Command-line Rendering and GPU Selection

When rendering from the command-line of your 3d app with Redshift, you can specify the GPU devices to use for the rendering job. When specifying the GPU devices from the command-line, the Redshift preferences.xml file is not updated, so running your 3d app in interactive mode will still use the GPU devices that you specified in the System tab of the Redshift render options.

Several render managers including Deadline and Royal Render natively support GPU selection when rendering with Redshift. Selecting only a subset of available GPUs for a job is useful for example to render multiple frames simultaneously on a single machine for optimal scaling.

Prerequisites (Windows only)

Your Redshift installation for Maya needs to be configured to run correctly from the command-line. This step is required for all command-line rendering with Maya and Redshift and is not directly related to command-line GPU selection. This requires that the full path to Maya's *Render.exe* must be part of your PATH environment variable. If you have multiple versions of Maya installed on your machine, the path to the Maya version you are rendering with must come before the path to other Maya versions.

Syntax

```
Render.exe -r redshift -gpu <device_id_array> <scene_file>
```

Where <device_id_array> is a list of the GPU device ids you wish to render with. The syntax for this array is 'mel style' - for example use {0,2} to render on devices 0 and 2.

Example

To render scene c:\path\to\scenetorender.ma on GPU device 1 only

```
Render.exe -r redshift -gpu {1} c:\path\to\scenetorender.ma
```

Same scene rendered using GPU device 0 and 1:

```
Render.exe -r redshift -gpu {0,1} c:\path\to\scenetorender.ma
```

On Linux and macOS, the curly braces around the GPU device id array may need to be escaped to prevent the shell from changing the syntax.

```
Render -r redshift -gpu \{0,1\} /path/to/scenetorender.ma
```

Prerequisites

Due to the nature of the fixed command-line parameters available for 3dsmaxcmd.exe, GPU selection for command-line rendering requires the GPU selection to be coded into a pre-render script, which is then passed to 3dsmaxcmd.exe and executed prior to rendering.

Syntax

```
3dsmaxcmd.exe <scene_file> -preRenderScript:<script_file>
```

Where <script_file> is the relative path to a MaxScript file containing a call to the command *rsSetCudaDevices*. The rsSetCudaDevices command takes a single array parameter in MaxScript synyax. The creation of the MaxScript file can be done directly from the command-line as shown in the examples below.

Example

To render scene c:\path\to\scenetorender.max on GPU device 1 only

```
echo rsSetCudaDevices #(1) > rsSetCudaDevices.ms3dsmaxcmd.exe c:\path\to\scenetorender.ma –preRenderScript:
rsSetCudaDevices.ms
```

Same scene rendered using GPU device 0 and 1:

```
echo rsSetCudaDevices #(0,1) > rsSetCudaDevices.ms3dsmaxcmd.exe c:\path\to\scenetorender.ma –preRenderScript:
rsSetCudaDevices.ms
```

Prerequisites

When running xsibatch, you should run from the XSI command prompt. You can do this by choosing the "Softimage Command Prompt" shortcut from the start menu. Alternatively, you can open a regular command prompt and "initialize" the required XSI environment variables by executing the following command:

```
call "C:\Program Files\Autodesk\Softimage 2015 SP2\Application\bin\setenv.bat"
```

Substitute the path above based on your Softimage version.

Syntax

```
xsibatch.exe -render <scene_file> -script C:\ProgramData\Redshift\Logic\RedshiftSelectCudaDevices.py -main
RedshiftSelectCudaDevices -args -deviceIds <device_id_array>
```

Where <device_id_array> is a list of the GPU device ids you wish to render with. The syntax for this array is 'python style' - for example use [0,2] to render on devices 0 and 2.

Example

To render scene c:\path\to\scenetorender.scn on GPU device 1 only:

```
xsibatch.exe -render c:\path\to\scenetorender.scn -script C:
\ProgramData\Redshift\Logic\RedshiftSelectCudaDevices.py -main RedshiftSelectCudaDevices -args -deviceIds [1]
```

Same scene rendered using GPU device 0 and 1:

```
xsibatch.exe -render c:\path\to\scenetorender.scn -script C:
\ProgramData\Redshift\Logic\RedshiftSelectCudaDevices.py -main RedshiftSelectCudaDevices -args -deviceIds [0,1]
```

Prerequisites

To use the Houdini command-line tools, you need to open the Houdini 'Command Line Tools' application (available as a shortcut from the start menu) in Windows.

In Linux, you'll need to source the Houdini environment as follows:

```
source houdini_setup
```

Syntax / Example

For example, using 'hbatch', the most basic command-line render session without scripts to load a scene, set the enabled GPUs and render the Redshift ROP node, can be:

```
> hbatch mySceneFile.hip
> Redshift_setGPU -s 011
> render /out/myRedshiftROP
```

Please note that the -s 001 parameter after the Redshift_setGPU command means that, on a system with 3 GPUs, the first GPU should be disabled ('0') while the second and third GPU should be enabled ('1').

Alternatively, you can write a script file like

```
mread mySceneFile.hip
Redshift_setGPU -s 011
render /out/myRedshiftROP
```

And execute it using:

```
> hbatch myScriptFile
```

Prerequisites

To render from the command-line you can use the CommandLine tool located in you Cinema 4D application installation folder. Alternatively the Cinema 4D application itself can also be used.

Syntax

```
 Commandline.exe -redshift-gpu <device_id> -render c:\path\to\scene.c4d
```

Where <device_id> is the GPU device id you wish to render with.

Example

To render a scene located at c:\path\to\scene.c4d using only GPU device 1:

```
Commandline.exe -redshift-gpu 1 -render c:\path\to\scene.c4d
```

To render the same scene rendered using both GPU device 0 and 1:

```
Commandline.exe -redshift-gpu 0 -redshift-gpu 1 -render c:\path\to\scene.c4d
```

The CommandLine tool supports multiple parameters for customizing rendering (frame ranges, image output options etc.) For more details please consult the "Cinema 4D and the Command Line" section of the Cinema 4D documentation.

How do I determine my GPU device ids?

There are a number of ways to determine the device ids associated with each of your GPUs.

One option is to open prefences.xml from C:\ProgramData\Redshift in a text editor and inspect the value of "AllCudaDevices". For example:

```
<preference name="AllComputeDevices" type="string" value="0:Quadro K5000,1:Quadro 6000," />
```

In this case the Quadro K5000 is device 0, while the Quadro 6000 is device 1.