

# Houdini

# Light, Shade, Render

M01: Setting up your Desktop



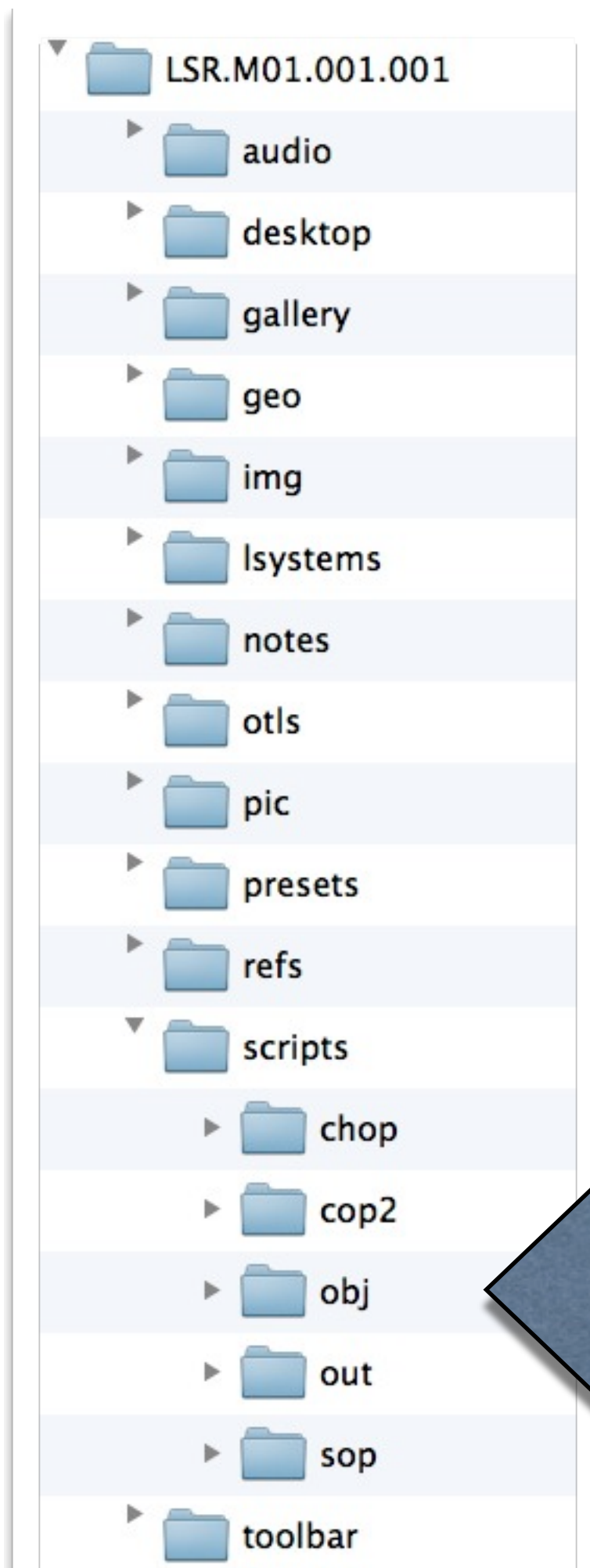
# Agenda

- Understanding the Shot Directory
- Create Custom Desktop for Lighting and Rendering
- Setting Up Houdini for a Linear Work Flow
- Naming Convention for Lights
- Brief Overview of:
  - Render Scheduler
  - Bundle Panel
  - Light Linker
  - Parameter Spread Sheet
- hconfig
- Command Scripts 123.cmd & 456.cmd
- Sourcing in Cameras



# Setting Up the Shot Directory

# Creating a Shot Directory



- **Naming Convention for Shot Directory**
  - `ProjectName.job#.shot#.seq#`

Place Camera Scripts Here

# Paths on Different Operating Systems

- ▶ **For Linux:**

- ▶ \$HOME/houdini12.1

- ▶ **For Mac OSX:**

- ▶ \$HOME/Library/Preferences/houdini/12.1

- ▶ DO NOT USE - \$HOME/houdini12.1

Only for  
legacy  
purposes

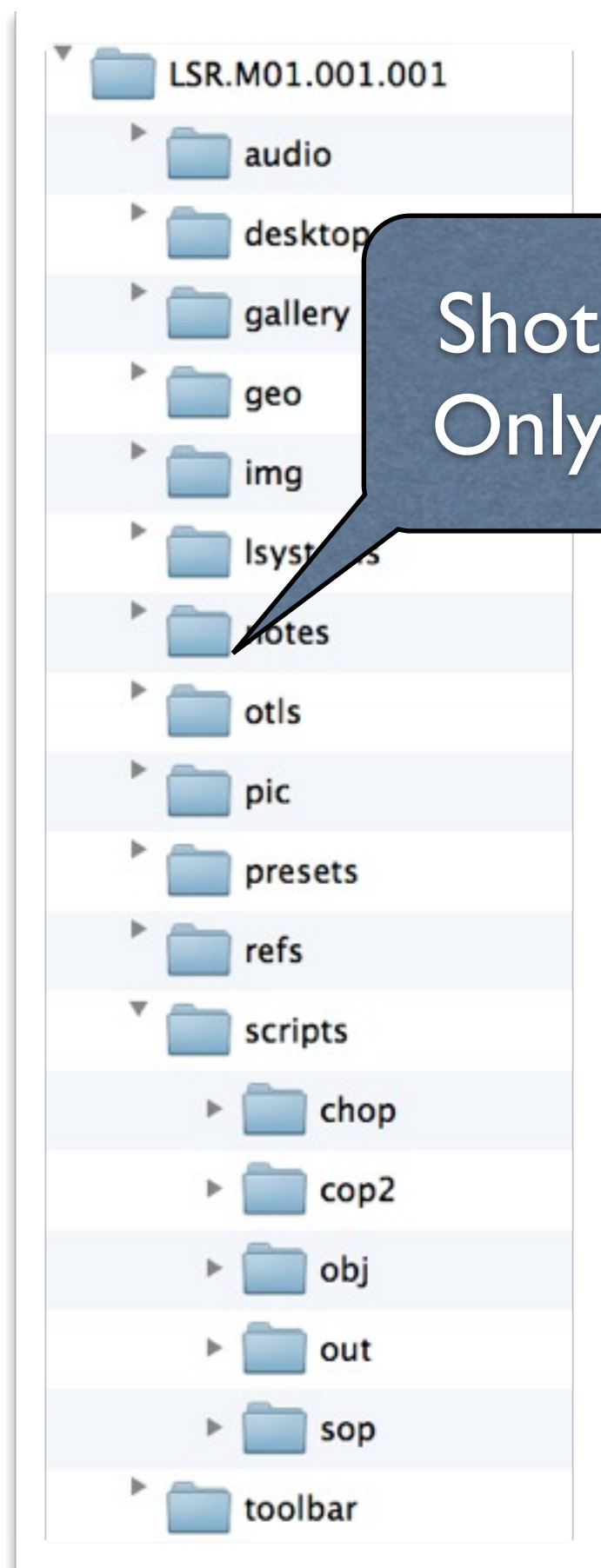
- ▶ If you want to share preferences among multiple users -/Users/Shared/houdini/12.1".

- ▶ **For Windows:**

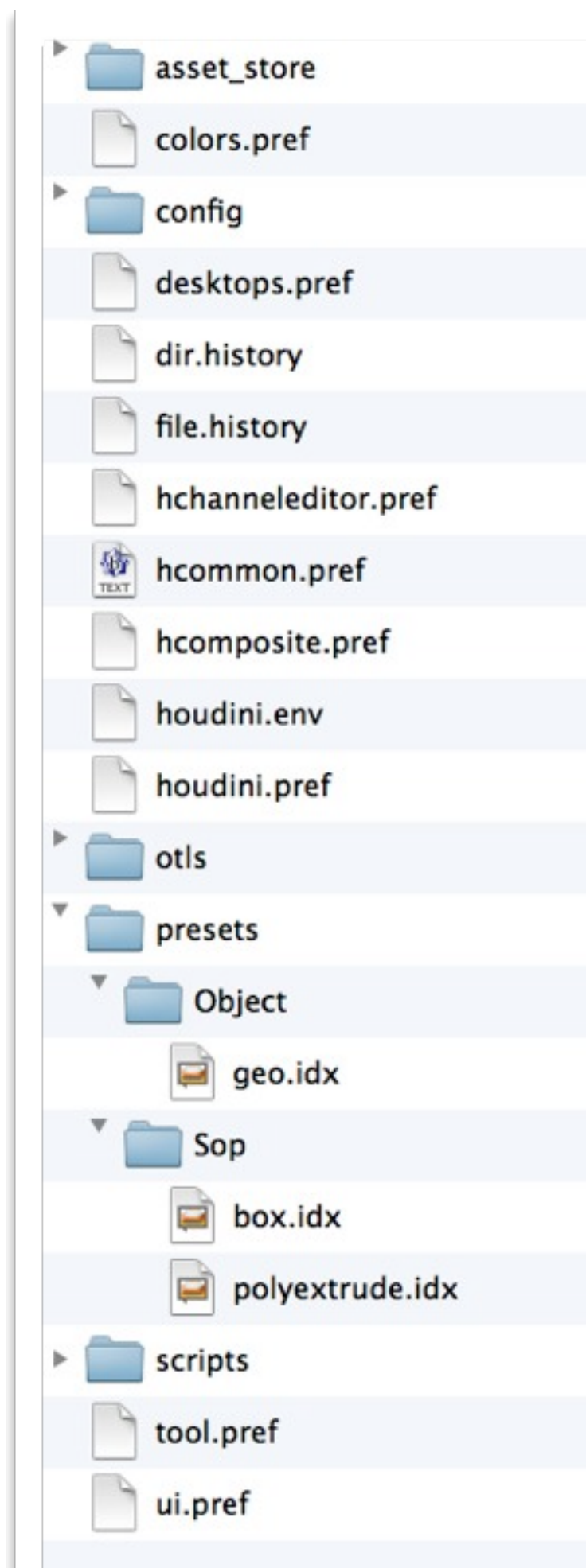
- ▶ %USERPROFILE%\My Documents\houdini12.1

# You can add Scripts and OTLs...

Also in two other folders

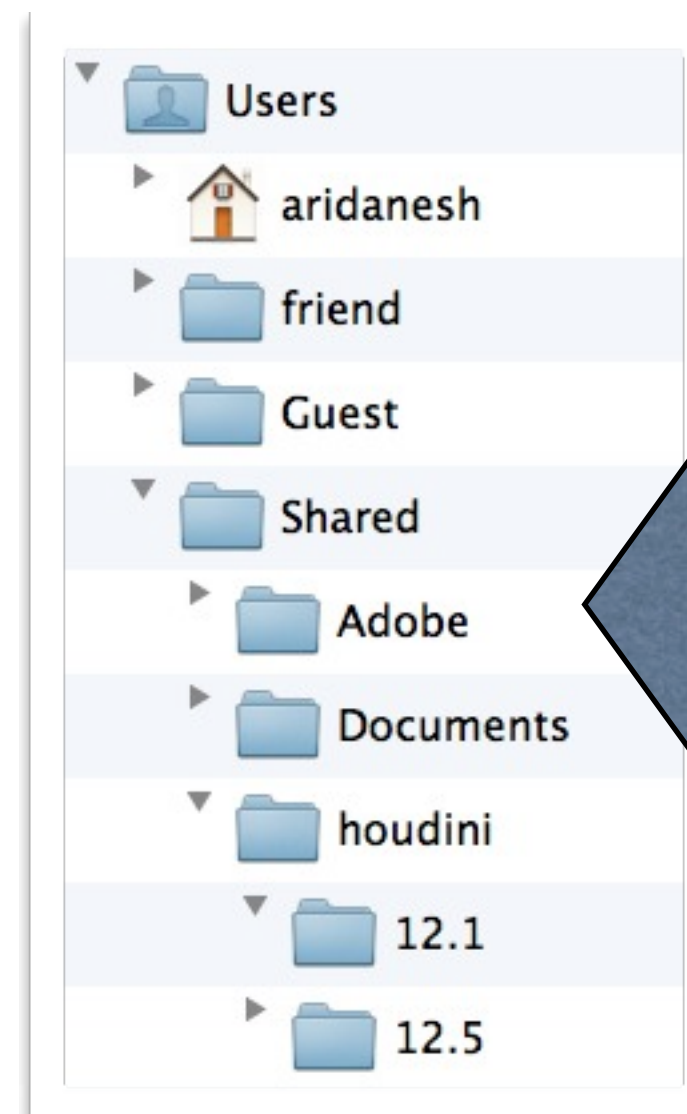


Shot Directory



`$HOME/Library/Preferences/houdini/12.1`

Add Your Own Scripts and OTLs Folder



`/users/shared/houdini/12.1`

Share Scripts Among Multiple Users



# Use Your Terminal (Shell)

- ▶ **Linux and Mac**

- ▶ So Simple - Just Open a Terminal
- ▶ On a Mac it is in your Houdini Folder
  - ▶ Applications --> Houdini 12.1.77 --> Houdini Shell.terminal

- ▶ **Windows**

- ▶ Need to install Software (Two Popular Options)
  - ▶ Cygwin - <http://cygwin.com/index.html>
  - ▶ PowerShell - <http://blogs.technet.com/b/heyscriptingguy/archive/2011/01/07/how-do-i-install-powershell-on-windows-7-and-other-questions.aspx>



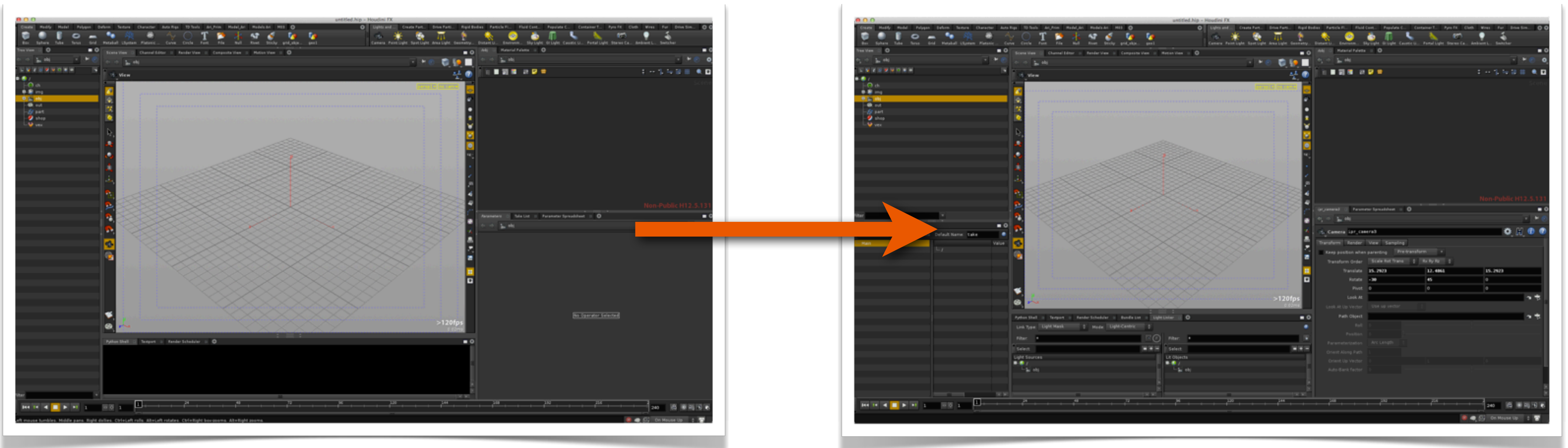
# **Creating a Desktop for Lighting and Rendering**

# Launch Houdini from the Terminal

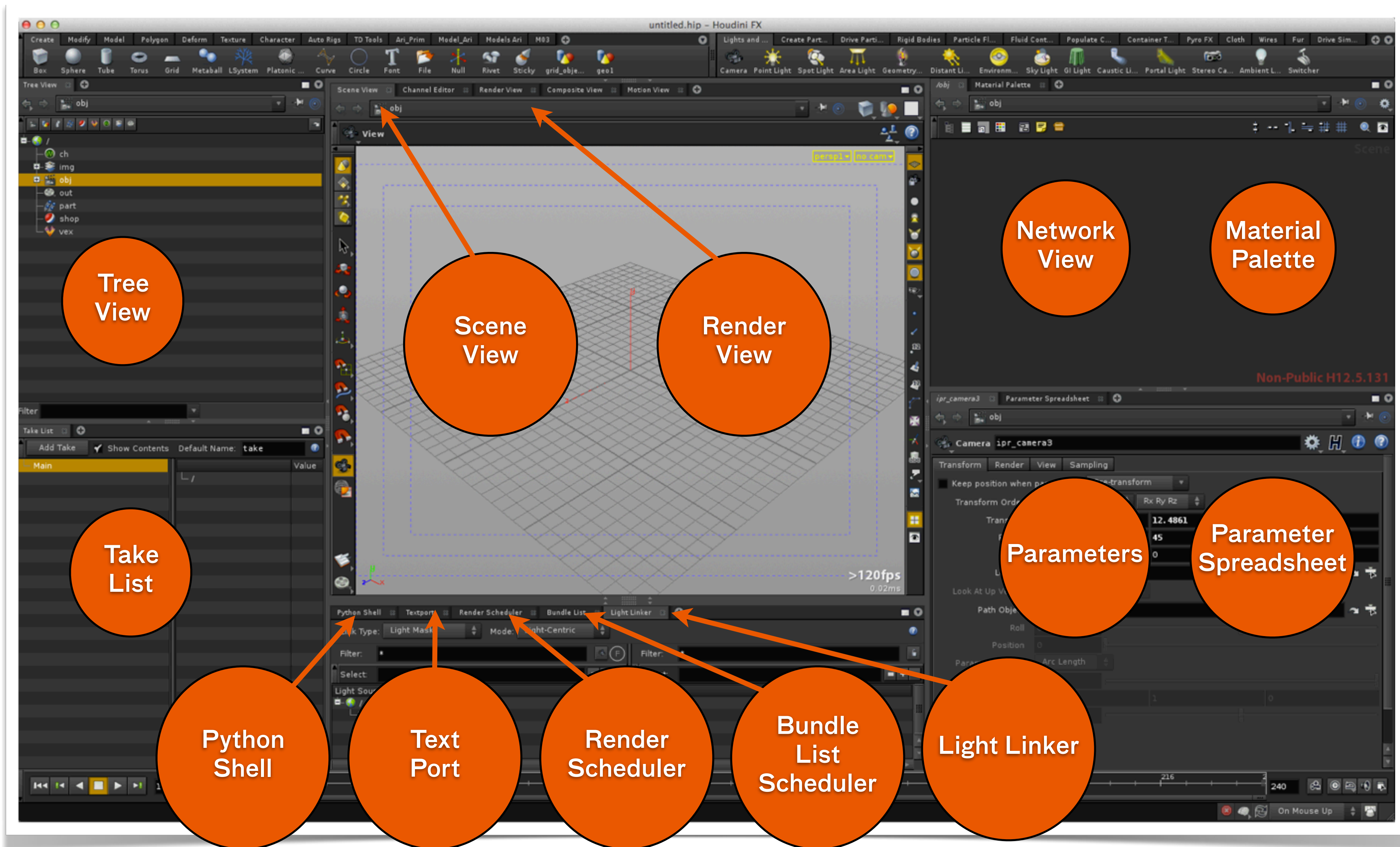
- Let's launch directly to the Technical Desktop
  - houdini -s Technical
    - -s - forces houdini to launch with a specific desktop
    - type houdini - h for all command line options
    - -f: force the use of asset definitions in OTL files on the command line
    - **-j: set maximum processors to nproc**
    - **-s: specify starting desktop by name**
    - **-n: start up in Never cook mode**
    - -foreground: starts process in foreground
    - -geometry: specify window width, height and x and y offsets

These three are  
very useful

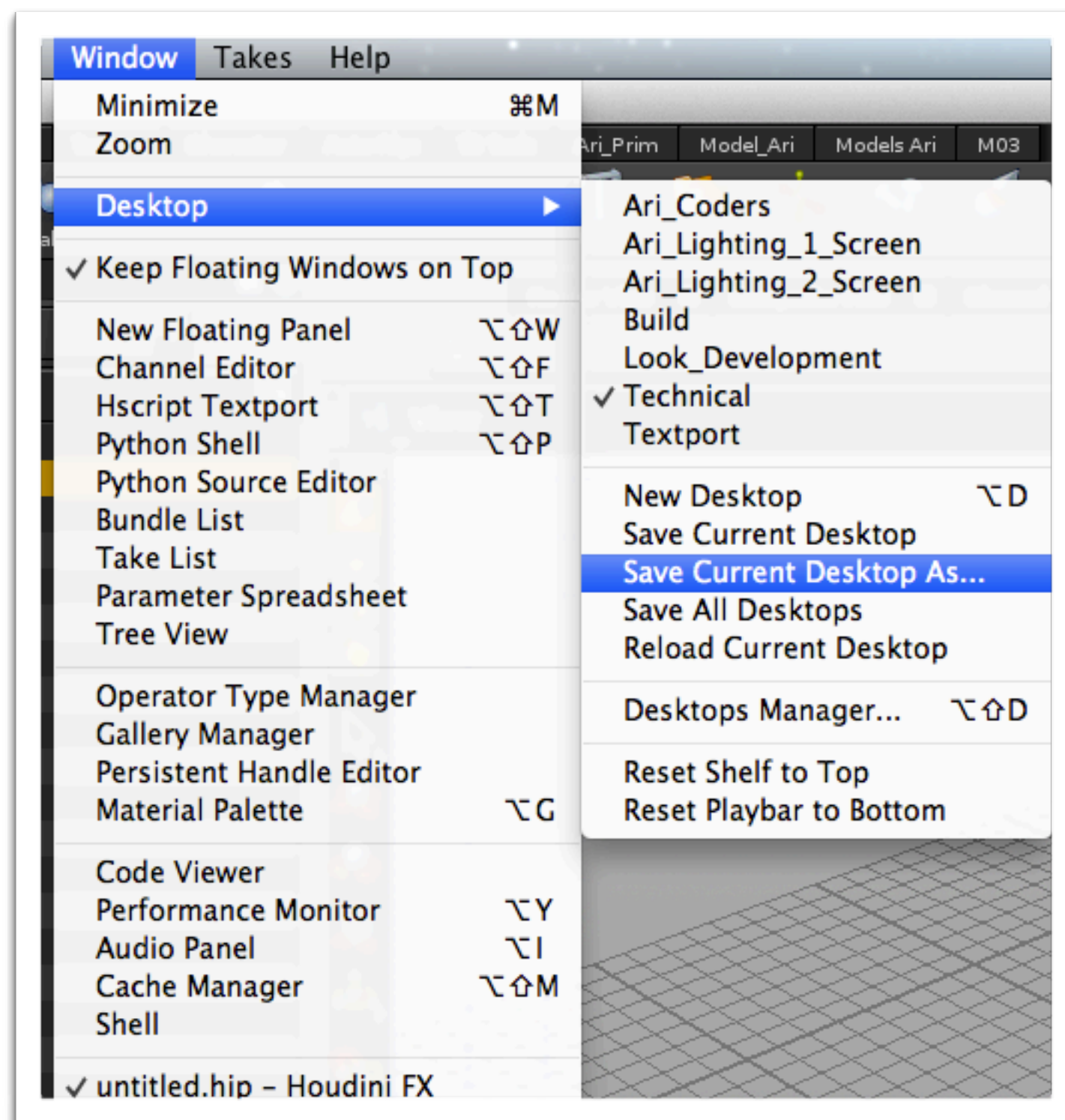
- ▶ Start with Technical Desktop and Modify until you have Desktop shown on next screen
- ▶ Save the Desktop



# Custom Desktop for Lighting and Rendering



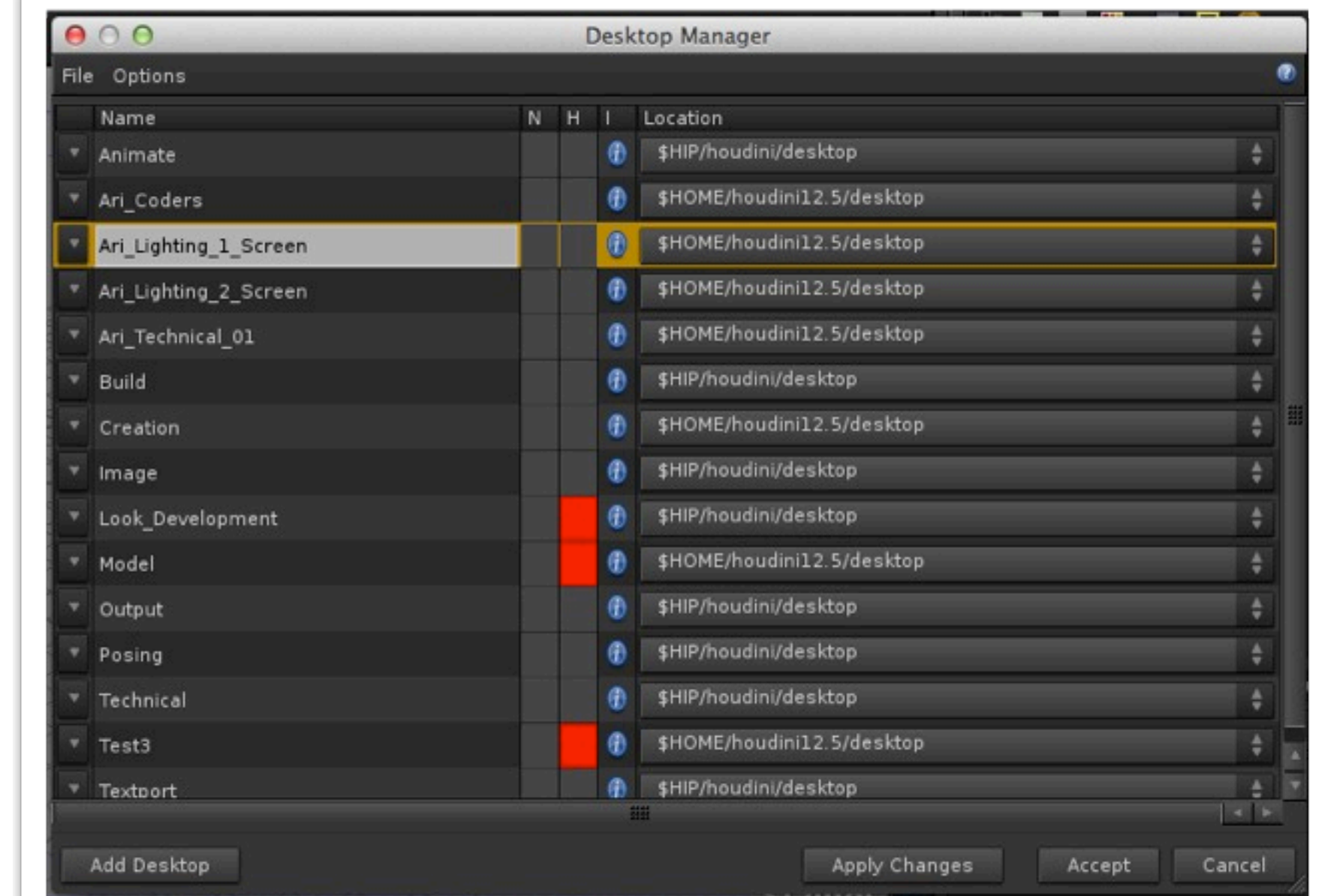
# Save LSR Desktop



- ▶ In the Menus
  - ▶ Window --> Desktop --> Save Current Desktop As..

# Review of the Desktop Manager

- ▶ **Let's Look at the Desktop Manager**
  - ▶ N - No Save flag
    - ▶ When this flag is on, changes to the desk cannot be saved.
  - ▶ H - Hide flag
    - ▶ When this flag is on, the desk does not appear in the search path.
  - ▶ I - Info button (Currently of no real use)
    - ▶ Click this button to show any noteworthy information about the desk and its contents.





# Panes in Brief

- **Render Scheduler** - Shows all active renders, and allows you to pause or kill them.
- **Bundle List** - Lets you create and edit the contents of bundles. Bundles are collections of objects, such as lights, that let you assign them as a group in various places, such as in the light linker.
- **Light Linker** - Lets you link objects to lights as well as do other types of linking
- **Take List** - A hierarchical list of takes. This pane gives lets you edit the list of takes. It can also be easier to select takes from the list in this pane than from the takes menu in the top right.
- **Parameter Spreadsheet** - A hierarchical list of takes. This pane gives lets you edit the list of takes. It can also be easier to select takes from the list in this pane than from the takes menu in the top right.

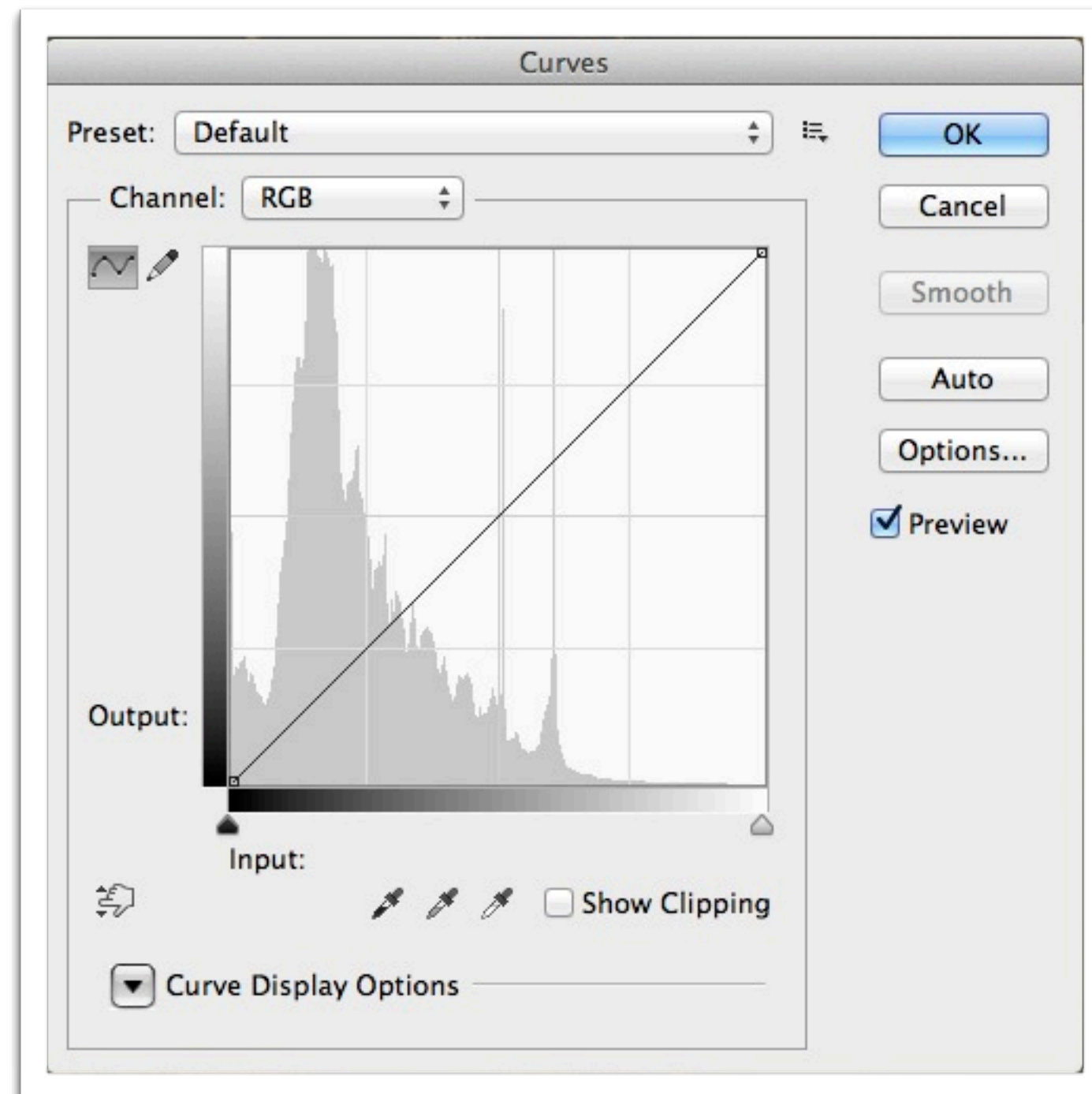


# Setting up a Linear Light Workflow

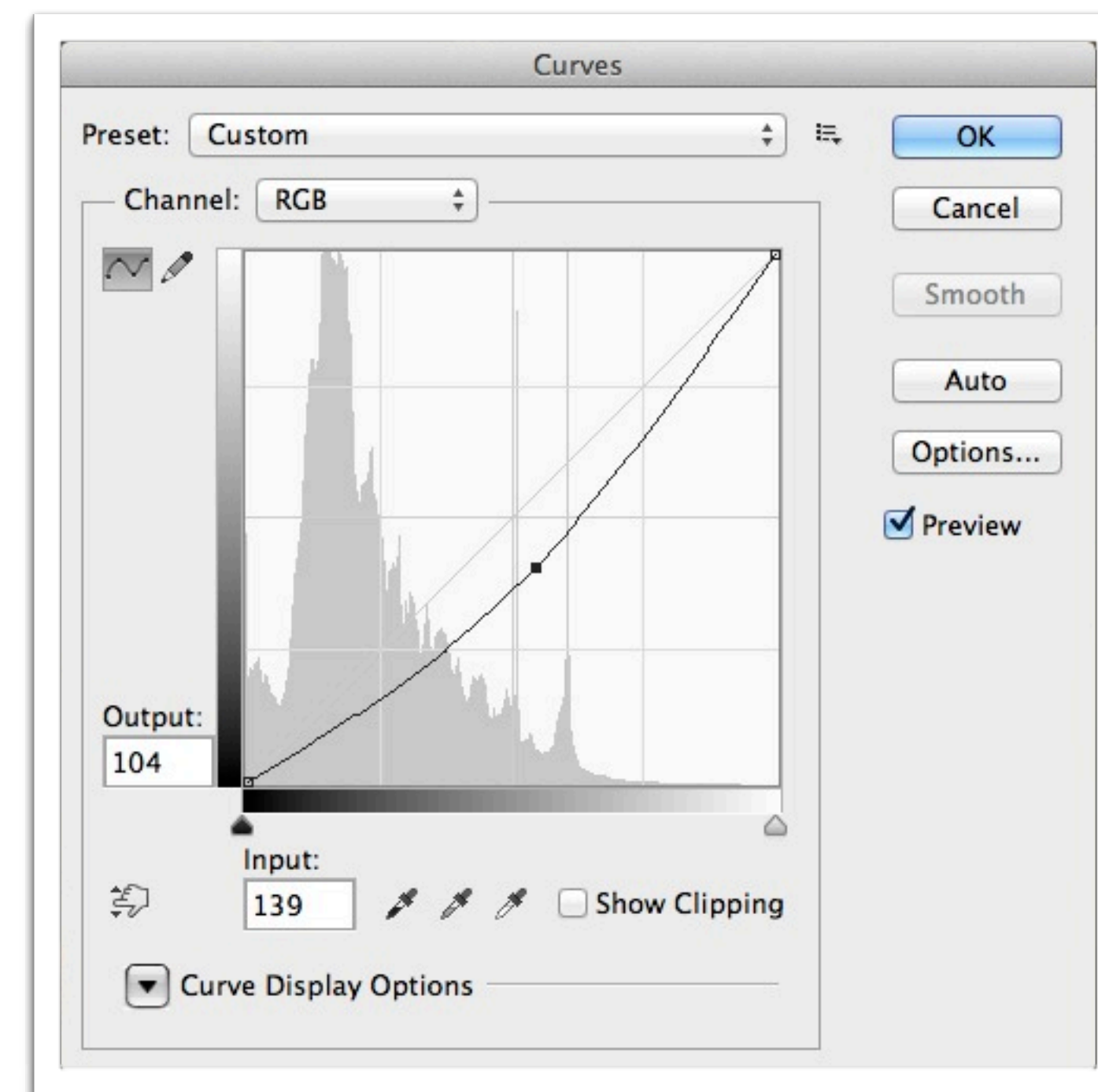
# What is Gamma Correction and Linear Workflow?

- “Gamma correction controls the overall brightness of an image. Images which are not properly corrected can look either bleached out, or too dark. “ - from SIGGRAPH link below
- **Link to explanations**
  - [http://www.siggraph.org/education/materials/HyperGraph/color/gamma\\_correction/gamma\\_intro.html](http://www.siggraph.org/education/materials/HyperGraph/color/gamma_correction/gamma_intro.html)
  - <http://www.youtube.com/watch?v=JJwoPhvQZBE>
- **Bottom line - You want to work in Linear Space**
  - Even after setting up Houdini for a Linear workflow you will have to correct all color maps brought into a material (future module lesson)
  - The only exception is HDRI maps which are already in linear space

# Linear vs Gamma Corrected

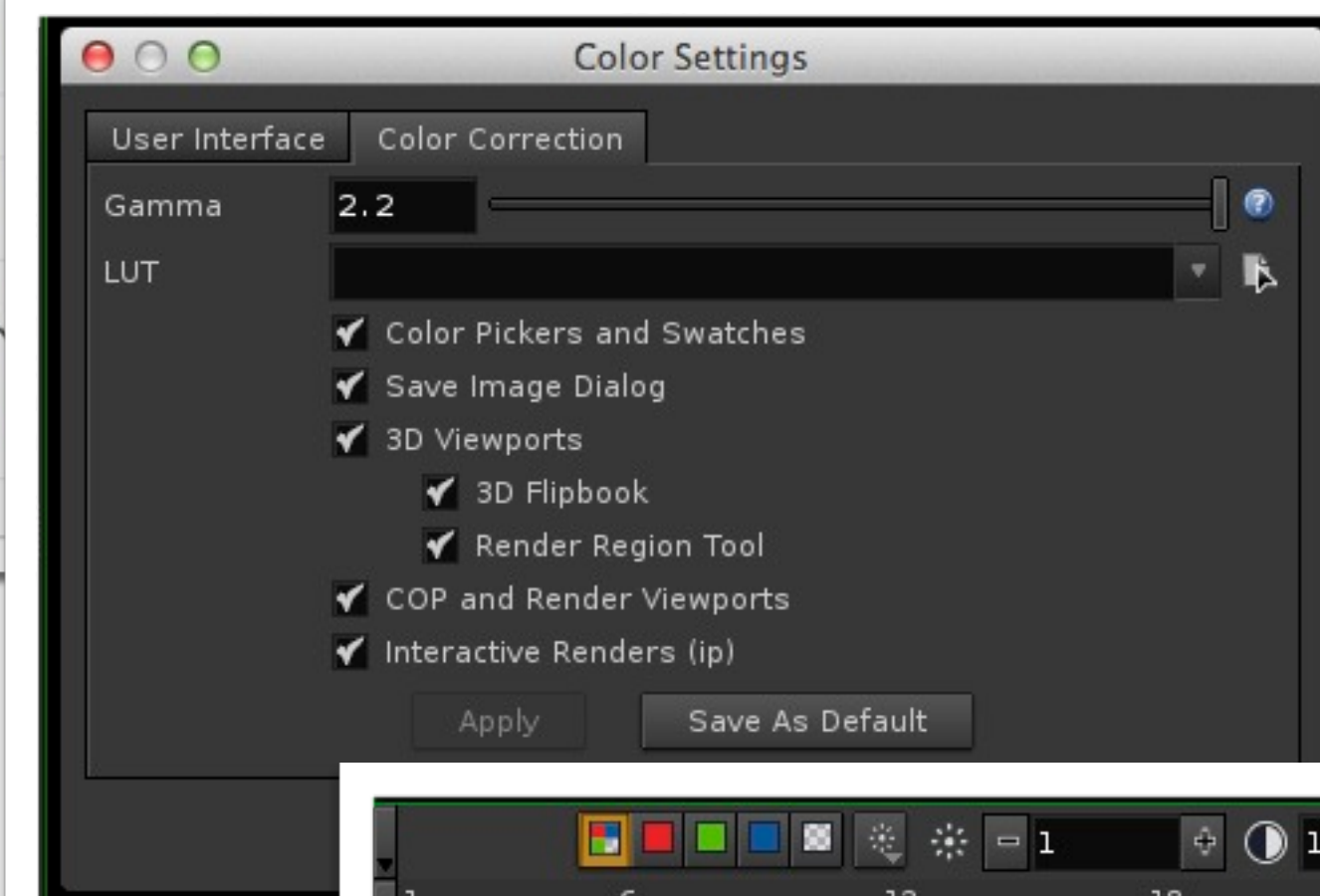
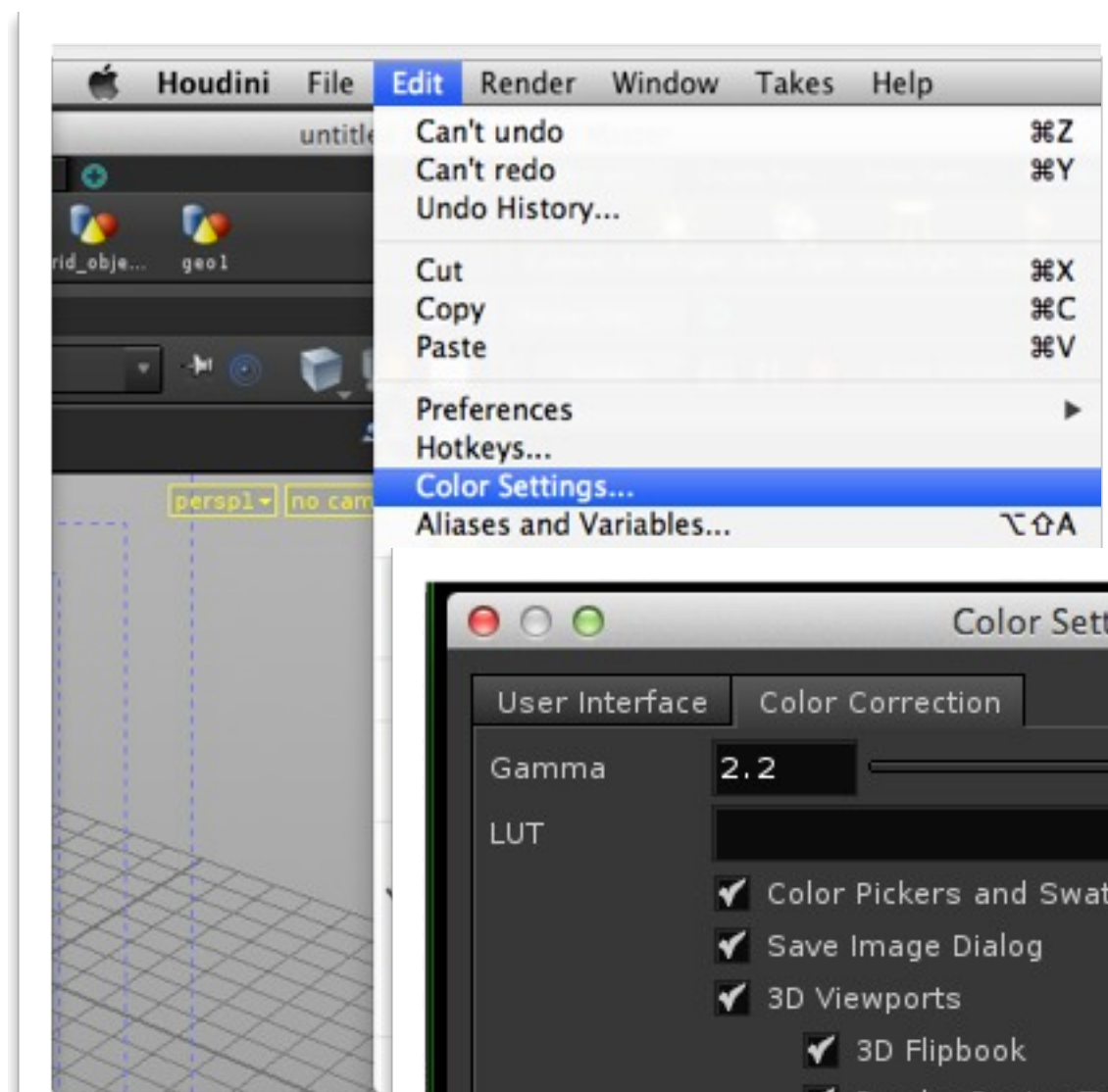


Linear



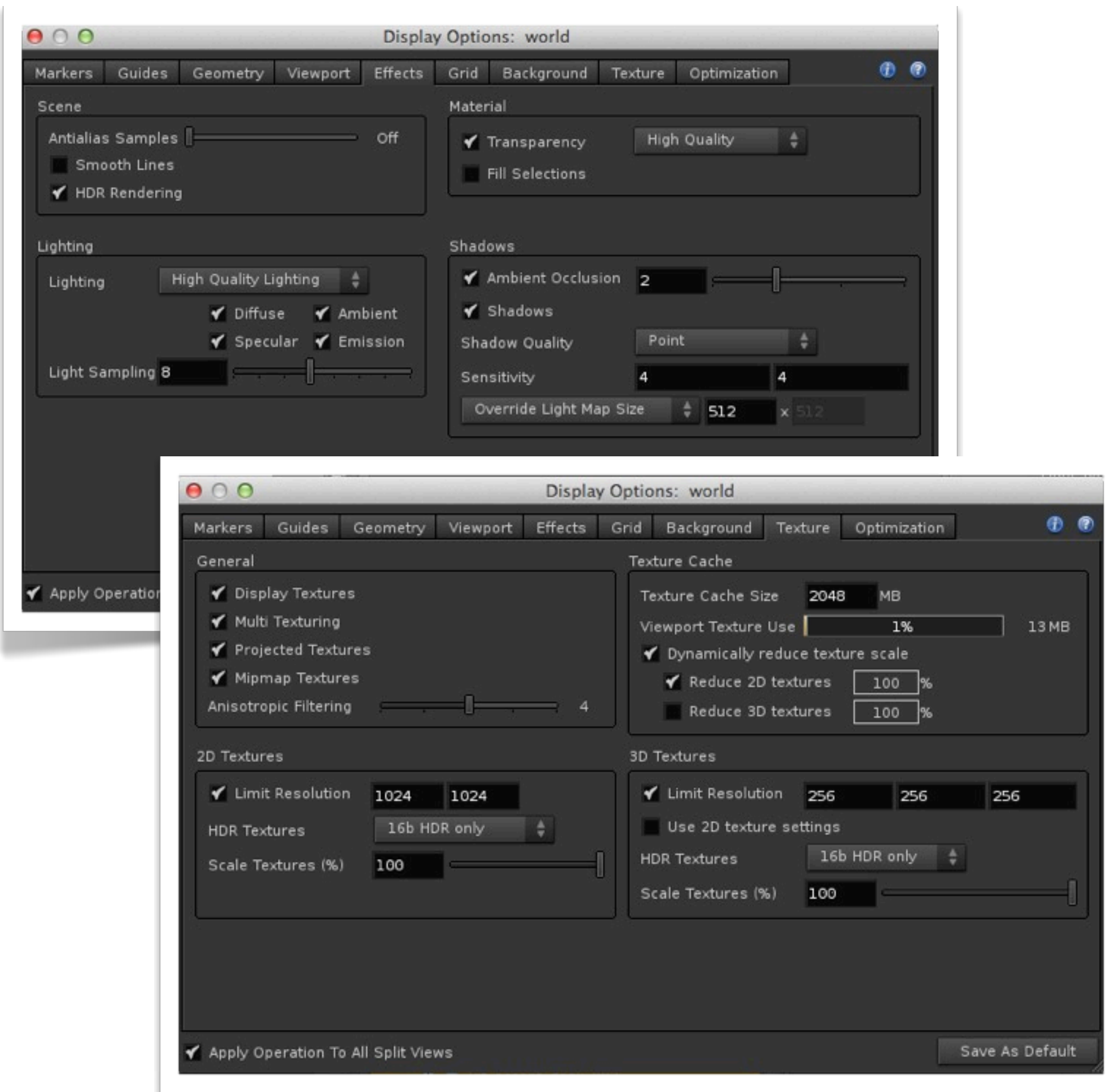
Gamma Corrected

# Setting up Houdini for Linear Light Workflow



- ▶ Go into Color Settings
- ▶ In the Color Correction Tab
  - ▶ Set Gamma to 2.2
  - ▶ Click all options (e.g., 3D Viewport) to on
  - ▶ Click “Save As Default”
- ▶ Open the IPR Render View
  - ▶ Make sure color correction toolbar is visible

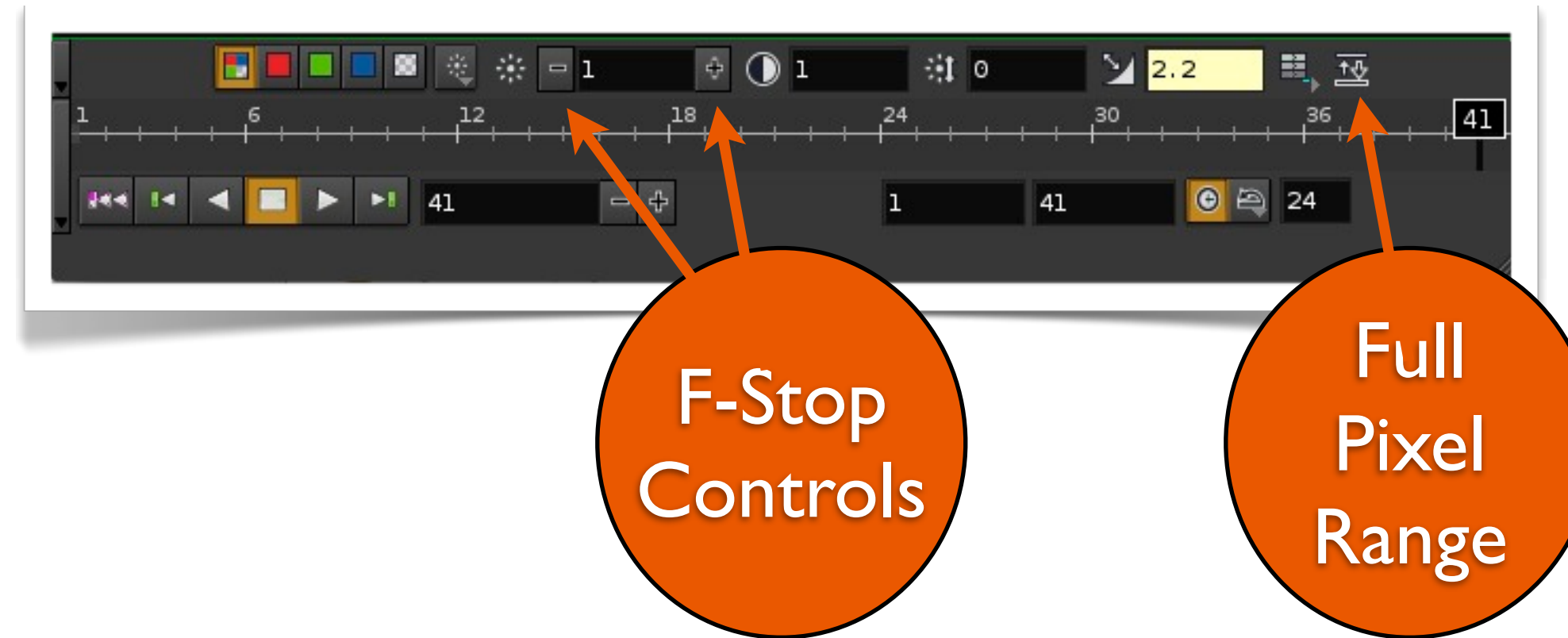
# Linear Light Workflow (cont.)



- ▶ In 3D Scene Display Options - “d” KB shortcut
  - ▶ In the Effects Tab
    - ▶ Set Scene to HDR Rendering
    - ▶ Set Lighting to “High Quality Lighting”
    - ▶ Set Material Transparency to “High Quality”
  - ▶ In the Texture Tab
    - ▶ 3D Textures - Limit resolution Turned on
    - ▶ 3D Textures - HDR Textures set to 16b HDR only
  - ▶ Save As Default

*These settings might not work on older graphic cards*

# Linear Light Workflow (cont.)

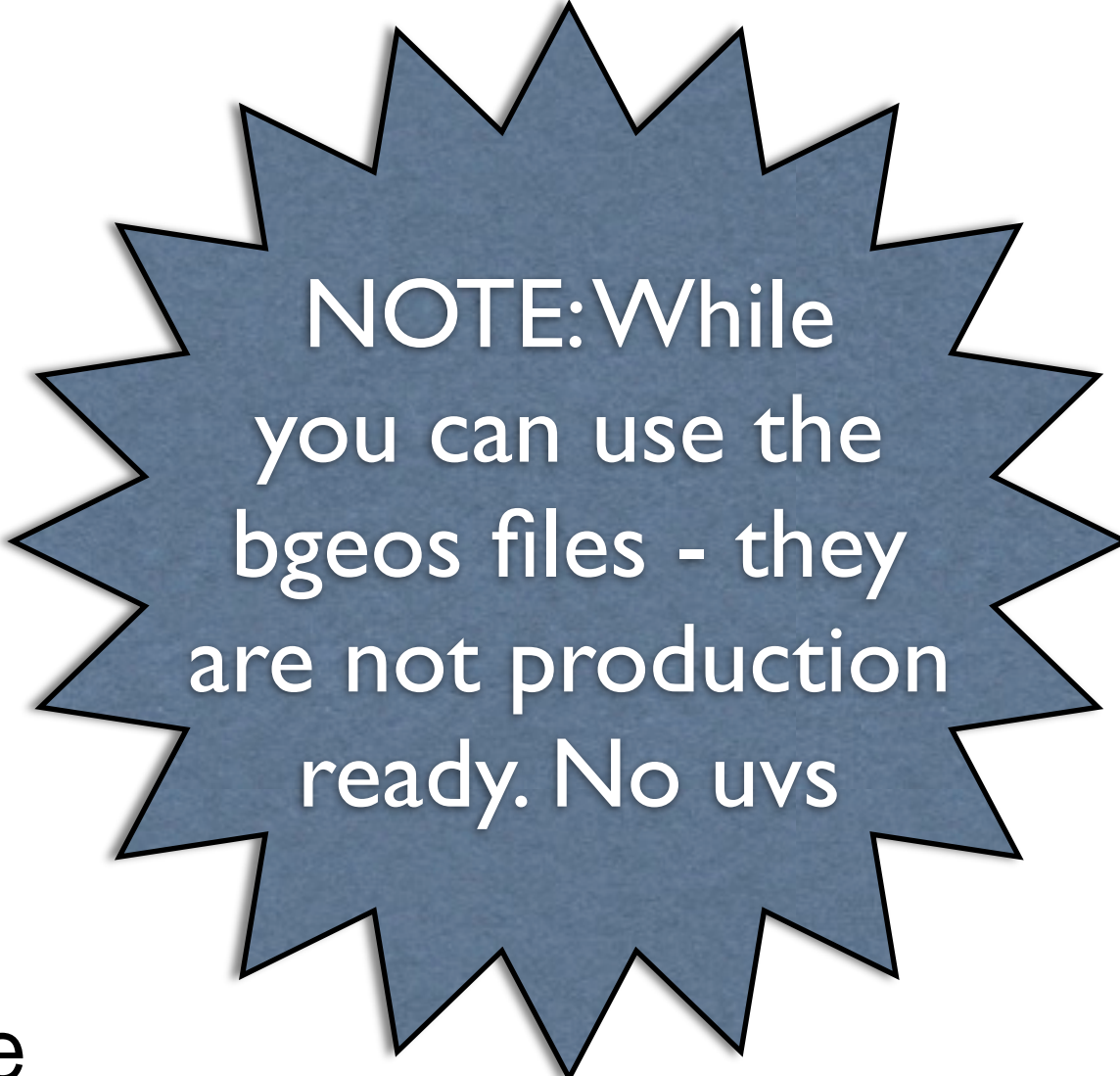


- ▶ Brightness Controls - If Linear Light Workflow is setup then the plus/minus buttons on the brightness control are actually f-stop controls.
- ▶ Adding 1 to the brightness will double the light
- ▶ Subtracting 1 to the brightness will halve the light
- ▶ Full Pixel Range - Adjusts the black and white points to fit the minimum and maximum pixel values of the current image.



# Desktop Tour

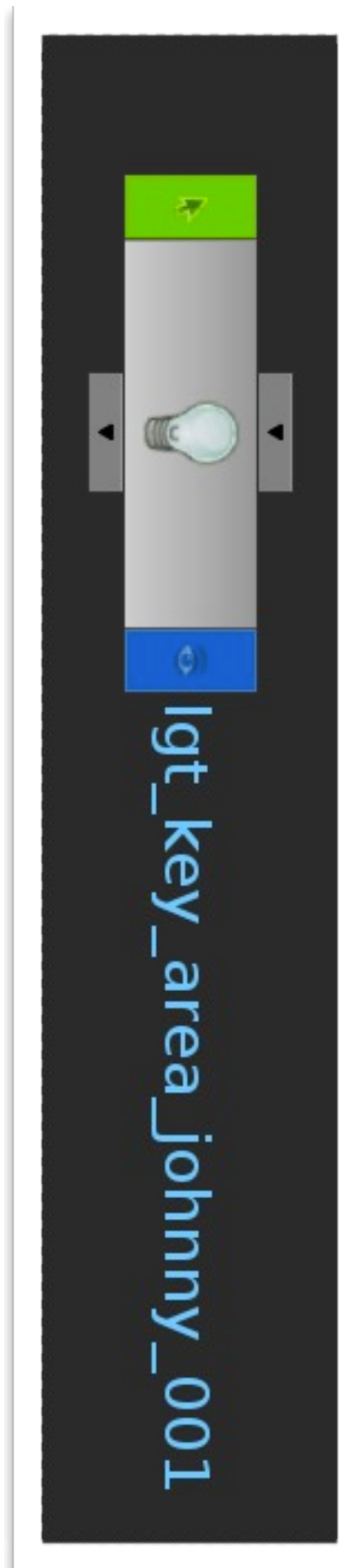
# Open LSR.M02.001.001



NOTE: While you can use the bgeos files - they are not production ready. No uvs

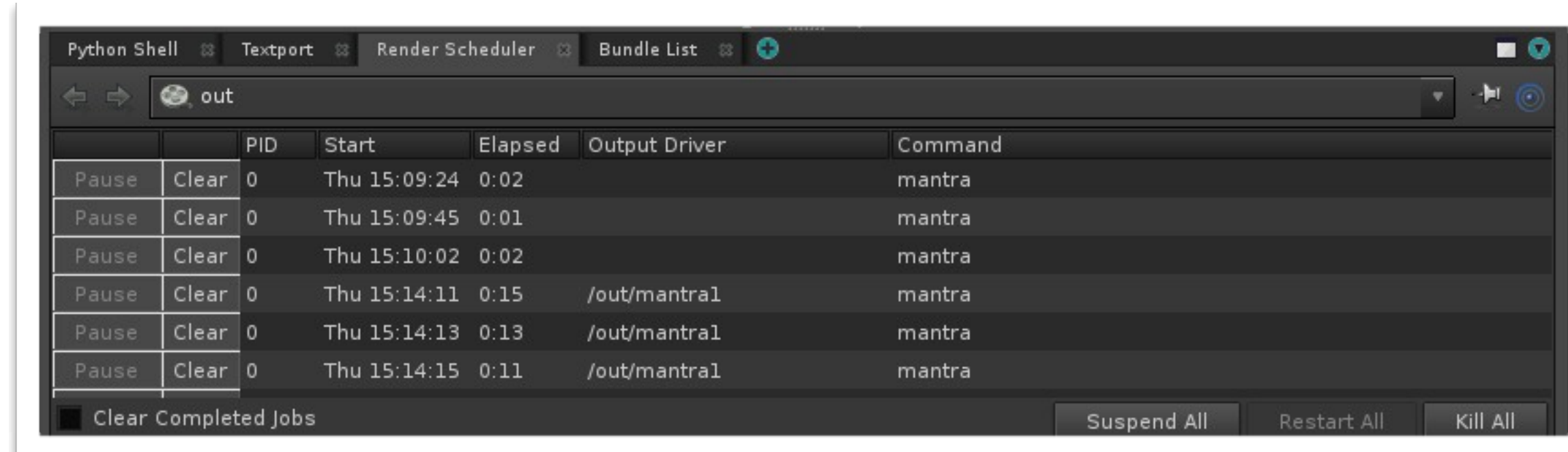
- ▶ In the terminal go to the project folder - In my case I type
  - ▶ `cd /Users/aridanesh/LSR_Projects/LSR.M01.001.002`
  - ▶ `houdini -s LSR_Desktop M01_start.hip`

# Naming Convention for Lights



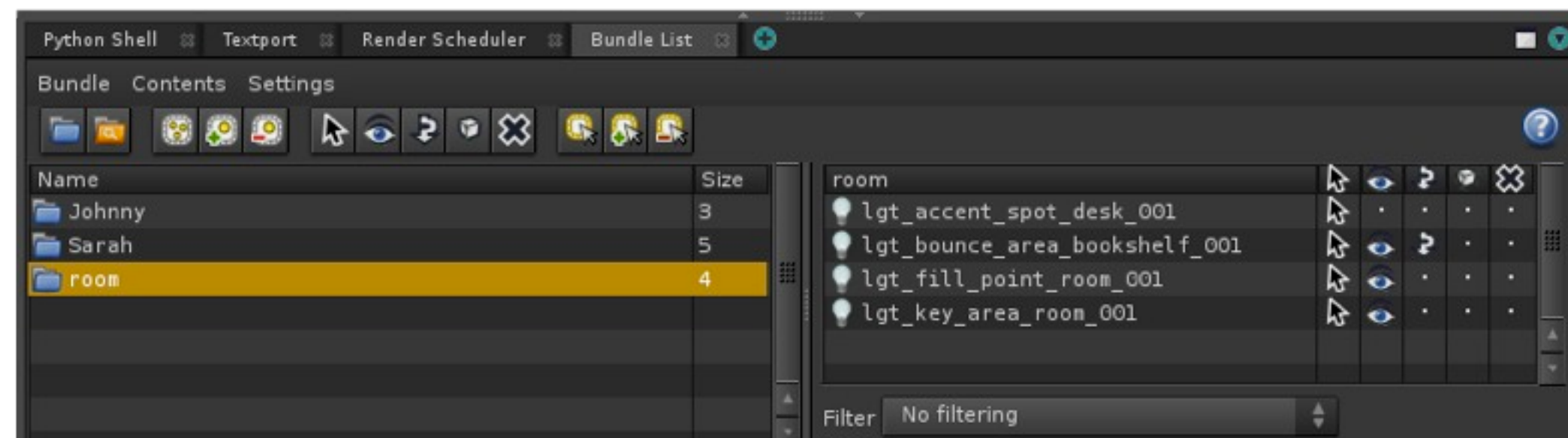
- In this course all lights will be named with the following naming convention
  - **lgt\_use\_type\_description\_number**
  - lgt - designating object is a light
    - This makes it easy to:
      - create OP Masks in the Parameter Sheet
      - create Smart Bundles
  - use - Is the light a key, fill, rim, accent....
  - type - Point, Area, Spot, Env light
  - description - living room, Joey, Mustang
  - number - multiple lights assigned to same description
  - **Example - lgt\_key\_area\_joey\_face\_001**

# Render Scheduler

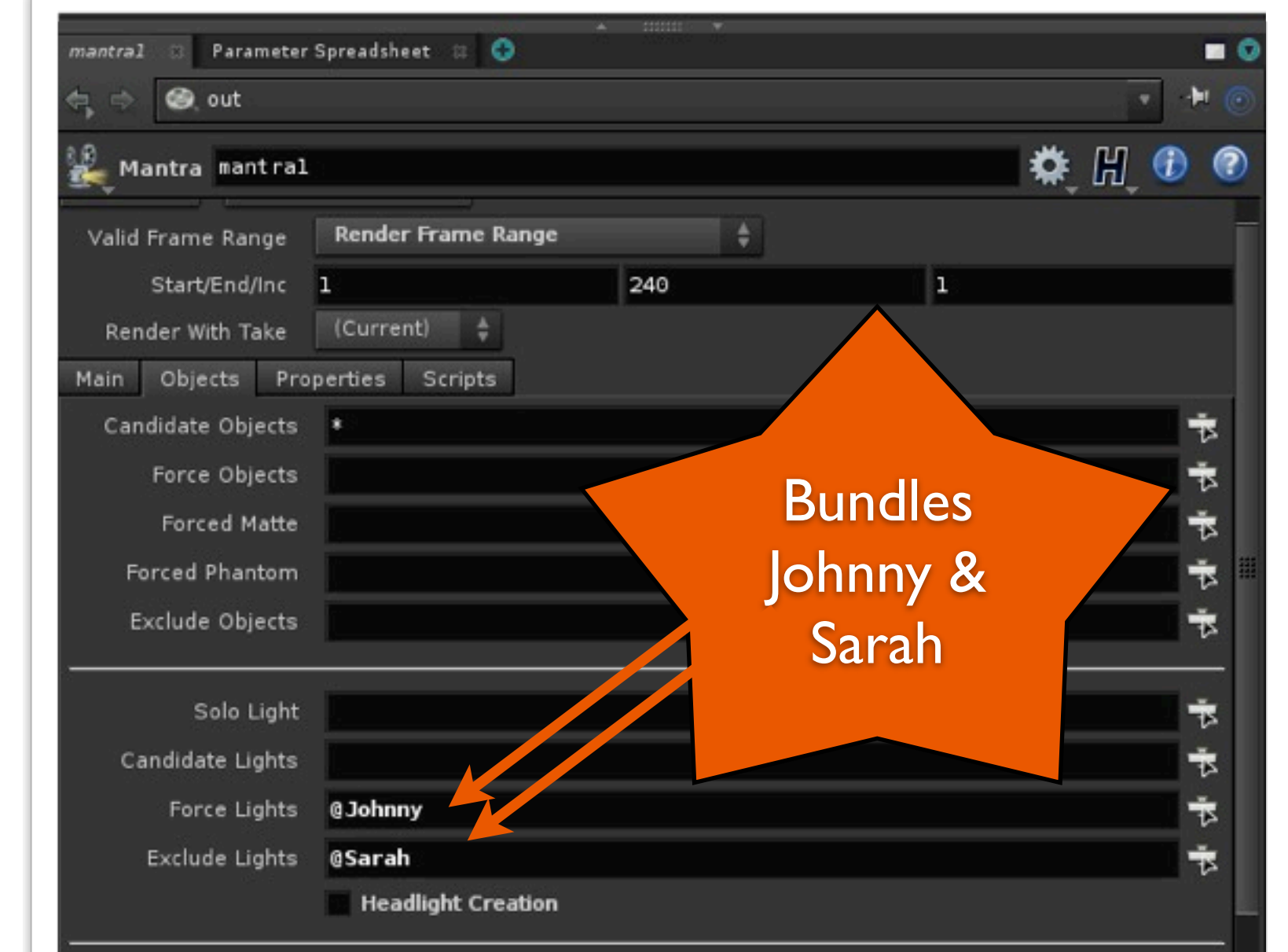


- ▶ This pane shows all active renders, and allows you to pause or kill them
- ▶ Pause - Pause individual render
- ▶ Clear - Kills individual render
- ▶ PID - Process ID
- ▶ Start - Start time of Render
- ▶ Elapsed - Elapsed Time of Render
- ▶ Clear Completed Jobs - When Job is done clear from panel
- ▶ Suspend All - Pause all renders
- ▶ Kill All - Kills all renders

# Bundle List



- ▶ Bundles allow you to refer to a group of objects as a single unit. This is useful in situations such as lighting where you can link a bundle of lights all at once.
- ▶ You can use a bundle name prefixed with a @ sign (for example, @lights) to stand for the contents of the bundle in any parameter that accepts a list of nodes.



# Example - Create Lights for Outhouse

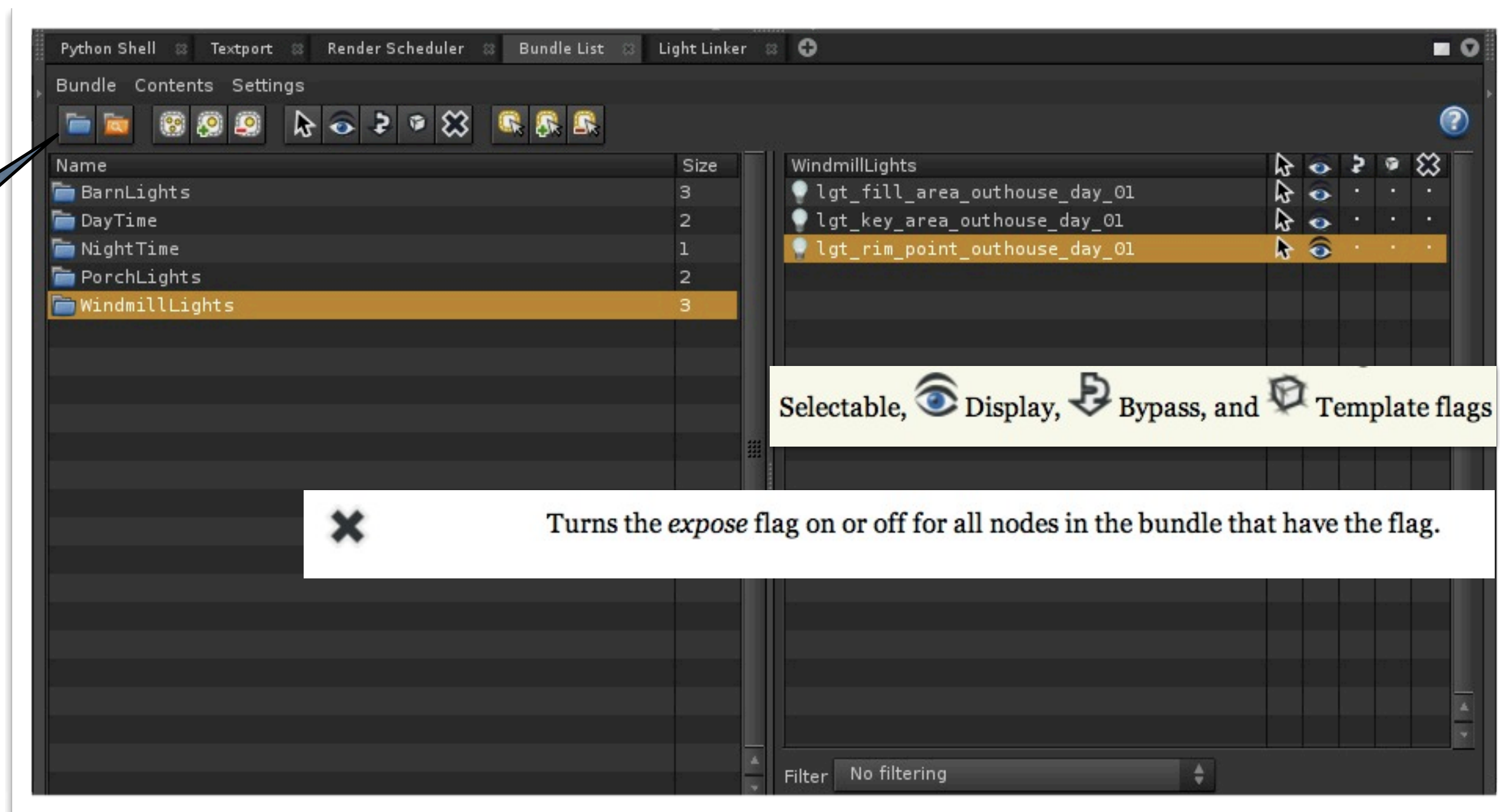
**Remember naming convention - lgt\_use\_type\_description\_number**

- **Deselect all objects except for outhouse**
- **Create a Key Light - I used an area light**
  - Name it - lgt\_key\_area\_outhouse\_day\_01
- **Create a Fill Light**
  - Name it - lgt\_fill\_area\_outhouse\_day\_01
- **Create a Rim Light - I used a point light**
  - Name it - lgt\_rim\_area\_outhouse\_day\_01

# Now Make a Bundle

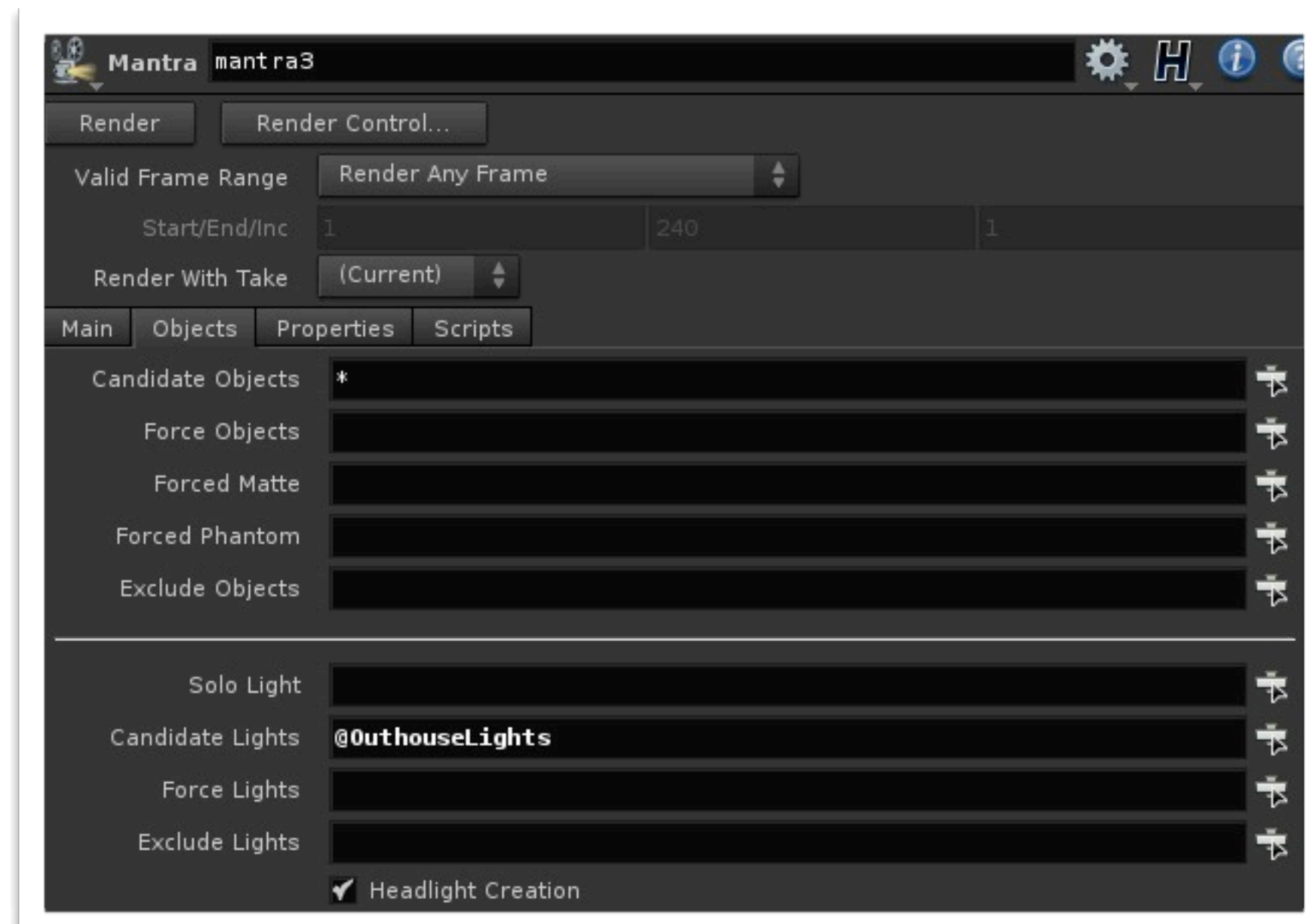
- ▶ Click on the Bundle Tab
  - ▶ Click on the New Bundle Icon
  - ▶ Name it OuthouseLights
  - ▶ Drag and Drop Lights into Bundle

New  
Bundle  
Icon

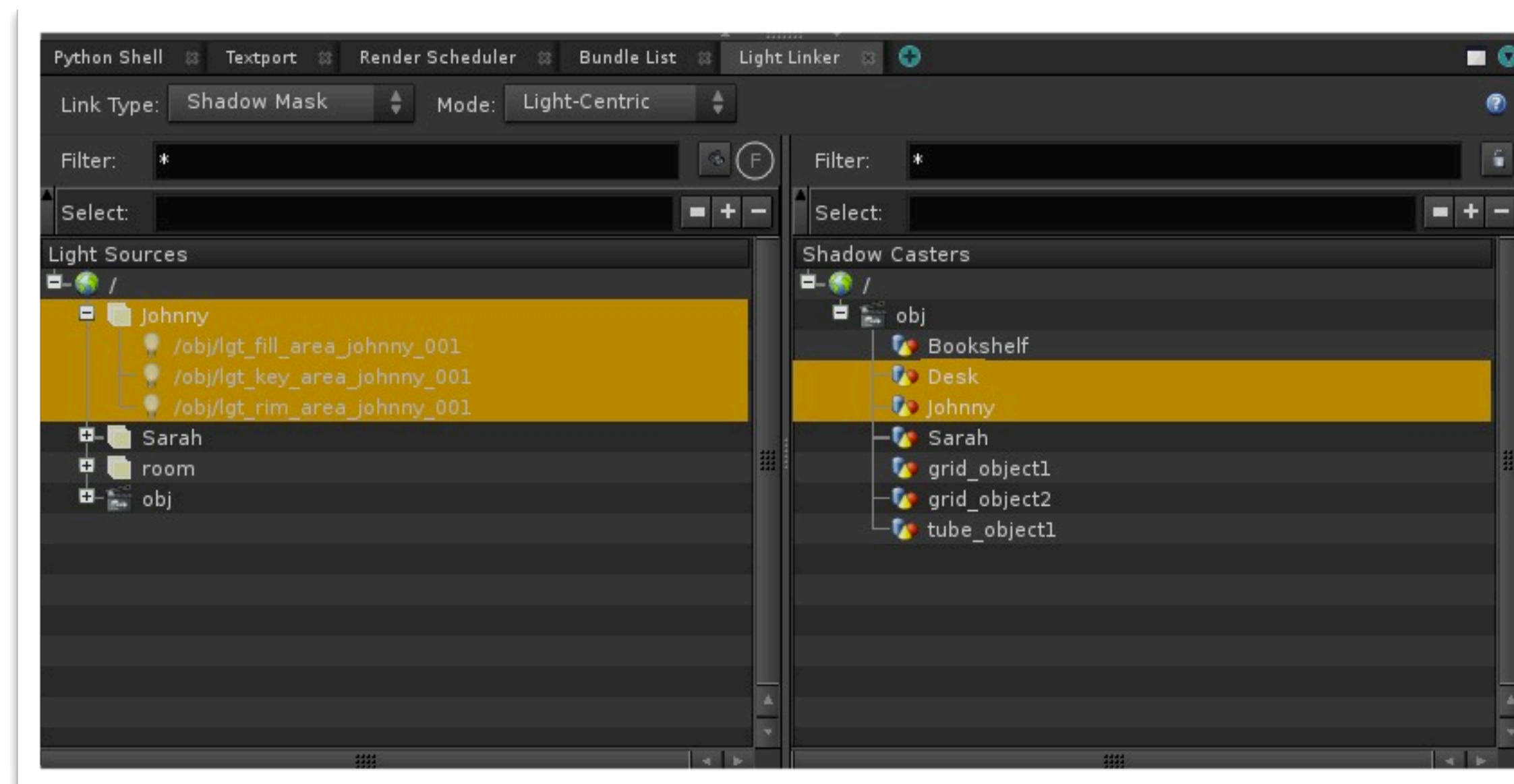


# Referencing a Bundle

- ▶ Let's render with only the Outhouse Lights
  - ▶ Drop Down a Mantra Node
  - ▶ In the Objects Tab
    - ▶ In the “Candidate Lights” parameter
    - ▶ Type @OuthouseLights
  - ▶ The “@” is the symbol to point to a bundle



# Light Linker

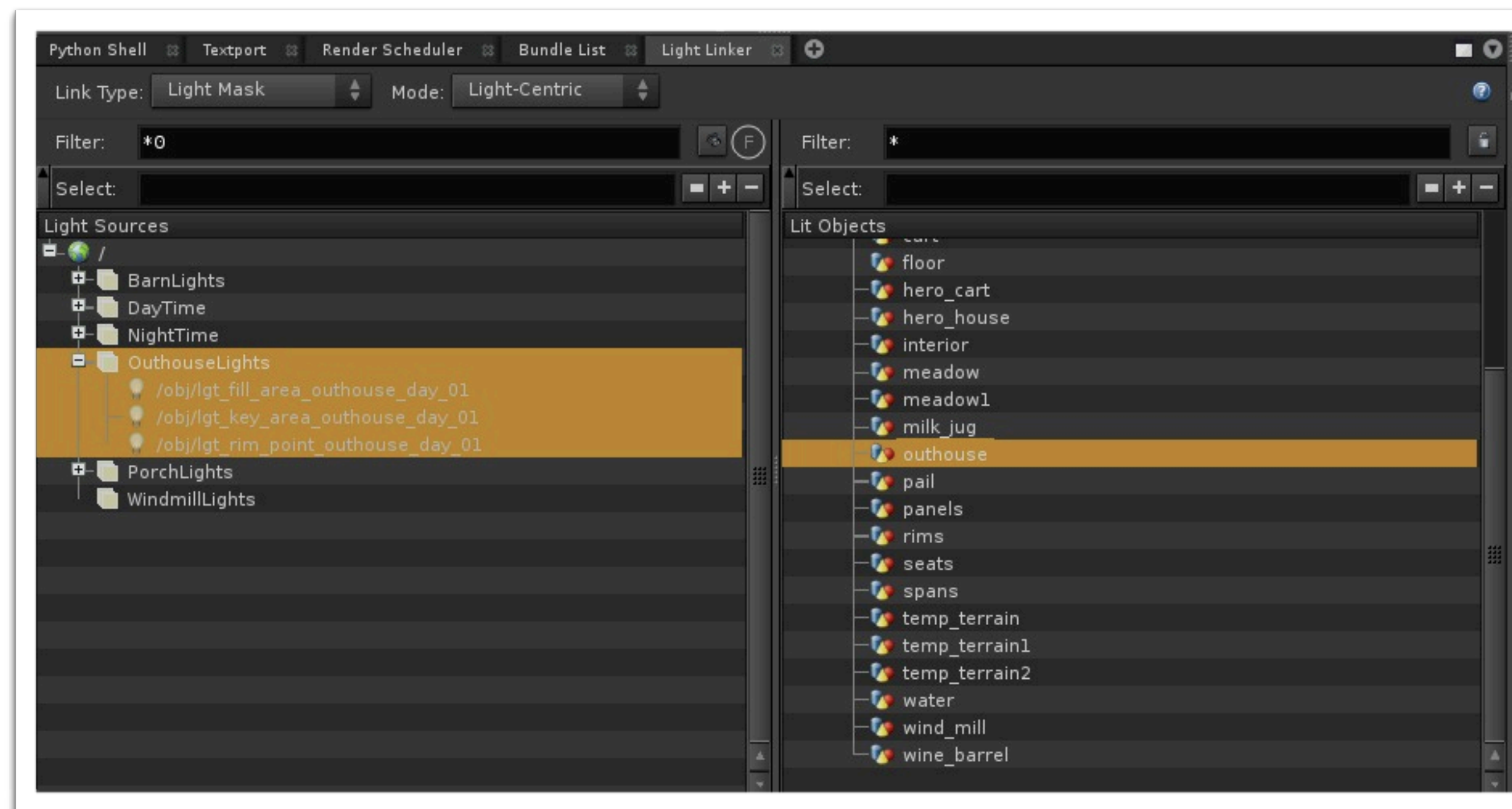


- The light linker lets you link lights to objects to control which lights illuminate which objects (Light Mask), which objects cast shadows from which lights (Shadow Mask), and which objects are reflected in other objects (Reflection Mask).
- *Remember Mantra is a Light-Centric Render Model*

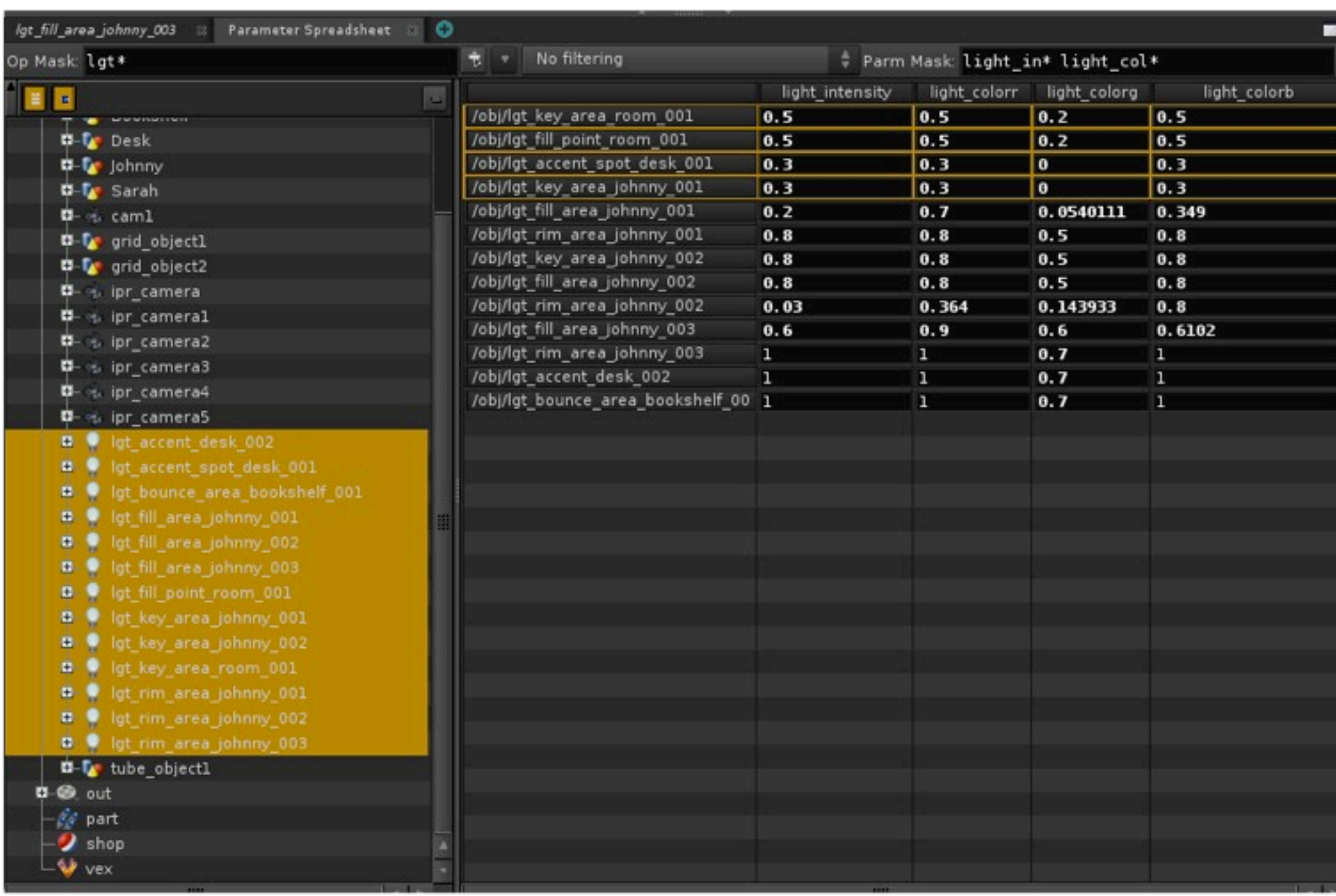
# Making Sure Outhouse Lights...

Only illuminate the Outhouse

- ▶ Select Outhouse Light on Left Panel
- ▶ On Right Panel Select Outhouse



# Parameter Spreadsheet



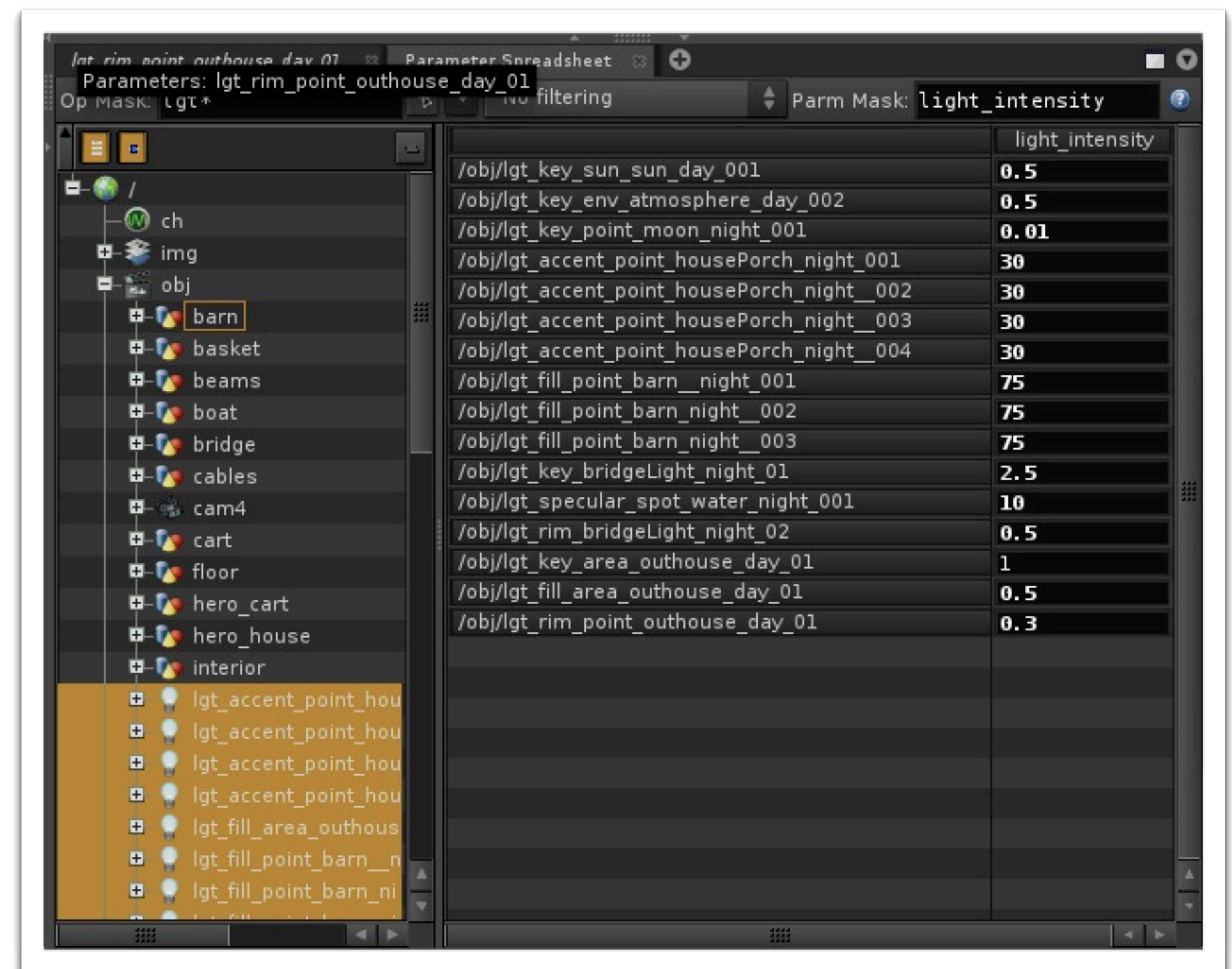
The screenshot shows the Houdini Parameter Spreadsheet window. The left pane displays a tree of nodes, with 'lgt\_fill\_area\_johnny\_003' selected. The right pane shows a table of parameters for the selected node and others. The table has columns for 'light\_intensity', 'light\_colorr', 'light\_colorg', and 'light\_colorb'. The 'Op Mask' is set to 'lgt\*' and the 'Parm Mask' is set to 'light\_in\* light\_col\*'. The table lists parameters for various light nodes, including 'lgt\_key\_area\_room\_001', 'lgt\_fill\_point\_room\_001', 'lgt\_accent\_spot\_desk\_001', 'lgt\_key\_area\_johnny\_001', 'lgt\_fill\_area\_johnny\_001', 'lgt\_rim\_area\_johnny\_001', 'lgt\_key\_area\_johnny\_002', 'lgt\_fill\_area\_johnny\_002', 'lgt\_rim\_area\_johnny\_002', 'lgt\_rim\_area\_johnny\_003', 'lgt\_accent\_desk\_002', and 'lgt\_bounce\_area\_bookshelf\_001'.

	light_intensity	light_colorr	light_colorg	light_colorb
/obj/lgt_key_area_room_001	0.5	0.5	0.2	0.5
/obj/lgt_fill_point_room_001	0.5	0.5	0.2	0.5
/obj/lgt_accent_spot_desk_001	0.3	0.3	0	0.3
/obj/lgt_key_area_johnny_001	0.3	0.3	0	0.3
/obj/lgt_fill_area_johnny_001	0.2	0.7	0.0540111	0.349
/obj/lgt_rim_area_johnny_001	0.8	0.8	0.5	0.8
/obj/lgt_key_area_johnny_002	0.8	0.8	0.5	0.8
/obj/lgt_fill_area_johnny_002	0.8	0.8	0.5	0.8
/obj/lgt_rim_area_johnny_002	0.03	0.364	0.143933	0.8
/obj/lgt_fill_area_johnny_003	0.6	0.9	0.6	0.6102
/obj/lgt_rim_area_johnny_003	1	1	0.7	1
/obj/lgt_accent_desk_002	1	1	0.7	1
/obj/lgt_bounce_area_bookshelf_001	1	1	0.7	1

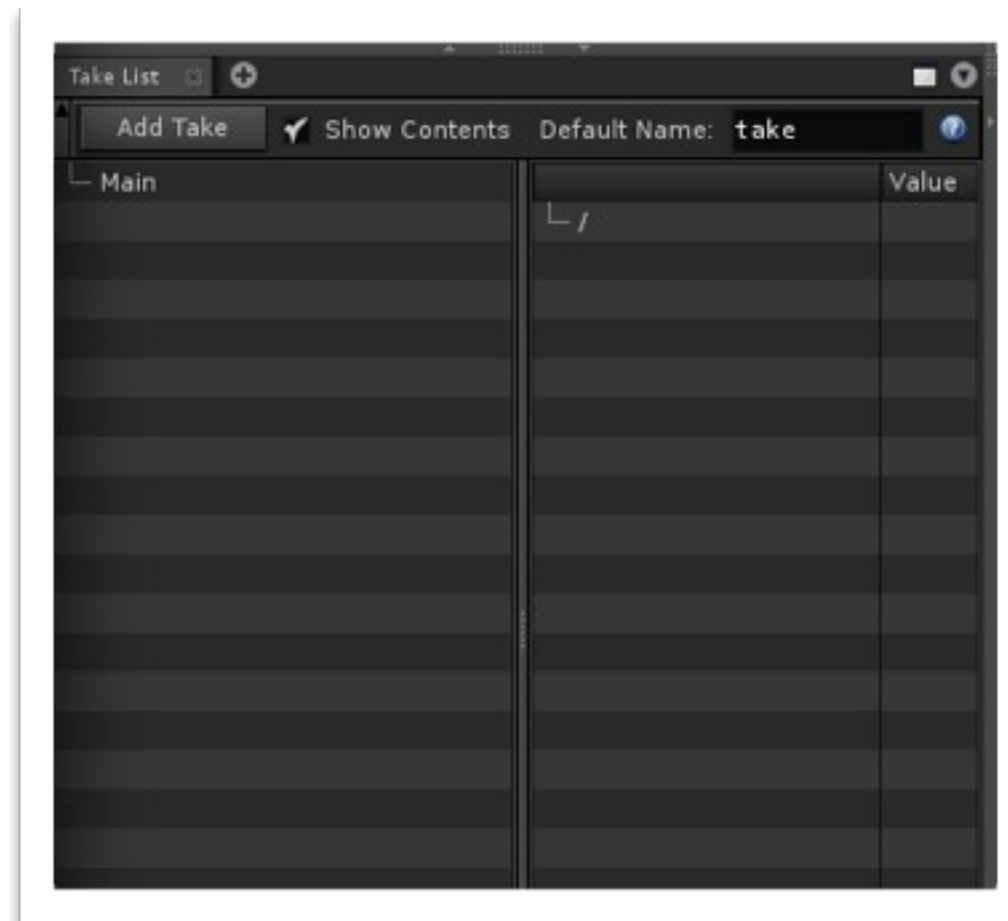
- ▶ The parameter spreadsheet shows a flat list of rows representing nodes in the network, and columns representing the flags and parameters (and parameter components) of each node.
- ▶ A quick way to modify parameters from multiple lights or objects at once.

# Parameter Spreadsheet Example

- In the Op Mask type
  - lgt\*
- In the Parm Mask type
  - light\_intensity
- Now you can quickly change the intensity of multiple lights



# Take List



- ▶ Takes are useful for:
- ▶ Exploring alternative designs with the ability to switch back and forth to compare the alternatives.
- ▶ Layering tweaks or explorations on top of a “known good” or approved shot/design. You can see which parameters you've changed and easily return to, copy-paste from, etc., the original scene.
- ▶ Holding different sets of parameters for different render passes. You can use the Render with take parameter on render nodes to control which take to use when you render.



# 123 and 456 Explained

# Using Command Line... Commands

`opcf` - change directory

`opadd` - add a node

`opparm` - change a parameter

Open a textport and  
use the command  
“commandecho on”  
to help learn HScript

- Create a new hip file
- `houdini -s Technical`
- Goto the Textport
  - type - `commandecho on`
  - type `opcf /obj`
  - type - `opadd geo myBall`
  - type - `opcf myBall`
  - type - `opadd sphere mySphere`
  - type - `opparm mySphere tx (3.0)`
  - type -



123.cmd & 456.cmd

```
# Variables set with the set command are local to this script only.
# Therefore, we don't have to worry about cleaning up.
alias cd      opcf
alias h       history
alias l       ls
alias pwd     oppwf
alias rm      oprm
alias cp      oppc
alias mv      oppm
alias hython  python
```

- ▶ **Houdini runs 123.cmd when Houdini is first launched**
- ▶ It does not run again if you choose File -> New
- ▶ Sets up the initial Houdini Environment (including loading the initial geometry)
- ▶ File can be found at: \$HFS/houdini/scripts/123.cmd
- ▶ **DO NOT EDIT THIS FILE** - copy and edit higher in the hierarchy
- ▶ Example edit the file in \$HOME/houdiniversion/scripts
- ▶ Better yet put in shots folder
- ▶ **Houdini runs 456.cmd after you open a saved scene or execute File -> New**
- ▶ Does not exist as a factory install, you must create from scratch
- ▶ Place script in \$HOME/houdiniversion/scripts or shots folder

```
# Set the prompt to a simple prompt
prompt "strcat(opwpr(), " -> ") "

# User Defined Commands
echo "Setting up..."
setenv JOB = /work/training_education_docs/training/ari/2012.06.27_pipeline_setup/
echo $JOB
setenv HSITE = $HOME/houdini.jeff
echo $HSITE
```

- Good place to
  - Create aliases
  - Create environment variables for
    - Maps stored in a show directory not stored locally but on server
    - Shortcut to render output directory
  - Create default
    - Geometry
    - Lights
    - Cameras
    - ROP nodes

```
opcf /obj
opadd -n geo temp_terrain
opparm temp_terrain material (brown)
opcf temp_terrain
opadd grid terrain
oplocate -x 0.0 -y 0.0 -z 0.0 terrain
opparm terrain sizex (50) sizey (120) rows (120) cols (40)
opset -d on -r on -C on terrain

opcf /obj
opadd cam temp_cam
opparm temp_cam tx (0) ty (10) tz (50) rx (10) ry (0) rz (0)
opset -d off -r off -C on -p off temp_cam

viewcamera -c temp_cam LSCR_Lighting_1_Screen.panetab2.world world

fps 24

tset -1 4

frange 1 72
```

- Good place to
  - Create aliases
  - Create environment variables for
    - Maps stored in a show directory not stored locally but on server
    - Shortcut to render output directory
  - Create default
    - Geometry
    - Lights
    - Cameras
    - ROP nodes

# Example 123.cmd

## 123.cmd

```
# Set up the default scene when Houdini starts up
#
source scripts/defaultScene.cmd
echo "loaded defaultScene.cmd"

#
# Create a default /img/comp1 so that we can easily put down COPs
#
opcf /img
opadd img comp1
oplocate -x 0 -y 0 comp1
opcf /

#
# UI control commands
#

# Set the prompt to display the current directory followed by an arrow
prompt `strcat(oppwf(), "Ari -> ")`

# User Defined Commands
echo "Setting up user env"
setenv JOB = /Volumes/ParticleKabob/Render/
echo $JOB
setenv HSITE = $HOME/houdini.ari
echo $HSITE
setenv DEM = /Volumes/ParticleKabob/DEM/
echo $DEM
echo "Completed environment setup"
```

## defaultScene.cmd

```
opcf /obj
# Add default geometry
opadd -n geo ari
opparm ari tx (0.0) ty (5.0) tz (0.0)
opparm ari shop_materialpath (/shop/red)
opcf ari
opadd font font_ari
oplocate -x 0.0 -y 0.0 font_ari
opset -d on -r on -C on -p off font_ari
opparm font_ari text ("Ari")
opcf ..

opcf /shop
opadd v_plastic red
opparm red diff (1.0 0.0 0.0) spec (0.75 0.5 0.5)
opset -C on -p on red
```

opcf - change directory  
opadd - add a node  
opparm - change a parameter

# Example 456.cmd

```
# Make a grid that is a proxy for the terrain
opcf /obj
opadd -n geo temp_terrain
opparm temp_terrain material (brown)
opcf temp_terrain
opadd grid terrain
oplocate -x 0.0 -y 0.0 -z 0.0 terrain
opparm terrain sizex (50) sizey (120) rows (120) cols (50)
opset -d on -r on -C on terrain

opcf /obj
opadd cam temp_cam
opparm temp_cam tx (0) ty (10) tz (50) rx (10) ry (0) rz (0)
opset -d off -r off -C on -p off temp_cam

fps 24
tset -l 4
frange 1 72
```



# Let's create a 456.cmd...

That adds a shopnet to the /obj level and creates a VOPMaterial

- ▶ In a text editor type
  - ▶ opcf /obj
  - ▶ opadd shopnet myMaterials
  - ▶ opcf obj/myMaterials
  - ▶ opadd vopmaterial generic
- ▶ save the file as 456.cmd and put in scripts folder

- ▶ use `hconfig - ap` at the terminal to see all your environment variables

Variable	Permissions	Description
\$HIP	R	Defaults to the directory where you started Houdini
\$JOB	R+W	A custom variable that determines where your jobs are located.
\$HOME	R	Your home directory
\$HFS	R	Houdini File Structure -The directory where Houdini is installed



# Sourcing in Cameras

- **Assumption - Camera is locked. The lighting artist is not allowed to touch it.**
- **The camera has been saved either as:**
  - a. an OTL
  - b. an hscript command
  - c. a Python script
- **The camera has been saved to either a local scripts folder or on the server somewhere**
- **To save a camera named cam1 into the local scripts/obj folder. In this example I will call the command `cam_M01_shot001.seq001.cmd`**
- **In the Textport write:**
  - `opscript -b /obj/cam1 > $HIP/scripts/obj/cam_shot001.seq001.cmd`

# Sourcing in Cameras (cont.)

- **opscript -b /obj/cam1 > \$HIP/scripts/obj/cam\_shot001.seq001.cmd**
  - The opscript command prints the commands necessary to recreate an operator.
  - For more information on hscript commands
    - <http://www.sidefx.com/docs/houdini12.0/commands/>
    - [http://www.sidefx.com/docs/houdini12.0/commands/\\_guide](http://www.sidefx.com/docs/houdini12.0/commands/_guide)
- **To read the camera back into a new hip file use the source command**
- **source - Executes the script commands in a file.**
  - `source $HIP/scripts/obj/cam_shot001.seq001.cmd`