SMEDGE

What's New in Smedge

Smedge 2018

Table of Contents

Introduction

Welcome to Smedge 2018!

The latest version of Smedge provides a whole new level of performance and reliability, and gives you more control over your rendering workflow than ever before. Production tested on networks over 1,500 nodes, scaling is no problem.

We recommend that you back up any old data before upgrading. While the data can be upgraded automatically, if you ever want to go back to the old version of Smedge, the upgraded data will not be successfully read by the old versions.

As always, if you have any questions about new features or our future development plans, we encourage you to contact us.

Thanks, and see you in Smedge!

New Product to control Mistika VR

Developed, tested and shared with the Smedge community by the Mistika developers.

Per-Minute Licensing

Smedge now includes both the legacy licensing and the ability to license render nodes on a per-minute basis. Both systems can be used at the same time on the same network.

The licensing model works differently now: any engine that is enabled is capable of working, and when the master finds work to send it, the master will first try to check out a license for that machine. By default, engines will try to get a legacy license first, and will try to get a point license if there are no legacy licenses available. You can configure engines to only try for a legacy or point license. If you disable a machine it will not be assigned work and will never consume a license.

Your legacy licenses will transfer just fine and work similar to before, with the addition that should you ever install a temporary license on top of your permanent license, the permanent license will automatically revert to being used when the temporary code is expired. When an engine is finished working, the master will release its legacy license immediately.

Point licenses work by licensing an engine for 1 minute at a time. When work is assigned to an engine and it gets a point license, that engine is then licensed for a full minute no matter what. If the work finishes within the minute, the engine will already be licensed when the next dispatch iteration comes, and will get to keep working. If the engine is working when the point expires, the master will allocate another point to the engine. As long as you have points available, your engines will keep being licensed and keep running work. When you run out of points, any currently running work will finish normally.

Points are purchased through the Uberware.net website or through your reseller. See http://www.uberware.net for current pricing.

List, Restore, and Remove Deleted (archived) jobs

Smedge 2016 added the ability to archive your deleted jobs, but did not give any easy access to restoring these jobs. This version adds several new ways to access these archived jobs, and restore or remove them.

The GUI now has the Archived Job Manager, available in the Job menu in Administrator Mode. The manager will open in a new frame window, and will request the list of jobs from the master. You can see when it starts and when it finishes receiving the list, which can take some time to download if you have a lot of deleted jobs. Using the list, you can select jobs to restore or remove. Restoring itself can take a little time, as the archives are compressed, and restoring them is handled sequentially by the master to avoid getting overwhelmed if you restore a bunch of jobs at one time.

The Job command line shell also has the ability to list, restore and remove archives. The underlying library API access added a few extra messages to handle requesting the list, notifying about the list start, list items, and list finish, as well as requests to restore and remove archives, available to any library level access to the API (compiled plug-ins, custom shells, etc...)

Improved View Filtering in the GUI

The new view filtering system is simpler to use and takes up less screen space, but gives you more control and more options for how you filter what you see. Instead of a bunch of separate controls, there is just a single text field where you can type a search filter string to find what you are looking for.

Start typing to search by name. Each word you type must be found in the job data for the job to pass through the filter. The following filter out any items that did not have “sc10” and “v08” somewhere in their names:

```
sc10 v08
```

To search in parameters other than the name fields, type the name of the parameter you want to search with a colon after it. You can supply more than one parameter to search, and all of them must be found. For example, to find work from creator “Robin” with a scene that includes “Hero” and “0010”:

```
Creator: Robin Scene: Hero 0010
```

You can use <, <=, >, >= and similar to filter out values less than or greater than a value. For example to see only items with a priority above 75:

```
Priority > 75
```

Note that white-space is ignored by default. You can use double quotes and/or the backslash to escape special characters you want to search for, including spaces, backslashes, or the <, >, :, or = characters.

Improved Success Message test system

The Success Message test has been improved to allow more flexible searching using the new **SuccessMessageType** parameter. The values you can use determine how the **SuccessMessage** parameter is searched for:

Match	The line matches the **SuccessMessage** value exactly (Smedge 2016 and earlier behavior)
Start	The output line starts with the **SuccessMessage** value
End	The output line ends with the **SuccessMessage** value
Contains	The output line contains the **SuccessMessage** value (the default if not specified)
Regex	The matches the regular expression in the **SuccessMessage** value

There is also a new option to force a timeout after detecting a success message. The **SuccessMessageTimeout** parameter specifies a number of seconds. Once the engine has detected the success message, after this timeout has elapsed, if the process is still running, it will be terminated, however, the engine will still consider the work as successful.

This system is useful for detecting hangs and issues during a shutdown or cleanup, but please be aware that terminating processes can lead to bad results if there is a delay after the success message has been sent by the renderer and when it has truly finished its work. If you set the timeout value to a blank string or a negative number, no timeout processing is done.

New Start-up Message error detection system

Using a similar set of parameters and options as the Success Message system, this system allows sophisticated control over detecting successful start-up of a render process within a reasonable amount of time. The parameters are **StartupMessage**, **StartupMessageType**, and **StartupMessageTimeout**, which have the same syntax and general operation as with the Success Message parameters. The difference is that if the Start-up Message timeout elapses without seeing the start-up message, the work is requeued as a failed work.

Access to Environment Variables in variable substitution system

A new common parameter **Environment** is added. Use this parameter with a colon and then the name of the environment variable you want, and the value will be replaced with the current system value of that environment variable. As a shortcut, you can leave off the word Environment, and simply start a parameter as just a colon. These two examples get the value of the HOME environment variable:

```
$(Environment:HOME)
$(:HOME)
```

New parameter command to find and replace text

There are three new parameter commands to find and replace characters in a parameter value as part of the Smedge variable substitution system. The commands are Replace, ReplaceAny, and ReplaceRegex. They all use the same syntax:

```
$(Name.Replace:abc|xyz)
```

This parameter syntax searches for all occurrences of the characters “abc” in the Job Name, and replaces them with the characters “xyz”. ReplaceAny uses the concept of replacing any of the characters “a” “b” or “c” and replaces then with the string “xyz”. ReplaceRegex assumes that “abc” is a regular expression, each match is replace with the string “xyz”. You may need to escape special characters like quotes and backslashes.

After Effects Module Improvements

Anyone that has used After Effects for any time on a production render farm knows that it can be temperamental. To help work around some of the common issues, there have been several changes:

- AE work units use the output frame files to look for the last expected frame as a “success” message. This is used both for verifying that all expected frames have rendered and that the process does not get hung in its clean-up process using the new success timeout feature.

- The engine now also uses the new startup message system to wait for an expected message that AE creates when starting a render. This can detect when AE gets hung from issues with movie files or plug-ins at startup time. The default timeout is 120 seconds, adjustable as both an engine product option and a job advanced parameter

- The After Effects submit script has also been changed to work more efficiently. The interface and operation has not changed from a user perspective, but behind the scenes it now creates a single .SJ file with all of the queued comps and submits them with a single call to the master, making it much faster to submit large numbers of jobs at one time.

- Added the -mp flag to the Render Overrides tab

Lightwave Module has SequenceBy parameter

This was actually a bug fix, as the parameter was supposed to be there but there was a typo causing it to look for the wrong name in the module definition file. This is fixed, adding this functionality to the Lightwave product.

Option to pack engines with work during dispatch loop

In Smedge 2016 and earlier, the master always “packed” each engine with work before moving on to the next. For example, if the engine has 32 cores, and a job called for 8 cores per worker and had plenty of work available, the master would find 4 work units for this engine before moving on to the next engine during the dispatch loop. This tended to cluster work based on engine priority, but could be less efficient for certain types of work loads.

This feature is now disabled by default, and instead, once the master dispatches a work unit to an engine during the dispatch loop, it will then move on to the next engine immediately. This dispatches work in a more “round-robin” style to all available nodes, even if the work does not consume all available resources on the node. If the engine still has space available for more work, that will be handled on the next dispatch iteration. You can re-enable the old style of operation if you prefer it by setting the master option **PackEngine**. You can set this in the Configure Master dialog box, on the Distribution tab.

Double Click Work to view output

Double clicking an item in any work list will try to use the View Process Output command for that item. If it is available, an output window will open with the process output just as if you used the menu command or toolbar button.

Improved Edit Controls

The basic text editing widget in the Smedge GUI libraries has been extended to include basic auto-complete functionality and to have a “clear” button to quickly reset the field to blank. The auto-complete helps you find parameter names, commands, product names, IDs, environment variables and other common values that Smedge understands. The clear button appears when you move the mouse to the far left side of the control.

Submit Pools by Name

Before, the Master required that you submitted the job with a pool ID only. The command line shells previously tried to work around this, when possible, but now they don't have to. Instead, you can submit a job using the Pool's name anywhere you would have used the ID only before, including in the .SJ job files. The Master will try to resolve the pool ID from the name you supply. If multiple pools have the same name, the first one found will be used. If no pool is found with that name, the pool will be set to the default Whole System pool ID.

Job List command has special syntax to list all possible parameters from the results

When you use the List command with the Job command line tool, you can optionally provide the names of the parameters you want to list. If you supply a list of parameter names, only these names are listed. If you do not supply any parameter names, a default set of parameters would be shown.

Because it is possible that jobs can have different parameters, in order to allow you to see all possible parameters from all of the jobs you requested, you can now supply the special parameter name: ***** (a single asterisk character).

wxConfirmDlg can change the button labels

If you use the library API, the Smedge customized confirmation dialog box system now allows you to change the strings on the buttons to make the choice of actions more logical for the user based on the question being asked.

Issues Resolved

Improved JSON formatting and handling

Improved handling and display of JSON formatted engine data by the Engine command line shell

Permanently Finished Work could be requeued from event failure

If a work had been set to a status that should have indicated permanent failure (usually one of: Complete, Canceled, Failed, Orphaned, or AdminCancel), that work could get re-queued if the work post-process event failed.

Pool Selector sorts pools but not engines

It sorted the pools first, then added the engines sorted by their IDs, which is not useful. Now it correctly sorts them by their names.

Switching history view grouping sometimes did not clean display

It correctly refreshes now when you switch modes.

List columns are not saving the ascending/descending option correctly

Now it remembers your choice correctly

Some dialogs have initial sizes that are awfully small

Some better basic window layout and minimum sizes will make things a little easier. Areas where controls could get hidden by small windows now include scrolling areas so you can still access controls in smaller spaces.

Some control fields would show the default value

There was a bug in the GUI that would sometimes cause certain data values to show as and possibly get reset to default values.

(Mac) File browser dialog box appears behind the job dialog where you request it

Mac dialog boxes sometimes had issues with Z-order.

(Mac) Browse dialog can sometimes pop up twice in a row

Pressing the button would cause the browser to show up twice

(Windows) SmedgeEngine was still setting its affinity after change

Leftover code from the 32 bit version was limiting processor affinity on some hardware

Conspectus does not create scrollbar correctly at startup

The window was not being rearranged correctly when an engine connected, resulting in display without the scrollbar until you sized the window.

ParameterInfo::DirList and FileList are not being translated

These are now translated automatically at the same time as the File and Dir parameters types are.

Button control issues

The button control had some display and functionality issues that have been resolved, for better display and use.