Rendering NSI file

`renderdl` reads a file containing scene description commands and “executes” them to produce an image. **`renderdl`** also has some auxiliary usage as explained in the parameters description below.

In the simplest form, to render a file named `file.nsi`, just type:

```
renderdl file.nsi
```

Command Line Options

Option	Description
-display	Display the images in 3Delight Display while rendering instead of writing them to file (as specified in the NSI file).
-cloud	Render the file using 3Delight Cloud instead of locally. Typically (depending on cloud ressources available), rendering will occur using around 96 cores per image for images up to 2K in resolution and proportionally more for higher resolutions.
-cloudspeed n	Boost the number of cores used when rendering by a factor of "n". The default value is "1" when rendering multiple images to file or "5" in the case when the -display option is used and there is only one image to render. The maximum value is 30. When rendering multiple images we recommend using the default value. This is because additional cores are used more efficiently by rendering several images simultaneously. How many depends on your internet speed and your Spending Rate Limit (for *Batch Renders*). You can read more about this in Cloud Rendering Speed. Note: When rendering a 4K resolution image, approximately 280 cores will be used at the "cloudspeed" of "1". Proportionally more or less cores for higher or lower resolution. Refer to the table in this page for more details on how many cores are used at various resolutions for “cloudspeed" of 1, 3, 9 and 27. **Faster cloud speed are more expensive due to efficiency degradation when using a large number of cores on a single image. This is especially true when the initialisation phase (time to first pixel) takes a fair proportion of the render time.**
-collective name	Render the file using the network collective "name" instead of locally. For this option work, you must have pre-defined your collectives as explained in 3Delight Collective.
-lowpriority	Launch the render using a lower system priority. This is useful when using the same machine for batch renders and 3Delight Collective renders.
-id	Launch 3Delight Display to display the image while rendering instead of writing images to files (as specified in the NSI file). (Obsolete option. It is replaced with the -display option.)
-t n	Controls the number of cores/threads to use when rendering. If "n" is a whole number it implies: n > 0 : Use exactly "n" threads. n < 0 : Use all but "n" threads. If "n" is a fraction between 0 and 1, then a fraction of the available cores will be used (eg. -t 0.5 specifies to use half of the available cores. By default, **renderdl** use as many threads as there are available cores (including virtual/logical cores). This option is ignored when the option -cloud or -collective is used.
`-stats`	Embed statistics in rendered images. This is supported for EXR and TIFF files only. Statistics are explained in more detail in Detailed Statistics.
`-progress`	Prints a progress status after each rendered bucket.

`-cat`	Print the NSI commands in human readable ascii format instead of processing them for rendering. This can be used to convert a binary (or compressed) NSI file into an human readable ASCII file: `renderdl -cat binary.nsi.gz > ascii.nsi`
`-cat -binary`	Outputs the NSI commands in binary format. For example: `renderdl -cat -binary ascii.nsi > binary.nsi`
`-cat -gzip`	Outputs the NSI file in compressed form. For example: `renderdl -cat -gzip ascii.nsi > ascii.nsi.gz`
`-cat -callprocedurals`	expand all procedurals and archives. This is very useful when packaging an NSI file.
`-cat -o filename`	output NSI stream to filename instead of stdout.
`-lua`	Interpret input file as a LUA file.
`-v`	Prints the current version of the renderer.
-h	Prints the help.

No File Name Specified

If no file name is specified, **`renderdl`** reads scene description commands from the standard in. This feature enables piping commands directly in **`renderd`**. For example, to enter scene description commands interactively (which is not really practical), do the following:

```
renderdl
Reading (stdin)
<enter commands here>
```

If you wish to pipe the content of `file.nsi` in **`renderdl`**, type:

```
cat file.nsi | renderdl
```

Shell Return values

The **`renderdl`** executable will return one of the following values:

Return Value	Description
0	No error.
1	Bad combination of parameters. An error message will explain why.
199	Option "licensing" "waitforlicense" 0 was used and no license was available.
255	The NSI file specified on the command line could not be read.