ulfbert_select_R_index(*args)
ulfbert_select_R_middle(*args)
ulfbert_select_R_pink(*args)
ulfbert_select_R_ring(*args)
ulfbert_select_R_thumb(*args)
ulfbert_select_all_L_fingers(*args)
ulfbert_select_all_R_fingers(*args)
```

```
helga.maya.arash.helga_login.helga_login.run()
Standardized run() method. Used to call modules functionality
```

6.4.5 helga.maya.hanna package

Module contents

helga.maya.hanna package

All packages and modules created by Hanna Binswanger can be found here. For bugs, questions or help please ask :

Author: [Hanna Binswanger](#) **Pipeline Lead:** [Timm Wagener](#)

6.4.6 helga.maya.johannes package

Module contents

helga.maya.johannes package

All packages and modules created by Johannes Franz can be found here. For bugs, questions or help please ask :

Author: [Johannes Franz](#) **Pipeline Lead:** [Timm Wagener](#)

6.4.7 helga.maya.manuel package

Module contents

helga.maya.manuel package

All packages and modules created by Manuel Seifert can be found here. For bugs, questions or help please ask :

Author: Manuel Seifert **Pipeline Lead:** Timm Wagener

6.4.8 helga.maya.rendering package

All packages and modules belonging to rendering in Maya can be found here.

Author: Timm Wagener

6.4.9 Packages

| | |
|--|---------------------------------------|
| <code>helga.maya.rendering.createUpdateRenderElements</code> | Create/Update Render Elements Package |
| <code>helga.maya.rendering.renderPathes</code> | Set Render Pathes Package |
| <code>helga.maya.rendering.setExrMetadata</code> | Set Exr Metadata Package |
| <code>helga.maya.rendering.setVrayRenderGlobals</code> | Set VRay Render Globals Package |

Create/Update Render Elements Package

Automatic setup of VRay Render Elements in Maya. Modules of this package set up light, frame-buffer, data and shadow passes automatically. Light is split into light select render elements. This is the initial automatic setup which sets up multichannel .exrs that should automatically rebuild in comp using `helga.nuke.reconstruction.renderReconstructVray`

Author: Timm Wagener *Version:* 0.1

Submodules

createDataPasses

Automatic setup of the following VRay Render Elements in Maya. They are also the names of the channels in the multichannel exr. You can call this function as often as you want. It will not duplicate existing elements, only add missing ones.

The following elements are created:

- Normals
- BumpNormals
- VelocityFiltered
- VelocityUnfiltered
- ZDepth.filtered
- ZDepth.unfiltered
- EtAmbOcc (Ambient Occlusion from extra tex)
- EtWorldPos (World Position from extra tex)
- EtSTMap
- EtFresnelLarge
- EtFresnelSmall

- `RenderId.index`
-

Usage

```
from helga.maya.rendering.createUpdateRenderElements import createDataPasses
reload(createDataPasses)

#Create instance
createDataPassesInstance = createDataPasses.CreateDataPasses()
#Create/Update data passes
createDataPassesInstance.createDataPasses()
```

```
class helga.maya.rendering.createUpdateRenderElements.createDataPasses.CreateDataPasses
```

```
REWithAttrAndValueExists (attrName, attrValue)
createBumpNormalsRE ()
createDataPasses ()
    Function to create/update data passes.
createEtAmbOccRE ()
createEtFresnelLargeRE ()
createEtFresnelSmallRE ()
createEtSTMapRE ()
createEtWorldPosRE ()
createNormalsRE ()
createRamp (rampDirection=1)
createRenderElement (renderElementName)
createRenderIdRE ()
createVelocityREFiltered ()
createVelocityREUnfiltered ()
createZDepthREFiltered ()
createZDepthREUnfiltered ()
vrayLoaded ()
```

`createFrameBufferContributions`

Automatic setup of the following V-Ray Render Elements in Maya. They are also the names of the channels in the multichannel exr. You can call this function as often as you want. It will not duplicate existing elements, only add missing ones.

The following elements are created:

- Diffuse
- Reflection

- Refraction
 - Specular
 - Subsurface
-

Usage

```
from helga.maya.rendering.createUpdateRenderElements import createFrameBufferContributions
reload(createFrameBufferContributions)
```

```
#Create instance
```

```
createFrameBufferContributionsInstance = createFrameBufferContributions.CreateFrameBufferContributionsInstance()
```

```
#Create/Update framebuffer passes
```

```
createFrameBufferContributionsInstance.createFrameBufferContributions()
```

```
class helga.maya.rendering.createUpdateRenderElements.createFrameBufferContributions.CreateFrameBufferContributionsInstance:
```

```
    REWithAttrAndValueExists (attrName, attrValue)
```

```
    createDiffuseRE ()
```

```
    createFrameBufferContributions ()
```

```
        Function to create/update framebuffer passes (see above for AOV types).
```

```
    createReflectionRE ()
```

```
    createRefractionRE ()
```

```
    createRenderElement (renderElementName)
```

```
    createSpecularRE ()
```

```
    createSubsurfaceRE ()
```

```
    vrayLoaded ()
```

createShadowPass

Automatic setup of the following VRay Render Elements in Maya. They are also the names of the channels in the multichannel exr. You can call this function as often as you want. It will not duplicate existing elements, only add missing ones.

The following elements are created:

- RawShadow
 - Shadow
-

Usage

```
from helga.maya.rendering.createUpdateRenderElements import createShadowPass
reload(createShadowPass)
```

```
#Create instance
```

```
createShadowPassInstance = createShadowPass.CreateShadowPass()
```



```
#Create/Update shadow passes
createShadowPassInstance.createShadowPass()
```

class helga.maya.rendering.createUpdateRenderElements.createShadowPass.**CreateShadowPass**

```
REWithAttrAndValueExists (attrName, attrValue)
createRawShadowRE ()
createRenderElement (renderElementName)
createShadowPass ()
    Function to create/update shadow passes (see above for AOV types).
createShadowRE ()
vrayLoaded ()
```

createUpdateLightRenderElements

Automatic setup of the following VRay Render Elements in Maya. They are also the names of the channels in the multichannel exr. You can call this function as often as you want. It will not duplicate existing elements, only add missing ones. The light is split into light select elements and GI. To use these passes in Nuke with the `helga.nuke.reconstruction.renderReconstruct` module you need a diffuse pass from `helga.maya.rendering.createUpdateRenderElements.createFrameBufferContributions`.

The following elements are created:

- Gi
 - Lighting
 - SelfIllum
 - Light Selects for each light where possible
-

Usage

```
from helga.maya.rendering.createUpdateRenderElements import createUpdateLightRenderElements
reload(createUpdateLightRenderElements)
```

```
#Create instance
createUpdateLightRenderElementsInstance = createUpdateLightRenderElements.CreateUpdateLightRenderElements
#Create/Update shadow passes
createUpdateLightRenderElementsInstance.createUpdateLightRenderElements()
```

class helga.maya.rendering.createUpdateRenderElements.createUpdateLightRenderElements.**CreateU**

```
REWithAttrAndValueExists (attrName, attrValue)
createGiRE ()
createLSREForEachLight (lightsList)
createLightingRE ()
```

```
createRenderElement (renderElementName)  
createSelfIllumRE ()  
createUpdateLSREs ()  
createUpdateLightRenderElements ()  
    Function to create/update light render elements (see above for AOV types).  
deleteAllLSRElementSets ()  
getAllSceneLights ()  
getRenderElementSetsByPrefix (prefix)  
vrayLoaded ()
```

createUpdateMmRenderElements

Automatic setup of multimatte V-Ray Render Elements in Maya. The script parses the scene for object ids on objects and creates multi matte render elements from them. The object ids are sorted numerically. You can call this function as often as you want. It will not duplicate existing elements, only add missing ones.

Usage

```
from helga.maya.rendering.createUpdateRenderElements import createUpdateMmRenderElements  
reload(createUpdateMmRenderElements)  
  
#Create instance  
createUpdateMmRenderElementsInstance = createUpdateMmRenderElements.CreateUpdateMmRenderElements()  
#Create/Update shadow passes  
createUpdateMmRenderElementsInstance.createUpdateMmRenderElements()
```

```
class helga.maya.rendering.createUpdateRenderElements.createUpdateMmRenderElements.CreateUpdateMmRenderElements
```

```
REWithAttrAndValueExists (attrName, attrValue)  
createMultiMattes (objectIdList)  
createRenderElement (renderElementName)  
createUpdateMultiMatteElements ()  
    Function to create/update multi matte render elements.  
deleteCurrentMultiMatteElements ()  
getRenderElementsByPrefix (prefix)  
getSceneObjectIdList ()  
renameMultiMatteRE (mMRE)  
vrayLoaded ()
```

vrayGlobals

Globals module to hosts constants for the `helga.maya.rendering.createUpdateRenderElements` package.

Usage

```
from helga.maya.rendering.createUpdateRenderElements import vrayGlobals
reload(vrayGlobals)

#my_fancy_prefix
my_fancy_prefix = vrayGlobals.PREFIX
```

`helga.maya.rendering.createUpdateRenderElements.vrayGlobals.PREFIX = 'gh'`
Prefix for the channels in multichannel exrs. For example: ghDiffuse, ghReflection

Set Render Pathes Package

Automatic setup of the correct rendering pathes for images according to Pipeline standards. It is ment to work with VRay for Maya exclusively.

Author: Timm Wagener Version: 0.1

Submodules

renderPathes

Automatic setup of the correct rendering pathes for images according to Pipeline standards. It is ment to work with VRay for Maya exclusively. This is a very simple module, it basically only determines where renderings will go to.

Usage

```
from helga.maya.rendering.renderPathes import renderPathes
reload(renderPathes)

#Create instance
renderPathesInstance = renderPathes.RenderPathes()
#set render pathes for final lighting
renderPathesInstance.setPathLighting()
#set render pathes for testing
renderPathesInstance.setPathTesting()
```

```
class helga.maya.rendering.renderPathes.renderPathes.RenderPathes
```

```
    setPathLighting()
```

Render path for images. The output here should be copied and used by compositing.

setPathTesting()

Render path for testing images. The output here is purely for your own testing and could be deleted anytime. If you want to keep it, then copy it to your rnd folder.

vrayLoaded()

Check if the VRay for Maya Plugin is loaded.

vrayRenderSettingsNode()

Get Vray rendersettings node

Set Exr Metadata Package

Package that holds functionality to parse the scene and set exr metadata. It is ment to work with VRay for Maya exclusively.

Author: Timm Wagener *Version:* 0.1

Submodules

setExrMetadata

Module that will parse the scene and add exr metadata to the rendersettings node. The module is designed to work with VRay for Maya exclusively. You can import and run it as a Pre-Frame script while batch rendering (on the farm).

Usage

```
from helga.maya.rendering.setExrMetadata import setExrMetadata
reload(setExrMetadata)
```

```
#run
setExrMetadata.run()
```

Warning: Module is known to cause crashes when rendering. Use with caution

```
helga.maya.rendering.setExrMetadata.setExrMetadata.run()
    General and only method to invoke to start adding metadata
```

Set VRay Render Globals Package

Package that holds functionality to set the VRay Render Globals according to pipeline standards.

Author: Timm Wagener *Version:* 0.1

Submodules

setVrayRenderGlobals

Set the VRay Render Globals according to pipeline standards.

Usage

```
from helga.maya.rendering.setVrayRenderGlobals import setVrayRenderGlobals
reload(setVrayRenderGlobals)

#Create instance
setVrayRenderGlobalsInstance = setVrayRenderGlobals.SetVrayRenderGlobals()

#set render globals completely
setVrayRenderGlobalsInstance.setVrayRenderSettings()
#set render globals linear light settings only
setVrayRenderGlobalsInstance.setVrayRenderSettingsLinear()
```

class `helga.maya.rendering.setVrayRenderGlobals.setVrayRenderGlobals.SetVrayRenderGlobals`

setVrayRenderSettings (*set_pre_render_mel_script=False*)

Set Vray Rendersettings

Parameters **set_pre_render_mel_script** (*bool*) – Set the pre render MEL script. Default is False.

setVrayRenderSettingsLinear ()

Set Vray Rendersettings, linear light settings only

setVrayRenderSettingsParameter (*linearOnly=False, set_pre_render_mel_script=False*)

Set Vray Rendersettings. Never use this method directly.

vrayLoaded ()

Return true or false whether or not Vray is loaded

vrayRenderSettingsNode ()

Get Vray rendersettings node

6.4.10 helga.maya.rigging package

Module contents

helga.maya.rigging package

All packages and modules belonging to rigging in Maya can be found here.

Author: [Timm Wagener](#)

6.4.11 helga.maya.setup package

This package is not necessarily a traditional package. In here is stuff like the central pipeline third party *scripts* and *plugins* folders. Also in here is the *userSetup.py* as well as shelves and icon folders.

Author: Timm Wagener

6.5 helga.nuke package

Helga script environment for Nuke. Here you can find scripts, tools and plugins used with Nuke.

6.6 Packages

| | |
|--|--|
| <code>helga.nuke.arash</code> | |
| <code>helga.nuke.hanna</code> | |
| <code>helga.nuke.johannes</code> | |
| <code>helga.nuke.manuel</code> | |
| <code>helga.nuke.photoscan</code> | <code>helga.nuke.photoscan package</code> |
| <code>helga.nuke.reconstruction</code> | <code>helga.nuke.reconstruction package</code> |
| <code>helga.nuke.setup</code> | <code>helga.nuke.setup package</code> |

6.6.1 helga.nuke.photoscan package

All packages and modules related to Photoscan action and Nuke can be found here.

Author: Timm Wagener

6.6.2 Packages

| | |
|--|-------------------------------|
| <code>helga.nuke.photoscan.auto_key</code> | <code>Auto Key Package</code> |
|--|-------------------------------|

Auto Key Package

Automatic live key for photoscan sessions.

Author: Timm Wagener *Version:* 0.1

Submodules

auto_key

Live key in Nuke for photoscan sessions.

Usage Copy this script in the Nuke script editor and execute it.

6.6.3 helga.nuke.reconstruction package

All packages and modules belonging to render or scene reconstruction in Nuke can be found here.

Author: Timm Wagener

6.6.4 Packages

| | |
|--|---------------------------------|
| <code>helga.nuke.reconstruction.renderReconstructVRay</code> | Render Reconstruct VRay Package |
| <code>helga.nuke.reconstruction.sceneReconstructVRay</code> | Scene Reconstruct VRay Package |

Render Reconstruct VRay Package

Automatic render reconstruction of multi channel exrs in Nuke. The Multichannel exrs need to be set up with this `helga.maya.rendering.createUpdateRenderElements` package in VRay for Maya in order to function properly.

Author: Timm Wagener *Version:* 0.1

Submodules

renderReconstruct

Automatic reconstruction of multichannel exrs that have been created according to our pipeline standards. This has happened most probably with a module from `helga.maya.rendering.createUpdateRenderElements`.

The following elements can be reconstructed separately:

- Light
- Data
- Framebuffer
- Multi Mattes
- Shadow

Of course it is also possible to recreate them altogether.

Note: The final result of the reconstruction might differ a little from the original rendering, since Ambient Occlusion is rendered as an Extra Tex pass and multiplied with the GI on reconstruction.

Usage

```
from helga.nuke.reconstruction.renderReconstructVray import renderReconstruct
reload(renderReconstruct)

#Create instance
renderReconstructInstance = renderReconstruct.RenderReconstruct()

#-----
#Select some Nuke Read nodes
#-----

#Reconstruct all elements
renderReconstructInstance.reconstructAll()

#Reconstruct light only
renderReconstructInstance.reconstructLightREs()

#Reconstruct data only
renderReconstructInstance.reconstructDataREs()

#Reconstruct framebuffer only
renderReconstructInstance.reconstructFramebufferREs()

#Reconstruct multi mattes only
renderReconstructInstance.reconstructMultiMatterEs()

#Reconstruct shadow only
renderReconstructInstance.reconstructShadowREs()
```

class helga.nuke.reconstruction.renderReconstructVray.renderReconstruct.**RenderReconstruct**

nodetypeMatches (*node, nodetype*)

reconstructAll ()

Reconstruct all render elements (AOVs) for the selected Read nodes. The elements are added together to reconstruct the final render. The rendering might differ a little from the original since Ambient Occlusion is multiplied with the GI.

reconstructDataREs ()

Reconstruct data render elements (AOVs) for the selected Read nodes.

reconstructFramebufferREs ()

Reconstruct framebuffer render elements (AOVs) for the selected Read nodes.

reconstructLightREs ()

Reconstruct light render elements (AOVs) for the selected Read nodes.

reconstructMultiMatterEs ()

Reconstruct multi matte render elements (AOVs) for the selected Read nodes.

reconstructShadowREs ()

Reconstruct shadow render elements (AOVs) for the selected Read nodes.

lib Package

Contains internal library modules for the renderReconstruct tool.

Author: Timm Wagener Version: 0.1

reconstruct_globals Globals module to hosts constants for the `helga.nuke.reconstruction.renderReconstructVRay` package.

Usage

```
from helga.nuke.reconstruction.renderReconstructVRay import reconstruct_globals
reload(reconstruct_globals)
```

```
#my_fancy_prefix
my_fancy_prefix = reconstruct_globals.PREFIX
```

`helga.nuke.reconstruction.renderReconstructVRay.lib.reconstruct_globals.PREFIX = 'gh'`
Prefix for the channels in multichannel exrs. For example: `ghDiffuse`, `ghReflection`

shuffleDataRE Internal module that reconstructs data render elements.

| | | | |
|-------------------|--------------------------|-------------------------------|--|
| Important: | This module is internal. | Access its functionality from | |
| | | | <code>helga.nuke.reconstruction.renderReconstructVRay.renderReconstruct</code> |

class `helga.nuke.reconstruction.renderReconstructVRay.lib.shuffleDataRE.ShuffleDataRE`

```
createBackdrop (nodesList, hexColor)
deselectAll ()
getLayers (node, prefix=False)
nodetypeMatches (node, nodetype)
reconstructDataRenderElements (node)
rgbToHexString (colorList=[0, 0, 0])
shuffleDepthRE (node, layer)
shuffleRE (node, layer)
```

shuffleFramebufferRE Internal module that reconstructs framebuffer render elements.

| | | | |
|-------------------|--------------------------|-------------------------------|--|
| Important: | This module is internal. | Access its functionality from | |
| | | | <code>helga.nuke.reconstruction.renderReconstructVRay.renderReconstruct</code> |

class `helga.nuke.reconstruction.renderReconstructVRay.lib.shuffleFramebufferRE.ShuffleFramebu`

```
createBackdrop (nodesList, hexColor)
deselectAll ()
```

```
getLayers (node, prefix=False)  
nodetypeMatches (node, nodetype)  
reconstructFramebufferRenderElements (node)  
rgbToHexString (colorList=[0, 0, 0])  
shuffleRE (node, layer, frameBuffer=False)
```

shuffleLightRE Internal module that reconstructs light render elements.

Important: This module is internal. Access its functionality from `helga.nuke.reconstruction.renderReconstructVRay.renderReconstruct`

class `helga.nuke.reconstruction.renderReconstructVRay.lib.shuffleLightRE.ShuffleLightRE`

```
createBackdrop (nodesList, hexColor)  
deselectAll ()  
getLayers (node, prefix=False)  
nodetypeMatches (node, nodetype)  
reconstructLightRE (node, layer, gi=False)  
reconstructLightRenderElements (node)  
rgbToHexString (colorList=[0, 0, 0])  
shuffleRE (node, layer, frameBuffer=False)
```

shuffleMmRE Internal module that reconstructs multi matte render elements.

Important: This module is internal. Access its functionality from `helga.nuke.reconstruction.renderReconstructVRay.renderReconstruct`

class `helga.nuke.reconstruction.renderReconstructVRay.lib.shuffleMmRE.ShuffleMmRE`

```
createBackdrop (nodesList, hexColor)  
deselectAll ()  
getLayers (node, prefix=False)  
nodetypeMatches (node, nodetype)  
reconstructMultiMatteRenderElements (node)  
rgbToHexString (colorList=[0, 0, 0])  
shuffleRE (node, layer)
```

shuffleShadowRE Internal module that reconstructs shadow render elements.

Important: This module is internal. Access its functionality from `helga.nuke.reconstruction.renderReconstructVRay.renderReconstruct`

class `helga.nuke.reconstruction.renderReconstructVRay.lib.shuffleShadowRE.ShuffleShadowRE`

```
createBackdrop (nodesList, hexColor)
deselectAll ()
getLayers (node, prefix=False)
nodetypeMatches (node, nodetype)
reconstructShadowRenderElements (node)
rgbToHexString (colorList=[0, 0, 0])
shuffleRE (node, layer)
```

Scene Reconstruct VRay Package

Automatic scene reconstruction of from exr metadata in Nuke. The exr metadata needs to be set up with this `helga.maya.rendering.setExrMetadata` package in VRay for Maya in order to function properly.

Author: Timm Wagener *Version:* 0.1

Submodules

sceneReconstruct

Automatic reconstruction of 3d scenes in Nuke from exr metadata. It works together with `helga.maya.rendering.setExrMetadata` and can recreate:

- Camera animation
- Alembics with textures
- Light setups

Note: The light reconstruction will try to find the best match for lights it doesnt know. For example a Vray Sunlight might translate to a directional light in Nuke. The mapping is far from complete and far from perfect though.

Please remember that this module only works, when exr metadata is set.

Usage

```
from helga.nuke.reconstruction.sceneReconstructVRay import sceneReconstruct
reload(sceneReconstruct)
```

```
#Create instance
sceneReconstructInstance = sceneReconstruct.SceneReconstruct()
```

```
#-----
#Select some Nuke Read nodes
#-----
```

```
#Reconstruct alembics
sceneReconstructInstance.reconstruct_alembic()
```

```
#Reconstruct exr cam
sceneReconstructInstance.create_exr_cam_vray()
```

```
#Reconstruct light
sceneReconstructInstance.reconstruct_light()
```

```
class helga.nuke.reconstruction.sceneReconstructVRay.sceneReconstruct.SceneReconstruct
```

```
    create_exr_cam_vray()
        Reconstruct a cam from a vray exr. ONLY works with vray exrs

    nodetypeMatches (node, nodetype)

    reconstruct_alembic()
        Reconstruct Alembic nodes from read nodes with exr input file

    reconstruct_light()
        Reconstruct Light nodes from read nodes with exr input file
```

lib Package

Contains internal library modules for the sceneReconstruct tool.

Author: Timm Wagener *Version:* 0.1

reconstruct_alembic Internal module that reconstructs nuke 3d scenes from metadata in exrs according to our pipeline standards

Important: This module is internal. Access its functionality from `helga.nuke.reconstruction.sceneReconstructVRay.sceneReconstruct`

```
helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_alembic.create_alembic_part
```

Create alembic node with given path

`helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_alembic.create_alembic_read`

Create Geo read nodes for alembic pathes in node and return list of them

`helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_alembic.create_backdrop` (node, hex-Color)

Create backdrop for nodesList with hexColor

`helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_alembic.deselect_all` ()
Deselect All

`helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_alembic.nodetypeMatches` (node, node type)

Check if the nodetype matches

`helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_alembic.reconstruct_alembic`

Reconstruct alembic from exr metada in read node

`helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_alembic.rgb_to_hex_string` (color, 0, 0)

Convert RGB List to hex color

reconstruct_camera_from_vray_exr Internal module that reconstructs nuke 3d camera from vray exrs according to our pipeline standards.

Important: This module is internal. Access its functionality from `helga.nuke.reconstruction.sceneReconstructVRay.sceneReconstruct`

`helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_camera_from_vray_exr.create`
Create a camera node based on VRay metadata. This works specifically on VRay data coming from maya.

reconstruct_globals Globals module to host constants for the `helga.nuke.reconstruction.sceneReconstructVRay` package.

`helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_globals.ALEMBIC_DICTIONARY_KEY`
Key for alembic data

`helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_globals.ALEMBIC_READ_NODE_BACKGROUND_COLOR`
Color of Alembic backdrop nodes

`helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_globals.LIGHT_DICTIONARY_KEY`
Key for light data

`helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_globals.LIGHT_NODE_BACKGROUND_COLOR`
Color of light backdrop nodes

`helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_globals.NUKE_EXR_METADATA_PREFIX`
Default exr metadata prefix. (Set by Nuke)

`helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_globals.PREFIX = 'gh'`
Prefix for channels

`helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_globals.align_nodes(nodes, direction='x')`
Align nodes either horizontally or vertically.

reconstruct_light Internal module that reconstructs nuke light from metadata in exrs according to our pipeline standards.

| | |
|-------------------|---|
| Important: | This module is internal. Access its functionality from <code>helga.nuke.reconstruction.sceneReconstructVRay.sceneReconstruct</code> |
|-------------------|---|

`helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_light.create_backdrop(nodesList, hexColor)`
Create backdrop for nodesList with hexColor

`helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_light.deselect_all()`
Deselect All

`helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_light.nodetypeMatches(node, nodetype)`
Check if the nodetype matches

`helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_light.reconstruct_VRayLightType`

`helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_light.reconstruct_VRaySunShade`

`helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_light.reconstruct_light(node, verbose)`
Reconstruct light from exr metada in read node

`helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_light.reconstruct_light_by_type`
Reconstruct light by type

`helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_light.rgb_to_hex_string(color, 0, 0)`
Convert RGB List to hex color

6.6.5 helga.nuke.setup package

This package is not neccessarily a traditional package. In here is stuff like the central pipeline third party *scripts* and *plugins* folders. Also in here are scripts and icons used to setup our pipeline nuke.

Author: Timm Wagener

6.7 helga.general package

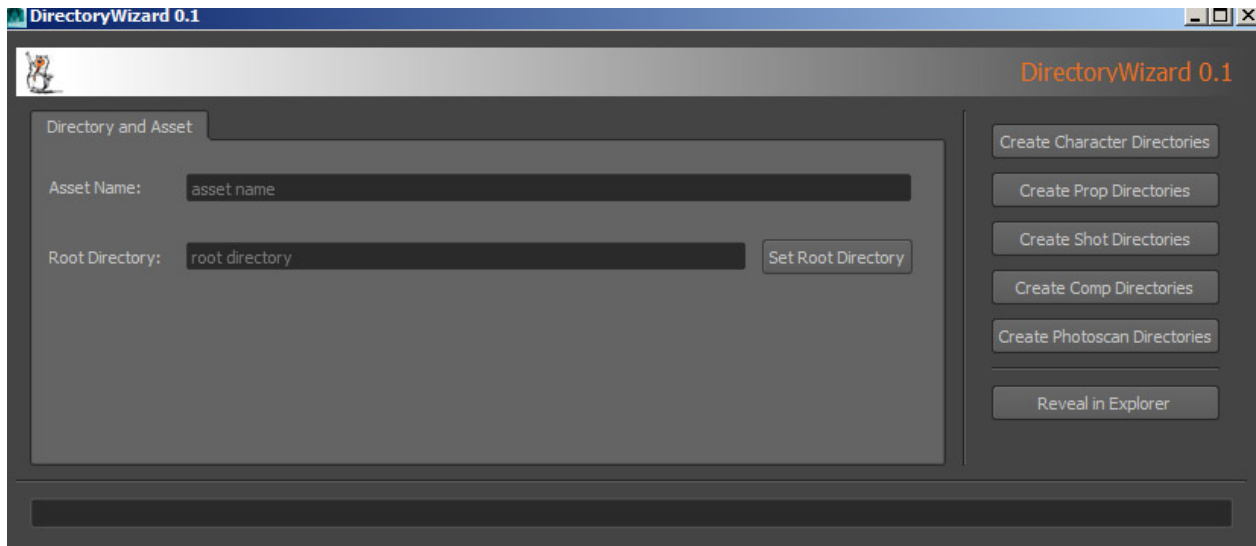
Helga general script environment. Here you can find scripts, tools and plugins that are not specific for any DCC.

6.8 Packages

| | |
|---|--|
| <code>helga.general.directory_wizard</code> | Directory Wizard Package |
| <code>helga.general.setup</code> | <code>helga.general.setup</code> package |

6.8.1 Directory Wizard Package

Simple GUI for creating directories.



Author: Timm Wagener *Version:* 0.1

6.8.2 Submodules

`directory_wizard`

Simple but hopefully helpful module to automatically generate folderstructures for assets, comps etc. It uses PySide and is ment to run from Mayas, Nukes and Houdinis Python interpreters.

Usage

```
from helga.general.directory_wizard import directory_wizard
reload(directory_wizard)

#run
directory_wizard.run()
```

```
helga.general.directory_wizard.directory_wizard.run()
```

Standardized run() method

lib Package

Contains internal library modules for the directory_wizard tool.

Author: Timm Wagener Version: 0.1

directory_creator

Module to create directories.

```
helga.general.directory_wizard.lib.directory_creator.CHARACTER_DIRECTORY_LIST = ['model', ['export']]
    Helga character directory structure

helga.general.directory_wizard.lib.directory_creator.COMP_DIRECTORY_LIST = ['2d_render', '3d_render']
    Helga comp directory structure

class helga.general.directory_wizard.lib.directory_creator.CharacterDirectoryCreator (asset_name,
                                                                                       root_directory,
                                                                                       log-
                                                                                       ging_level=10)
    Bases: helga.general.directory_wizard.lib.directory_creator.DirectoryCreator

class helga.general.directory_wizard.lib.directory_creator.CompDirectoryCreator (asset_name,
                                                                                   root_directory,
                                                                                   log-
                                                                                   ging_level=10)
    Bases: helga.general.directory_wizard.lib.directory_creator.DirectoryCreator

class helga.general.directory_wizard.lib.directory_creator.DirectoryCreator (asset_name,
                                                                              root_directory,
                                                                              log-
                                                                              ging_level=10)
    Bases: object

    create_directories (name_list=None, parent_directory=None)
        Recurse self.directory_list and create directories.

    get_asset_directory ()
        Return self.asset_directory

    get_directories_string (name_list=None, tabs='')
        Return string representation of directory_list

    get_directory_type ()
        Return self.directory_type

helga.general.directory_wizard.lib.directory_creator.PHOTOSCAN_DIRECTORY_LIST = ['photos', 'mask']
    Helga photoscan directory structure

helga.general.directory_wizard.lib.directory_creator.PROP_DIRECTORY_LIST = ['model', ['export'], 'photo']
    Helga prop directory structure

class helga.general.directory_wizard.lib.directory_creator.PhotoscanDirectoryCreator (asset_name,
                                                                                       root_directory,
                                                                                       log-
                                                                                       ging_level=10)
    Bases: helga.general.directory_wizard.lib.directory_creator.DirectoryCreator
```



```
class helga.general.directory_wizard.lib.directory_creator.PropDirectoryCreator(asset_name,
                                                                              root_directory,
                                                                              log-
                                                                              ging_level=10)

    Bases: helga.general.directory_wizard.lib.directory_creator.DirectoryCreator

helga.general.directory_wizard.lib.directory_creator.SHOT_DIRECTORY_LIST = ['animation', 'lighting',
Helga shot directory structure

class helga.general.directory_wizard.lib.directory_creator.ShotDirectoryCreator(asset_name,
                                                                              root_directory,
                                                                              log-
                                                                              ging_level=10)

    Bases: helga.general.directory_wizard.lib.directory_creator.DirectoryCreator
```

directory_wizard_logging_handler

Automaticaly log to le_status from given widget.

```
class helga.general.directory_wizard.lib.directory_wizard_logging_handler.StatusStreamHandler

    Bases: logging.StreamHandler

    Stream handler subclass that calls the set_status() method of the wdgt_status object used for initialization.
    Normally the set_status() method would set a QLineEdit or similiar to display the log message.

    emit(record)
        Custom emit for StreamHandler subclass
```

6.8.3 helga.general.setup package

All packages and modules doing general setup stuff or providing setup ressources can be found here.

Author: [Timm Wagener](#)

6.8.4 Packages

| | |
|---|---|
| <code>helga.general.setup.doc_link</code> | Open the Pipeline Documentation Package |
| <code>helga.general.setup.global_functions</code> | Helga global_functions Package |
| <code>helga.general.setup.global_variables</code> | Helga global_variables Package |
| <code>helga.general.setup.quick_setup</code> | Quick Setup Package |

Open the Pipeline Documentation Package

Open a browser to display our pipeline documentation. Simple but (hopefully) helpfull.

Author: [Timm Wagener](#) *Version:* 0.1

Submodules

doc_link

Simple but hopefully helpful module to open up a browser to our pipeline documentation.

Usage

```
from helga.general.setup.doc_link import doc_link
reload(doc_link)

#run
doc_link.run()
```

```
helga.general.setup.doc_link.doc_link.run(url=None)
    Open web browser with pipeline documentation
```

Helga global_functions Package

This package contains all modules that hold all relevant general functions to be used in the Helga pipeline.

Author: Timm Wagener Version: 0.1

Submodules

global_functions

Simple but hopefully helpful module to that bundles all functions in a central place.

Usage

```
from helga.general.setup.global_functions import global_functions
reload(global_functions)

#some example here
global_functions.example_function()
```

```
helga.general.setup.global_functions.global_functions.check_interpreter(keyword)
    Check if the current interpreter matches keyword
```

```
helga.general.setup.global_functions.global_functions.copy_file(source_file,
                                                                source_dir,
                                                                destina-
                                                                tion_dir)
```

`helga.general.setup.global_functions.global_functions.create_widget_from_ui_file(ui_file_path, par-ent=None)`

Create widget that remains stylesheets from .ui file. This method is crashy at least in Maya. Probably because of stylesheet issues with Maya QApp custom style. Use with caution.

`helga.general.setup.global_functions.global_functions.divider()`
Print divider line to `__stdout__`

`helga.general.setup.global_functions.global_functions.get_helga_header_widget(title, icon_path)`
Return QWidget that represents a helga header. The returned widget needs to be added to a layout.

`helga.general.setup.global_functions.global_functions.get_main_window()`
Determine current application and return main window for parenting of own windows.

`helga.general.setup.global_functions.global_functions.get_maya_main_window()`
Return the Maya main window.

`helga.general.setup.global_functions.global_functions.get_maya_toplevel_shelf_widget(shelf_obj)`
Convert ptr to shelf object to Qt instance.

`helga.general.setup.global_functions.global_functions.get_nuke_main_window()`
Return the Maya main window.

`helga.general.setup.global_functions.global_functions.get_user()`

`helga.general.setup.global_functions.global_functions.get_user_setup_destination_dir(dcc)`

`helga.general.setup.global_functions.global_functions.get_user_setup_destination_dir houdini`

`helga.general.setup.global_functions.global_functions.get_user_setup_destination_dir maya`

`helga.general.setup.global_functions.global_functions.get_user_setup_destination_dir nuke`

`helga.general.setup.global_functions.global_functions.load_ui_type(ui_file)`
Pyside lacks the “loadUiType” command, so we have to convert the ui file to py code in-memory first and then execute it in a special frame to retrieve the form_class. This function return the form and base classes for the given qtdesigner ui file.

`helga.general.setup.global_functions.global_functions.style_maya_shelves()`
Style Maya shelves.

Helga global_variables Package

This package contains all modules that hold all relevant constants to be used in the Helga pipeline.

Author: [Timm Wagener](#) Version: 0.1

Submodules

`global_variables`

Simple but hopefully helpful module to that bundles all constants and pipeline variables in a central place.

Usage

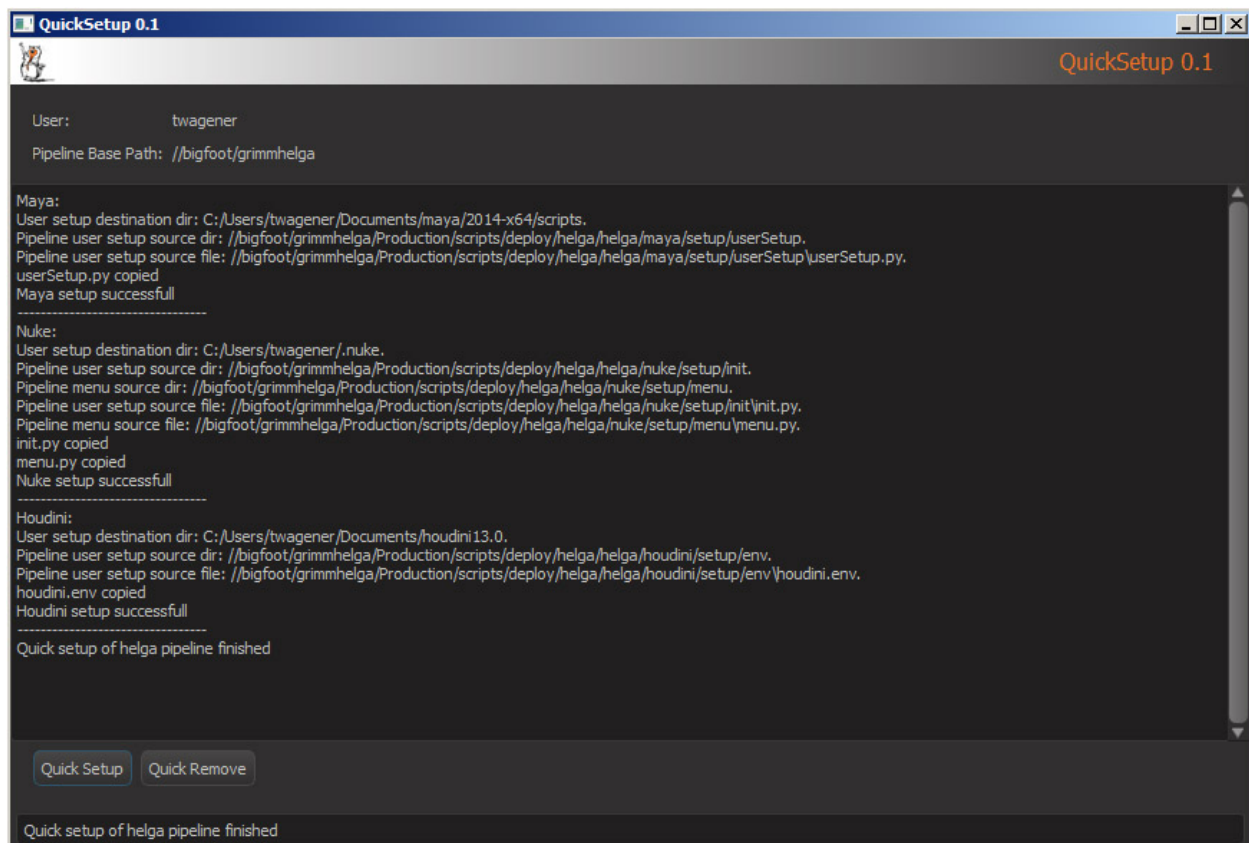
```
from helga.general.setup.global_variables import global_variables
reload(global_variables)

# pipeline_base_path
pipeline_base_path = global_variables.PIPELINE_BASE_PATH
```

```
helga.general.setup.global_variables.global_variables.PIPELINE_BASE_PATH = False
Helga pipeline base path
```

Quick Setup Package

Simple GUI for putting/removing the helga pipeline applications (Houdini, Maya, Nuke) into/from deployment. The tool is not ment to run within a DCC but is compiled and bundled with py2exe. It is published under */bin/QuickSetup*.



Dependencies

1. PyQt4
2. py2exe
3. qdarkstyle

Author: Timm Wagener *Version:* 0.1

Submodules

quick_setup_loader

Helper module to enable import of quick_setup. This module makes sure the `__file__` attribute of the quick_setup module delivers a path.

```
helga.general.setup.quick_setup.quick_setup_loader.load_quick_setup()  
    Start an QApp loop and load an instance of QuickSetup in it.
```

quick_setup

Simple but hopefully helpful module to automatically put the helga pipeline DCCs into deployment or remove them from it.

The tool is available as precompiled binary under `$PIPELINE_SCRIPTS_BASE_PATH/bin`

Author: Timm Wagener

```
helga.general.setup.quick_setup.quick_setup.clean_path_for_favorites(path)  
    Modify path to match favorite neccessities.
```

lib Package

quick_setup_functionality Module that holds the functionality for quick_setup.

```
helga.general.setup.quick_setup.lib.quick_setup_functionality.add_favorite(shortcut_name,  
                                                                              tar-  
                                                                              get_path,  
                                                                              de-  
                                                                              scrip-  
                                                                              tion='')
```

Add favorite shortcut. If shortcut already exists in Links, the shortcut will be updated with the given path.

```
helga.general.setup.quick_setup.lib.quick_setup_functionality.copy_file(source_file,  
                                                                           source_dir,  
                                                                           des-  
                                                                           ti-  
                                                                           na-  
                                                                           tion_dir)
```

Copy file.

```
helga.general.setup.quick_setup.lib.quick_setup_functionality.get_favorite_directory()  
    Return favorite directory
```

```
helga.general.setup.quick_setup.lib.quick_setup_functionality.get_links_directory()  
    Return user Links directory
```

`helga.general.setup.quick_setup.lib.quick_setup_functionality.get_nuke_menu_source_dir()`

`helga.general.setup.quick_setup.lib.quick_setup_functionality.get_pipeline_base_path()`

Get pipeline base path. Hardcoded or from environment variable. The environment variable has preference.

`helga.general.setup.quick_setup.lib.quick_setup_functionality.get_pipeline_scripts_base_path()`

Get pipeline scripts base path.

`helga.general.setup.quick_setup.lib.quick_setup_functionality.get_time_string()`

Return time string.

`helga.general.setup.quick_setup.lib.quick_setup_functionality.get_user()`

Get the currently logged on user or Unknown if False.

`helga.general.setup.quick_setup.lib.quick_setup_functionality.get_user_setup_destination_dir()`

`helga.general.setup.quick_setup.lib.quick_setup_functionality.get_user_setup_destination_dir()`

`helga.general.setup.quick_setup.lib.quick_setup_functionality.get_user_setup_destination_dir()`

`helga.general.setup.quick_setup.lib.quick_setup_functionality.get_user_setup_destination_dir()`

`helga.general.setup.quick_setup.lib.quick_setup_functionality.get_user_setup_source_dir(destination_dir)`

`helga.general.setup.quick_setup.lib.quick_setup_functionality.get_user_setup_source_dir(destination_dir)`

`helga.general.setup.quick_setup.lib.quick_setup_functionality.get_user_setup_source_dir(destination_dir)`

`helga.general.setup.quick_setup.lib.quick_setup_functionality.get_user_setup_source_dir(destination_dir)`

quick_setup_logging_handler Automatically log to `le_status` from given widget.

class `helga.general.setup.quick_setup.lib.quick_setup_logging_handler.StatusStreamHandler(widget, log_level)`

Bases: `logging.StreamHandler`

Stream handler subclass that calls the `set_status()` method of the `wdgt_status` object used for initialization.

Normally the `set_status()` method would set a `QLineEdit` or similiar to display the log message.

emit (*record*)

Custom emit for `StreamHandler` subclass

Indices and tables

- *genindex*
- *modindex*

h

```

helga.maya.arash.clean_save,33
helga.maya.arash.clean_save.clean_save,
34
helga.maya.arash.helga_login,35
helga.maya.arash.helga_login.helga_login,
35
helga.maya.hanna,41
helga.maya.johannes,41
helga.maya.manuel,41
helga.maya.rendering_logging_handler,
41
helga.maya.rendering,42
helga.maya.rendering.createUpdateRenderElements,
42
helga.maya.rendering.createUpdateRenderElements.cre
42
helga.maya.rendering.createUpdateRenderElements.cre
43
helga.maya.rendering.createUpdateRenderElements.cre
44
helga.maya.rendering.createUpdateRenderElements.cre
45
helga.maya.rendering.createUpdateRenderElements.cre
46
helga.maya.rendering.renderPathes,47
helga.maya.rendering.renderPathes.renderPathes,
47
helga.maya.rendering.setExrMetadata,48
helga.maya.rendering.setExrMetadata.setExrMetadata,
48
helga.maya.rendering.setVrayRenderGlobals,
48
helga.maya.rendering.setVrayRenderGlobals.setVrayRe
49
helga.maya.rigging,49
helga.maya.setup,49
helga.nuke,50
helga.nuke.photoscan,50
helga.nuke.photoscan.auto_key,50
helga.nuke.photoscan.auto_key.auto_key,
50

```

helga.nuke.reconstruction, [51](#)
helga.nuke.reconstruction.renderReconstructVRay,
 [51](#)
helga.nuke.reconstruction.renderReconstructVRay.lib,
 [52](#)
helga.nuke.reconstruction.renderReconstructVRay.lib.reconstruct_globals,
 [53](#)
helga.nuke.reconstruction.renderReconstructVRay.lib.shuffleDataRE,
 [53](#)
helga.nuke.reconstruction.renderReconstructVRay.lib.shuffleFramebufferRE,
 [53](#)
helga.nuke.reconstruction.renderReconstructVRay.lib.shuffleLightRE,
 [54](#)
helga.nuke.reconstruction.renderReconstructVRay.lib.shuffleMmRE,
 [54](#)
helga.nuke.reconstruction.renderReconstructVRay.lib.shuffleShadowRE,
 [54](#)
helga.nuke.reconstruction.renderReconstructVRay.renderReconstruct,
 [51](#)
helga.nuke.reconstruction.sceneReconstructVRay,
 [55](#)
helga.nuke.reconstruction.sceneReconstructVRay.lib,
 [56](#)
helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_alembic,
 [56](#)
helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_camera_from_vray_exr,
 [57](#)
helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_globals,
 [57](#)
helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_light,
 [58](#)
helga.nuke.reconstruction.sceneReconstructVRay.sceneReconstruct,
 [55](#)
helga.nuke.setup, [58](#)

A

add_favorite() (in module helga.maya.arash.helga_login.helga_login.Save_character_pose_method), 38
 helga.general.setup.quick_setup.lib.quick_setup_functionality), 38
 65 change_pose_time_value_status()
 ALEMBIC_DICTIONARY_KEY (in module helga.maya.arash.helga_login.helga_login.Save_character_pose_method), 38
 helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_globals), 38
 57 change_saved_image() (helga.maya.arash.helga_login.helga_login.Load_ch
 ALEMBIC_READ_NODE_BACKDROP_COLOR (in module helga.maya.arash.helga_login.helga_login.Load_ch
 module helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_globals), 37
 57 method), 34
 align_nodes() (in module changeColors() (helga.maya.arash.clean_save.clean_save.cleanSave_UI
 helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_globals), 34
 58 changeDepartmentColor()
 artistField() (helga.maya.arash.clean_save.clean_save.cleanSave_UI (helga.maya.arash.clean_save.clean_save.cleanSave_UI
 method), 34 method), 34

B

BakeAnimation (class in changeFactorColor() (helga.maya.arash.clean_save.clean_save.cleanSave_U
 helga.maya.animation.bakeAnimation.bakeAnimation), 34
 32 method), 34
 bakeAnimation() (helga.maya.animation.bakeAnimation.bakeAnimation.BakeAnimation
 method), 33 character_body_button() (helga.maya.arash.helga_login.helga_login.Helga
 bakeAnimationTransformOnly() method), 36
 (helga.maya.animation.bakeAnimation.bakeAnimation.BakeAnimation
 method), 33 character_choice() (helga.maya.arash.helga_login.helga_login.Helga_cms_a
 method), 36
 Base (class in helga.maya.arash.helga_login.helga_login), CHARACTER_DIRECTORY_LIST (in module
 35 helga.general.directory_wizard.lib.directory_creator),
 blue_a (helga.maya.arash.helga_login.helga_login.Color 60
 attribute), 36 character_hand_button() (helga.maya.arash.helga_login.helga_login.Helga
 method), 36

C

call_hand_load_pose() (helga.maya.arash.helga_login.helga_login.Ulfbert_hand_UI
 method), 40 (helga.maya.arash.helga_login.helga_login.Load_character_pose
 method), 37
 call_hand_save_pose() (helga.maya.arash.helga_login.helga_login.Ulfbert_hand_UI
 method), 40 (helga.maya.arash.helga_login.helga_login.Load_character_pose
 method), 37
 call_load_pose() (helga.maya.arash.helga_login.helga_login.Ulfbert_body_UI
 method), 39 character_save_pose_main_UI()
 call_save_pose() (helga.maya.arash.helga_login.helga_login.Ulfbert_body_UI
 method), 39 (helga.maya.arash.helga_login.helga_login.Save_character_pose
 method), 38
 CharacterDirectoryCreator (class in
 helga.general.directory_wizard.lib.directory_creator),

60 create_check_module_head()
check_interpreter() (in module (helga.maya.arash.helga_login.helga_login.Helga_cms_td_UI
helga.general.setup.global_functions.global_functions), method), 36
62 create_check_module_leg()
check_perspective() (helga.maya.arash.helga_login.helga_login.Save (helga.maya.arash.helga_login.helga_login.Helga_cms_td_UI
method), 38 method), 36
check_scene_jnt() (helga.maya.arash.helga_login.helga_login.Helga_cms_td_UI
method), 36 method), 36
check_scene_polygone() (helga.maya.arash.helga_login.helga_login.Helga_cms_td_UI
method), 36 method), 36
clean_path_for_favorites() (in module method), 60
helga.general.setup.quick_setup.quick_setup), create_exr_cam_vray() (helga.nuke.reconstruction.sceneReconstructVRay.s
65 method), 56
cleanSave_UI (class in create_exr_cam_vray() (in module
helga.maya.arash.clean_save.clean_save), helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_
34 57
cleanTextFields() (helga.maya.arash.clean_save.clean_save.cleanSaveUI_from_ui_file() (in module
method), 34 helga.general.setup.global_functions.global_functions),
Color (class in helga.maya.arash.helga_login.helga_login), 62
36 createBackdrop() (helga.nuke.reconstruction.renderReconstructVRay.lib.shu
color_Editor() (helga.maya.arash.helga_login.helga_login.Helga_cms_td_UI), 53
method), 36 createBackdrop() (helga.nuke.reconstruction.renderReconstructVRay.lib.shu
colorConstant (helga.maya.arash.helga_login.helga_login.Color method), 53
attribute), 36 createBackdrop() (helga.nuke.reconstruction.renderReconstructVRay.lib.shu
COMP_DIRECTORY_LIST (in module method), 54
helga.general.directory_wizard.lib.directory_creator), createBackdrop() (helga.nuke.reconstruction.renderReconstructVRay.lib.shu
60 method), 54
CompDirectoryCreator (class in createBackdrop() (helga.nuke.reconstruction.renderReconstructVRay.lib.shu
helga.general.directory_wizard.lib.directory_creator), method), 55
60 createBumpNormalsRE()
copy_file() (in module (helga.maya.rendering.createUpdateRenderElements.createDataP
helga.general.setup.global_functions.global_functions), method), 43
62 CreateDataPasses (class in
copy_file() (in module helga.maya.rendering.createUpdateRenderElements.createDataPa
helga.general.setup.quick_setup.lib.quick_setup_functionality), 43
65 createDataPasses() (helga.maya.rendering.createUpdateRenderElements.cre
create_alembic_parts() (in module method), 43
helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstructVRay.createDiffuseRE() (helga.maya.rendering.createUpdateRenderElements.cre
56 method), 44
create_alembic_read_nodes() (in module createEtAmbOccRE() (helga.maya.rendering.createUpdateRenderElements
helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstructVRay.createEtAmbOccRE(), method), 43
56 createEtFresnelLargeRE()
create_backdrop() (in module (helga.maya.rendering.createUpdateRenderElements.createDataP
helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstructVRay.createEtFresnelLargeRE(), method), 43
57 createEtFresnelSmallRE()
create_backdrop() (in module (helga.maya.rendering.createUpdateRenderElements.createDataP
helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstructVRay.createEtFresnelSmallRE(), method), 43
58 createEtSTMapRE() (helga.maya.rendering.createUpdateRenderElements.c
create_check_module_arm() method), 43
(helga.maya.arash.helga_login.helga_login.Helga_cms_td_UI), createEtWorldPosRE() (helga.maya.rendering.createUpdateRenderElements
method), 36 method), 43
create_check_module_biped() CreateFrameBufferContributions (class in
(helga.maya.arash.helga_login.helga_login.Helga_cms_td_UI), helga.maya.rendering.createUpdateRenderElements.createFrameB
method), 36 44

deletewindow() (helga.maya.arash.clean_save.clean_save.cleanSave_UI
 method), 34
 depField() (helga.maya.arash.clean_save.clean_save.cleanSave_UI
 method), 34
 deselect_all() (in module helga.general.setup.quick_setup.lib.quick_setup_functionality),
 helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_alembic),
 57
 get_links_directory() (in module
 deselect_all() (in module helga.general.setup.quick_setup.lib.quick_setup_functionality),
 helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_alembic),
 58
 get_main_window() (in module
 deselectAll() (helga.nuke.reconstruction.renderReconstructVRay.lib.shuffleDataRE.ShuffleDataREFunctions.global_functions),
 method), 53
 63
 deselectAll() (helga.nuke.reconstruction.renderReconstructVRay.lib.shuffleFramebuffersRE.ShuffleFramebuffersRE
 method), 53
 helga.general.setup.global_functions.global_functions),
 deselectAll() (helga.nuke.reconstruction.renderReconstructVRay.lib.shuffleLightRE.ShuffleLightRE
 method), 54
 get_maya_toplevel_shelf_widget() (in module
 deselectAll() (helga.nuke.reconstruction.renderReconstructVRay.lib.shuffleMenuBarRE.ShuffleMenuBarREFunctions.global_functions),
 method), 54
 63
 deselectAll() (helga.nuke.reconstruction.renderReconstructVRay.lib.shuffleShadowsRE.ShuffleShadowsRE module
 method), 55
 helga.general.setup.global_functions.global_functions),
 DirectoryCreator (class in 63
 helga.general.directory_wizard.lib.directory_creator),
 60
 get_nuke_menu_source_dir() (in module
 display_all_member_lists() 66
 helga.general.setup.quick_setup.lib.quick_setup_functionality),
 (helga.maya.arash.helga_login.helga_login.UlfbertCharacterLib_base_path() (in module
 method), 39
 helga.general.setup.quick_setup.lib.quick_setup_functionality),
 divider() (in module helga.general.setup.global_functions.global_functions),
 63
 get_pipeline_scripts_base_path() (in module
 dontSave() (helga.maya.arash.clean_save.clean_save.cleanSave_UI helga.general.setup.quick_setup.lib.quick_setup_functionality),
 method), 34
 66
 get_time_string() (in module
 helga.general.setup.quick_setup.lib.quick_setup_functionality),
 E
 emit() (helga.general.directory_wizard.lib.directory_wizard_logging_handler.StatusStreamHandler
 method), 61
 get_user() (in module
 emit() (helga.general.setup.quick_setup.lib.quick_setup_logging_handler.StatusStreamHandler
 method), 66
 63
 get_user() (in module
 helga.general.setup.quick_setup.lib.quick_setup_functionality),
 F
 faktorField() (helga.maya.arash.clean_save.clean_save.cleanSave_UI 66
 method), 34
 get_user_setup_destination_dir() (in module
 file_path() (helga.maya.arash.clean_save.clean_save.cleanSave_UI helga.general.setup.global_functions.global_functions),
 method), 34
 63
 get_user_setup_destination_dir() (in module
 fk2ik_ulfbert_character() helga.general.setup.quick_setup.lib.quick_setup_functionality),
 (helga.maya.arash.helga_login.helga_login.Base 66
 method), 35
 get_user_setup_destination_dir_houdini() (in module
 G
 helga.general.setup.global_functions.global_functions),
 63
 get_asset_directory() (helga.general.directory_wizard.lib.directory_creator.DirectoryCreator
 method), 60
 get_user_setup_destination_dir_houdini() (in module
 get_directories_string() (helga.general.directory_wizard.lib.directory_creator.DirectoryCreator
 method), 60
 66
 get_user_setup_destination_dir_maya() (in module
 get_directory_type() (helga.general.directory_wizard.lib.directory_creator.DirectoryCreator
 method), 60
 helga.general.setup.global_functions.global_functions),
 63

[get_user_setup_destination_dir_maya\(\)](#) (in module [helga.general.directory_wizard.lib.directory_creator](#)
[helga.general.setup.quick_setup.lib.quick_setup_functionality](#)(module), 60
66 [helga.general.directory_wizard.lib.directory_wizard_logging_handler](#)
[get_user_setup_destination_dir_nuke\(\)](#) (in module (module), 61
[helga.general.setup.global_functions.global_functions](#)(module), 61
63 [helga.general.setup.doc_link](#) (module), 61
[get_user_setup_destination_dir_nuke\(\)](#) (in module [helga.general.setup.doc_link.doc_link](#) (module), 62
[helga.general.setup.quick_setup.lib.quick_setup_functionality](#)(module), 62
66 [helga.general.setup.global_functions.global_functions](#)
[get_user_setup_source_dir\(\)](#) (in module (module), 62
[helga.general.setup.quick_setup.lib.quick_setup_functionality](#)(module), 63
66 [helga.general.setup.global_variables.global_variables](#)
[get_user_setup_source_dir_houdini\(\)](#) (in module (module), 63
[helga.general.setup.quick_setup.lib.quick_setup_functionality](#)(module), 64
66 [helga.general.setup.quick_setup.lib](#) (module), 65
[get_user_setup_source_dir_maya\(\)](#) (in module [helga.general.setup.quick_setup.lib.quick_setup_functionality](#)
[helga.general.setup.quick_setup.lib.quick_setup_functionality](#)(module), 65
66 [helga.general.setup.quick_setup.lib.quick_setup_logging_handler](#)
[get_user_setup_source_dir_nuke\(\)](#) (in module (module), 66
[helga.general.setup.quick_setup.lib.quick_setup_functionality](#)(module), 65
66 [helga.general.setup.quick_setup.quick_setup](#) (module),
65
[getAllSceneLights\(\)](#) ([helga.maya.rendering.createUpdateRenderElements.setupUpdateLightRenderElements](#)
method), 46 (module), 65
[getLayers\(\)](#) ([helga.nuke.reconstruction.renderReconstructVRay.lib.shuffleDataRE.ShuffleDataRE](#)
method), 53 [helga.houdini.setup](#) (module), 31
[getLayers\(\)](#) ([helga.nuke.reconstruction.renderReconstructVRay.lib.shuffleFrameBufferRE.ShuffleFrameBufferRE](#)
method), 53 [helga.maya.animation](#) (module), 32
[getLayers\(\)](#) ([helga.nuke.reconstruction.renderReconstructVRay.lib.shuffleLightRE.ShuffleLightRE](#)(module), 32
method), 54 [helga.maya.animation.bakeAnimation.bakeAnimation](#)
[getLayers\(\)](#) ([helga.nuke.reconstruction.renderReconstructVRay.lib.shuffleMmRE.ShuffleMmRE](#)
method), 54 [helga.maya.arash](#) (module), 33
[getLayers\(\)](#) ([helga.nuke.reconstruction.renderReconstructVRay.lib.shuffleShadowRE.ShuffleShadowRE](#)
method), 55 [helga.maya.arash.clean_save.clean_save](#) (module), 34
[getRenderElementsByPrefix\(\)](#) [helga.maya.arash.helga_login](#) (module), 35
([helga.maya.rendering.createUpdateRenderElements.createUpdateMmRenderElements](#)(module), 41
method), 46 [helga.maya.hanna](#) (module), 41
[getRenderElementSetsByPrefix\(\)](#) [helga.maya.johannes](#) (module), 41
([helga.maya.rendering.createUpdateRenderElements.createUpdateLightRenderElements](#)
method), 46 [helga.maya.rendering](#) (module), 42
[getSceneObjectIdList\(\)](#) ([helga.maya.rendering.createUpdateRenderElements.createUpdateMmRenderElements](#)
method), 46 (module), 42
[getSelection\(\)](#) ([helga.maya.animation.bakeAnimation.bakeAnimation](#)(module), 42
method), 33 [helga.maya.rendering.createUpdateRenderElements.createDataPasses](#)
([helga.maya.rendering.createUpdateRenderElements.createDataPasses](#)
module), 42
[gray_a](#) ([helga.maya.arash.helga_login.helga_login.Color](#) [helga.maya.rendering.createUpdateRenderElements.createFrameBufferCon](#)
attribute), 36 (module), 43
[green_a](#) ([helga.maya.arash.helga_login.helga_login.Color](#) [helga.maya.rendering.createUpdateRenderElements.createShadowPass](#)
attribute), 36 (module), 44
[helga.maya.rendering.createUpdateRenderElements.createUpdateLightRender](#)
(module), 45
[helga.maya.rendering.createUpdateRenderElements.createUpdateMmRender](#)
(module), 46
[helga.maya.rendering.createUpdateRenderElements.vrayGlobals](#)
(module), 46
[helga.maya.rendering.renderPathes](#) (module), 47

H

[helga.general](#) (module), 58
[helga.general.directory_wizard](#) (module), 59
[helga.general.directory_wizard.directory_wizard](#) (mod-
ule), 59
[helga.general.directory_wizard.lib](#) (module), 60

[helga.maya.rendering.renderPathes.renderPathes \(module\)](#), [47](#)
[helga.maya.rendering.setExrMetadata \(module\)](#), [48](#)
[helga.maya.rendering.setExrMetadata.setExrMetadata \(module\)](#), [48](#)
[helga.maya.rendering.setVrayRenderGlobals \(module\)](#), [48](#)
[helga.maya.rendering.setVrayRenderGlobals.setVrayRenderGlobals \(module\)](#), [49](#)
[helga.maya.rigging \(module\)](#), [49](#)
[helga.maya.setup \(module\)](#), [49](#)
[helga.nuke \(module\)](#), [50](#)
[helga.nuke.photoscan \(module\)](#), [50](#)
[helga.nuke.photoscan.auto_key \(module\)](#), [50](#)
[helga.nuke.photoscan.auto_key.auto_key \(module\)](#), [50](#)
[helga.nuke.reconstruction \(module\)](#), [51](#)
[helga.nuke.reconstruction.renderReconstructVRay \(module\)](#), [51](#)
[helga.nuke.reconstruction.renderReconstructVRay.lib \(module\)](#), [52](#)
[helga.nuke.reconstruction.renderReconstructVRay.lib.reconstruct_globals \(module\)](#), [53](#)
[helga.nuke.reconstruction.renderReconstructVRay.lib.shuffle_data_ref \(module\)](#), [53](#)
[helga.nuke.reconstruction.renderReconstructVRay.lib.shuffle_from_buffer_ref \(module\)](#), [53](#)
[helga.nuke.reconstruction.renderReconstructVRay.lib.shuffle_left_fingers \(module\)](#), [54](#)
[helga.nuke.reconstruction.renderReconstructVRay.lib.shuffle_middle_fingers \(module\)](#), [54](#)
[helga.nuke.reconstruction.renderReconstructVRay.lib.shuffle_right_fingers \(module\)](#), [54](#)
[helga.nuke.reconstruction.renderReconstructVRay.lib.shuffle_left_leg \(module\)](#), [54](#)
[helga.nuke.reconstruction.renderReconstructVRay.lib.shuffle_right_leg \(module\)](#), [54](#)
[helga.nuke.reconstruction.renderReconstructVRay.renderReconstruct \(module\)](#), [51](#)
[helga.nuke.reconstruction.sceneReconstructVRay \(module\)](#), [55](#)
[helga.nuke.reconstruction.sceneReconstructVRay.lib \(module\)](#), [56](#)
[helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_alembic \(module\)](#), [56](#)
[helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_camera_from_vray_exr \(module\)](#), [57](#)
[helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_globals \(module\)](#), [57](#)
[helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_light \(module\)](#), [58](#)
[helga.nuke.reconstruction.sceneReconstructVRay.sceneReconstruct \(module\)](#), [55](#)
[helga.nuke.setup \(module\)](#), [58](#)
[Helga_cms_anim_UI \(class in helga.maya.arash.helga_login.helga_login\)](#), [36](#)
[Helga_cms_login_UI \(class in helga.maya.arash.helga_login.helga_login\)](#), [36](#)
[helga_cms_td_main_UI\(\) \(helga.maya.arash.helga_login.helga_login.Helga_cms_td_UI method\)](#), [37](#)
[Helga_cms_td_UI \(class in helga.maya.arash.helga_login.helga_login\)](#), [36](#)
[Helja_body_main_UI\(\) \(helga.maya.arash.helga_login.helga_login.Helja_body_UI method\)](#), [37](#)
[Helja_body_UI \(class in helga.maya.arash.helga_login.helga_login\)](#), [37](#)
[hide_all_bends\(\) \(helga.maya.arash.helga_login.helga_login.Ulfbert_body method\)](#), [39](#)
[hide_all_controls\(\) \(helga.maya.arash.helga_login.helga_login.Ulfbert_body method\)](#), [39](#)
[hide_left_arm\(\) \(helga.maya.arash.helga_login.helga_login.Ulfbert_body method\)](#), [39](#)
[hide_left_fingers\(\) \(helga.maya.arash.helga_login.helga_login.Ulfbert_body method\)](#), [39](#)
[hide_left_leg\(\) \(helga.maya.arash.helga_login.helga_login.Ulfbert_body method\)](#), [39](#)
[hide_right_side\(\) \(helga.maya.arash.helga_login.helga_login.Ulfbert_body method\)](#), [39](#)
[hide_from_buffer\(\) \(helga.maya.arash.helga_login.helga_login.Ulfbert_body method\)](#), [39](#)
[hide_left_fingers\(\) \(helga.maya.arash.helga_login.helga_login.Ulfbert_body method\)](#), [39](#)
[hide_right_leg\(\) \(helga.maya.arash.helga_login.helga_login.Ulfbert_body method\)](#), [39](#)
[hide_right_side\(\) \(helga.maya.arash.helga_login.helga_login.Ulfbert_body method\)](#), [39](#)
[ik2fk_ulfbert_character\(\) \(helga.maya.arash.helga_login.helga_login.Base method\)](#), [35](#)
[load_alembic \(module\)](#), [56](#)
[job_reload_UI\(\) \(helga.maya.arash.helga_login.helga_login.Helga_cms_td method\)](#), [37](#)
[load_globals \(module\)](#), [57](#)
[load_pose_open_directory\(\) \(helga.maya.arash.helga_login.helga_login.Load_character_pose method\)](#), [37](#)
[LIGHT_DICTIONARY_KEY \(in module helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_globals\)](#), [57](#)
[LIGHT_NODE_BACKDROP_COLOR \(in module helga.nuke.reconstruction.sceneReconstructVRay.lib.reconstruct_globals\)](#), [57](#)

run_helga_cms_td_UI() (helga.maya.arash.helga_login.helga_login.Helga_cms_td_UI method), 36

S

Save_character_pose_UI (class in select_character_L_leg() (helga.maya.arash.helga_login.helga_login.Save_character_pose_UI method), 38

save_pose() (helga.maya.arash.helga_login.helga_login.Save_character_pose_UI method), 38

save_pose_action() (helga.maya.arash.helga_login.helga_login.Save_character_pose_UI method), 38

save_pose_action_b() (helga.maya.arash.helga_login.helga_login.Save_character_pose_UI method), 38

save_pose_clean_list() (helga.maya.arash.helga_login.helga_login.Save_character_pose_UI method), 38

save_pose_export_animation() (helga.maya.arash.helga_login.helga_login.Save_character_pose_UI method), 38

save_pose_export_animation_action() (helga.maya.arash.helga_login.helga_login.Save_character_pose_UI method), 38

save_pose_image_action() (helga.maya.arash.helga_login.helga_login.Save_character_pose_UI method), 38

save_pose_image_sub_action() (helga.maya.arash.helga_login.helga_login.Save_character_pose_UI method), 38

save_pose_radioButton_Current_namespace() (helga.maya.arash.helga_login.helga_login.Save_character_pose_UI method), 38

save_pose_radioButton_custom_namespace() (helga.maya.arash.helga_login.helga_login.Save_character_pose_UI method), 38

save_pose_screen_animation() (helga.maya.arash.helga_login.helga_login.Save_character_pose_UI method), 38

save_pose_screen_save() (helga.maya.arash.helga_login.helga_login.Save_character_pose_UI method), 38

saveLogFile() (helga.maya.arash.clean_save.clean_save.cleanSave_UI method), 34

saveScene() (helga.maya.arash.clean_save.clean_save.cleanSave_UI method), 34

saveSceneOverwrite() (helga.maya.arash.clean_save.clean_save.cleanSave_UI method), 34

saveValue() (helga.maya.arash.helga_login.helga_login.Base method), 35

sceneCheck() (helga.maya.arash.clean_save.clean_save.cleanSave_UI method), 34

SceneReconstruct (class in helga.nuke.reconstruction.sceneReconstructVRay.sceneReconstruct method), 56

select_character_L_arm() (helga.maya.arash.helga_login.helga_login.Save_character_pose_UI method), 49

select_character_L_fingers() (helga.maya.arash.helga_login.helga_login.Save_character_pose_UI method), 38

select_character_L_leg() (helga.maya.arash.helga_login.helga_login.Save_character_pose_UI method), 38

select_character_main_body() (helga.maya.arash.helga_login.helga_login.Save_character_pose_UI method), 38

select_character_R_arm() (helga.maya.arash.helga_login.helga_login.Save_character_pose_UI method), 38

select_character_R_fingers() (helga.maya.arash.helga_login.helga_login.Save_character_pose_UI method), 38

select_character_R_leg() (helga.maya.arash.helga_login.helga_login.Save_character_pose_UI method), 38

select_character_selected_member() (helga.maya.arash.helga_login.helga_login.Save_character_pose_UI method), 38

select_module() (helga.maya.arash.helga_login.helga_login.Helga_cms_td_UI method), 37

select_ulfbert_character_controller() (helga.maya.arash.helga_login.helga_login.Base method), 36

select_ulfbert_hand_controller() (helga.maya.arash.helga_login.helga_login.Base method), 36

sendDebug() (helga.maya.arash.clean_save.clean_save.cleanSave_UI method), 34

set_image_sequence() (helga.maya.arash.helga_login.helga_login.Helga_cms_td_UI method), 36

set_name_save_pose() (helga.maya.arash.helga_login.helga_login.Base method), 36

setAnimationRange() (helga.maya.animation.bakeAnimation.bakeAnimation method), 33

setPathLighting() (helga.maya.rendering.renderPathes.renderPathes.RenderPathes method), 47

setPathTesting() (helga.maya.rendering.renderPathes.renderPathes.RenderPathes method), 48

setValue() (helga.maya.arash.helga_login.helga_login.Base method), 36

SetVrayRenderGlobals (class in helga.maya.rendering.setVrayRenderGlobals.setVrayRenderGlobals method), 49

setVrayRenderSettings() (helga.maya.rendering.setVrayRenderGlobals.setVrayRenderGlobals method), 49

setVrayRenderSettingsLinear() (helga.maya.rendering.setVrayRenderGlobals.setVrayRenderGlobals method), 49

setVrayRenderSettingsParameter() (helga.maya.rendering.setVrayRenderGlobals.setVrayRenderGlobals method), 49

SHOT_DIRECTORY_LIST (in module `helga.general.directory_wizard.lib.directory_creator`), 61

ShotDirectoryCreator (class in `helga.general.directory_wizard.lib.directory_creator`), 61

show_all_bends() (helga.maya.arash.helga_login.helga_login.Ulfbert_body_UI method), 39

show_all_controls() (helga.maya.arash.helga_login.helga_login.Ulfbert_body_UI method), 39

show_helper() (helga.maya.arash.helga_login.helga_login.Helga_cms_td_UI method), 37

ShuffleDataRE (class in `helga.nuke.reconstruction.renderReconstructVRay.lib.shuffleDataRE`), 53

shuffleDepthRE() (helga.nuke.reconstruction.renderReconstructVRay.lib.shuffleDataRE.ShuffleDataRE method), 53

ShuffleFramebufferRE (class in `helga.nuke.reconstruction.renderReconstructVRay.lib.shuffleFramebufferRE`), 53

ShuffleLightRE (class in `helga.nuke.reconstruction.renderReconstructVRay.lib.shuffleLightRE`), 54

ShuffleMmRE (class in `helga.nuke.reconstruction.renderReconstructVRay.lib.shuffleMmRE`), 54

shuffleRE() (helga.nuke.reconstruction.renderReconstructVRay.lib.shuffleDataRE.ShuffleDataRE method), 53

shuffleRE() (helga.nuke.reconstruction.renderReconstructVRay.lib.shuffleFramebufferRE.ShuffleFramebufferRE method), 54

shuffleRE() (helga.nuke.reconstruction.renderReconstructVRay.lib.shuffleLightRE.ShuffleLightRE method), 54

shuffleRE() (helga.nuke.reconstruction.renderReconstructVRay.lib.shuffleMmRE.ShuffleMmRE method), 54

shuffleRE() (helga.nuke.reconstruction.renderReconstructVRay.lib.shuffleShadowRE.ShuffleShadowRE method), 55

ShuffleShadowRE (class in `helga.nuke.reconstruction.renderReconstructVRay.lib.shuffleShadowRE`), 55

skinning() (helga.maya.arash.helga_login.helga_login.Helga_cms_td_UI method), 37

startSubThread() (helga.maya.arash.helga_login.helga_login.Helga_cms_anim_UI method), 36

StatusStreamHandler (class in `helga.general.directory_wizard.lib.directory_wizard_logging_handler`), 61

StatusStreamHandler (class in `helga.general.setup.quick_setup.lib.quick_setup_logging_handler`), 66

style_maya_shelves() (in module `helga.general.setup.global_functions.global_functions`), 63

tool_jelly() (helga.maya.arash.helga_login.helga_login.Helga_cms_td_UI method), 37

ulfbert_about() (helga.maya.arash.helga_login.helga_login.Ulfbert_body_UI method), 39

ulfbert_body_graph_editor() (helga.maya.arash.helga_login.helga_login.Ulfbert_body_UI method), 39

ulfbert_body_main_UI() (helga.maya.arash.helga_login.helga_login.Ulfbert_body_UI method), 39

ulfbert_change_namespace_text_color_green() (helga.maya.arash.helga_login.helga_login.Ulfbert_body_UI method), 39

ulfbert_change_namespace_text_color_red() (helga.maya.arash.helga_login.helga_login.Ulfbert_body_UI method), 39

ulfbert_create_reference() (helga.maya.arash.helga_login.helga_login.Ulfbert_body_UI method), 40

ulfbert_debug() (helga.maya.arash.helga_login.helga_login.Ulfbert_body_UI method), 40

ulfbert_get_rotate_data() (helga.maya.arash.helga_login.helga_login.Ulfbert_body_UI method), 40

ulfbert_get_scale_data() (helga.maya.arash.helga_login.helga_login.Ulfbert_body_UI method), 40

ulfbert_get_trans_data() (helga.maya.arash.helga_login.helga_login.Ulfbert_body_UI method), 40

ulfbert_hand_body_main_UI() (helga.maya.arash.helga_login.helga_login.Ulfbert_hand_UI method), 40

ulfbert_hand_graph_editor() (helga.maya.arash.helga_login.helga_login.Ulfbert_hand_UI method), 40

ulfbert_L2R_rotate() (helga.maya.arash.helga_login.helga_login.Ulfbert_body_UI method), 39

ulfbert_L2R_scale() (helga.maya.arash.helga_login.helga_login.Ulfbert_body_UI method), 39

ulfbert_L2R_translate() (helga.maya.arash.helga_login.helga_login.Ulfbert_body_UI method), 39

ulfbert_L_arm_fkik() (helga.maya.arash.helga_login.helga_login.Ulfbert_body_UI method), 39
 ulfbert_L_Leg_fkik() (helga.maya.arash.helga_login.helga_login.Ulfbert_body_UI method), 39
 ulfbert_mirror_L2R_action() (helga.maya.arash.helga_login.helga_login.Ulfbert_hand_UI method), 41
 ulfbert_mirror_R2L_action() (helga.maya.arash.helga_login.helga_login.Ulfbert_body_UI method), 41
 ulfbert_namespace() (helga.maya.arash.helga_login.helga_login.Ulfbert_body_UI method), 40
 ulfbert_R2L_rotate() (helga.maya.arash.helga_login.helga_login.Ulfbert_body_UI method), 39
 ulfbert_R2L_scale() (helga.maya.arash.helga_login.helga_login.Ulfbert_body_UI method), 39
 ulfbert_R2L_translate() (helga.maya.arash.helga_login.helga_login.Ulfbert_body_UI method), 39
 ulfbert_R_arm_fkik() (helga.maya.arash.helga_login.helga_login.Ulfbert_body_UI method), 39
 ulfbert_R_Leg_fkik() (helga.maya.arash.helga_login.helga_login.Ulfbert_body_UI method), 39
 ulfbert_reference_editor() (helga.maya.arash.helga_login.helga_login.Ulfbert_body_UI method), 40
 ulfbert_reset_body_controller() (helga.maya.arash.helga_login.helga_login.Ulfbert_body_UI method), 40
 ulfbert_reset_pose() (helga.maya.arash.helga_login.helga_login.Ulfbert_body_UI method), 40
 ulfbert_reset_rotate() (helga.maya.arash.helga_login.helga_login.Ulfbert_body_UI method), 40
 ulfbert_reset_scale() (helga.maya.arash.helga_login.helga_login.Ulfbert_body_UI method), 40
 ulfbert_reset_translate() (helga.maya.arash.helga_login.helga_login.Ulfbert_body_UI method), 40
 ulfbert_select_all_L_fingers() (helga.maya.arash.helga_login.helga_login.Ulfbert_body_UI method), 41
 ulfbert_select_all_R_fingers() (helga.maya.arash.helga_login.helga_login.Ulfbert_body_UI method), 41
 ulfbert_select_L_index() (helga.maya.arash.helga_login.helga_login.Ulfbert_body_UI method), 41
 ulfbert_select_L_middle() (helga.maya.arash.helga_login.helga_login.Ulfbert_hand_UI method), 41
 ulfbert_select_L_pink() (helga.maya.arash.helga_login.helga_login.Ulfbert_hand_UI method), 41
 ulfbert_select_L_ring() (helga.maya.arash.helga_login.helga_login.Ulfbert_body_UI method), 41
 ulfbert_select_L_thumb() (helga.maya.arash.helga_login.helga_login.Ulfbert_hand_UI method), 41

vrayLoaded() (helga.maya.rendering.createUpdateRenderElements.createShadowPass.CreateShadowPass
method), 45

vrayLoaded() (helga.maya.rendering.createUpdateRenderElements.createUpdateLightRenderElements.CreateUpdateLightRenderElem
method), 46

vrayLoaded() (helga.maya.rendering.createUpdateRenderElements.createUpdateMmRenderElements.CreateUpdateMmRenderElement
method), 46

vrayLoaded() (helga.maya.rendering.renderPathes.renderPathes.RenderPathes
method), 48

vrayLoaded() (helga.maya.rendering.setVrayRenderGlobals.setVrayRenderGlobals.SetVrayRenderGlobals
method), 49

vrayRenderSettingsNode()
(helga.maya.rendering.renderPathes.renderPathes.RenderPathes
method), 48

vrayRenderSettingsNode()
(helga.maya.rendering.setVrayRenderGlobals.setVrayRenderGlobals.SetVrayRenderGlobals
method), 49

W

wrongChangeArtistColor()
(helga.maya.arash.clean_save.clean_save.cleanSave_UI
method), 35

wrongChangeDepartmentColor()
(helga.maya.arash.clean_save.clean_save.cleanSave_UI
method), 35

wrongChangeFaktorColor()
(helga.maya.arash.clean_save.clean_save.cleanSave_UI
method), 35

wrongChangeNameColor()
(helga.maya.arash.clean_save.clean_save.cleanSave_UI
method), 35

wrongChangeSaveColor()
(helga.maya.arash.clean_save.clean_save.cleanSave_UI
method), 35

wrongChangeVersionColor()
(helga.maya.arash.clean_save.clean_save.cleanSave_UI
method), 35