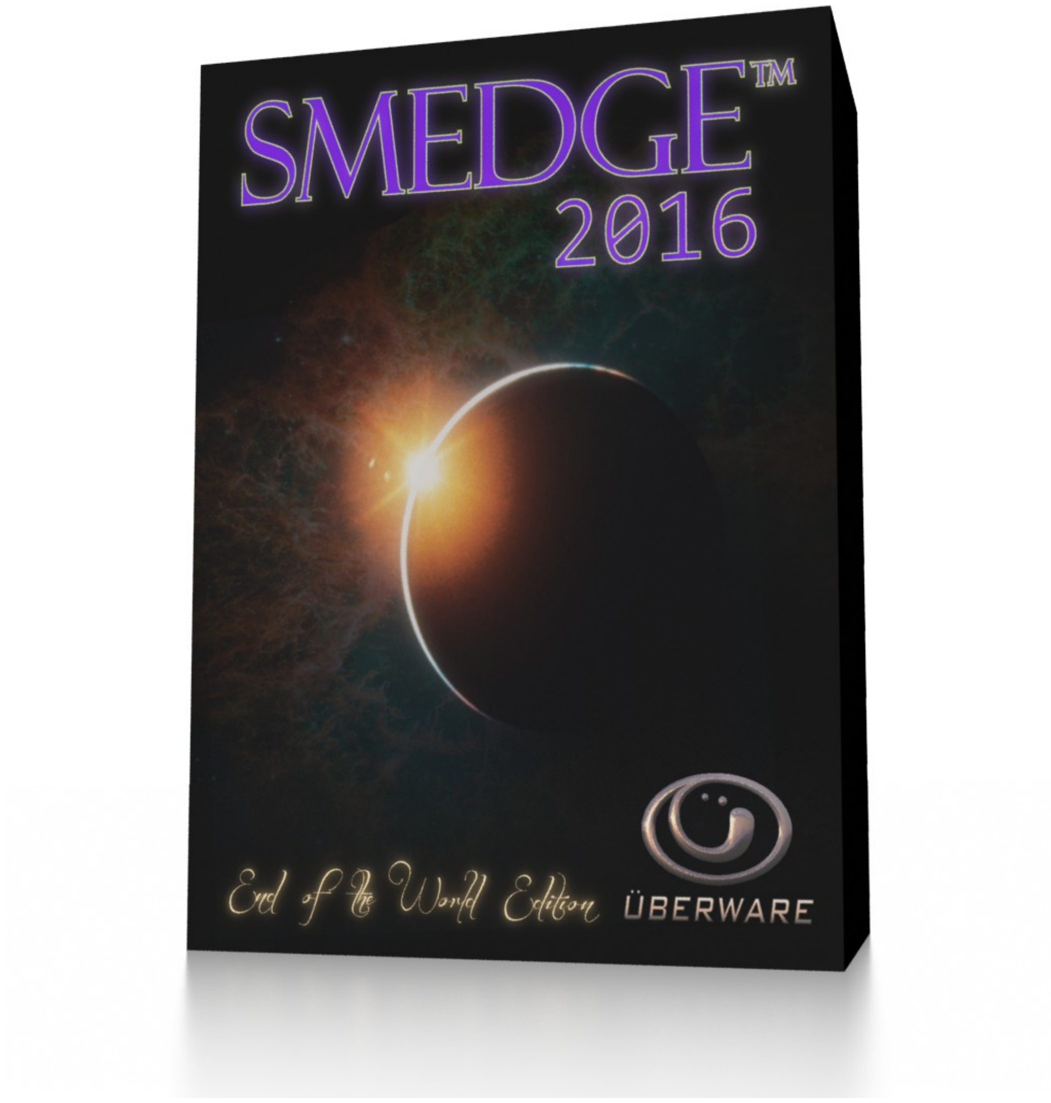
SMEDGE

What's New in Smedge

Smedge 2016

Table of Contents

Introduction



Welcome to Smedge 2016!

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 up to 2,000 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!

GPU distribution

Smedge can now track the GPU devices on a node, and can use that information to distribute work to nodes based on GPU resources. Currently, this is implemented for the RedShift for Maya product, but it can be added to any Maya based product or virtual module file, if desired. To add it to a Maya product, add a line To add it to a Virtual Module, you would add a GPUs field to the product. Set the Job's GPUs value to 0 to have the job use all available GPUs, or a positive integer to use that number of GPU devices for each worker (e.g. GPUS=1 on a machine with 2 devices will allow 2 workers, one on each device).

The engine data now includes a field for defining a list of GPU devices available on the machine by an index number. These devices then can be used analogously to CPUs in Smedge, with work assigned to one, some, or all of the GPU devices. This setting is left blank by default but can be set to a list of integers that identify the GPU devices available to the renderer.

Currently, the system requires manual configuration of the GPU device indices in the Configure Engine Settings dialog. You can choose to include some or all of the devices on your system, which can help if you need to ensure that at least one device is left idle to support use of the machine console attached to an actual monitor.

RedShift for Maya

The first product to make use of the GPU distribution system is the new product to support the RedShift GPU renderer plugin to Maya. This product is implemented in the default Maya.ini file distributed with Smedge in the Modules folder. Note that you can use this product definition file with older versions of Smedge, if you want, but you will not be able to use the GPU distribution feature unless you are using Smedge 2016 or later.

Lightwave Module is now a Virtual Module

The compiled module was not being well maintained and lacked features available to the other products. Switching to a virtual module gave up a few arcane features for the greater overall functionality of a Virtual Module. It also makes it simper to customize and update for end users, who can now take advantage of new features and customize the module to their preferences.

New engine command line control options including JSON formatting

The Engine command line shell has a new option for formatting how the engine data is reported back to the user when you use the List command. By default, the listing will include all of the engine data, but if you supply the -Raw flag, it will only list items in each engine that override the defaults without listing the defaults as key=value pairs, and if you use the -JSON flag, it lists that same data in JSON format.

Additionally, you can now configure engines and the engine default data using JSON format. Using the Set command, you can supply a JSON structure object with settings to apply to the specified engines or to use as the default engine values. See the Engine shell command line help for more details.

Override the Machine folder and User folder

You can now set environment variables to override the base folder used for all settings, both for the “machine” folder settings and the “user” folder settings. The variables are: SMEDGE_MACHINE_FOLDER_BASE and SMEDGE_USER_FOLDER_BASE. The variable points to the base folder, where each app creates its own folder internally, so you only have to set these one time on a machine and all processes that inherit that environment will read and write files from the correct subfolders.

New Engine command line listing of licensed engines

Track down your licensing more easily with scripts using the new -ReportLicenses command line option.

New interface color options

Because everyone's displays and everyone's tastes are different, we have added two new color options for the interface. Light and Dark are the same, and Lighter and Darker give more contrast. There is also now a Custom option, allowing you to fully specify the colors you want for each interface element type. Currently, these colors are specified manually in the Colors.ini file where the color scheme is saved.

Other improvements to the UI include the tooltips in the submit window, which now allow you to see the help information and internal name for parameters in the window, and fixes to the ability to to change colors even with multiple components open at the same time.

Comparison parameter commands

You can now do basic comparisons of equal to, not equal to, less than, greater than, less or equal, and greater or equal. These are implemented as new parameter commands with symbols based on the C code style symbols:

=	equal to
!=	not equal to (you can also use <>)
<	less than
>	greater than
<=	less or equal
>=	greater or equal

The result of the parameter command will be a 1 if the prior value passes the comparison, and 0 if not. You can combine this with the conditional parameter command (the question mark: ?) to build conditional parameters like this:

```
$(Status.!=:7.?:|$(Scene.CutExtension)_$(SubRange.Start.+:1.Pad).$(Scene.Extension))
```

This will create a file name from the scene name and sub range, but only if the job's status value is equal to 7. See the Administrator Manual section on parameter commands to see a breakdown of how the above parameter would be expanded at run-time when it is used in a command field, like an event handler.

Master command line option to disable automatic location system

Add the flag `-NoUDP` to the SmedgeMaster command line to disable the automatic master location system, which uses UDP broadcasting to try to find the master automatically. When this system is disabled, you will have to manually point your clients to where the master is. You can do this either with a Connection.ini file, options, or command line flags to the client process.

Set local client port in the Connection.ini

You can now override the local port to use for the client in a Connection.ini file without having to supply the interface address.

Herald can now wait for a process to complete

A new option on the execute process event notification allows you to tell herald to wait for the process to finish before moving on to the next notification. This can be used to ensure only a single process command is run at a time, in case you must not run the same process in parallel for multiple events or notifications at the same time. Use this with caution, as it can cause Herald to hang, if the process spawned never completes.

CheckFileSequence shows stats of total, missing and too-small files

The count of total, missing and too-small files is now listed at the bottom of the window. It is updated at each count. Display and performance will be improved in future releases.

SequenceDistributor is smarter about padding

When you use the automatic name padding, it now checks the highest frame number it should see and bases the required padding on the size of that number. This should simplify working with sequences where numbers are in the 10s or 100s of thousands or more.

New Sequence parameters RangeCount, CustomRangeCount and UserRangeCount

These return the total number of items in the $(Range) and/or $(CustomRange) sets. For most Products in Smedge, this is the frame count. This will work even if the frame count is complex, out of order or overlapping. RangeCount gets the count from Range, CustomRangeCount gets the count from the CustomRange, and UserRangeCount gets the count from the custom range, if there is one, or the Range if there is no custom range.

Master sets the work note to the ID for identification

By default, the Master will set the work unit note to the work ID when the work unit is created. This can be useful for identifying the work by its ID in the graphical interface, if you need to drill into log files or data files on disk that reference the work ID. You can disable this feature with an environment variable: SMEDGE_NO_WORK_ID_NOTE. Set this value to a non-zero integer to revert to the old behavior, where the work note is a copy of the parent Job note.

Performance Improvements in GUI

A bottleneck in the history control is resolved that improves GUI performance significantly, especially when you are monitoring a job with a high volume of history. Also, all platforms are updated to the most recent updated supporting libraries, which improves performance and resolves some performance and display issues in various platforms that resulted from issues in the underlying libraries. Work is continuing to streamline and scale display and update performance.

Internal API optimization and improvements

Most of the work for Smedge 2016 is in optimizing the system for performance and scaling. Smedge 2016 is now running networks over 2,000 nodes, rendering millions of frames every month, with performance and reliability that is unparalleled.

In Smedge 2016, both the Job and the Engine data share a common, version agnostic data format that can easily be converted to or from many industry standard "object" formats, like JSON. These features will be expanded upon more in future development, simplifying, modernizing and standardizing Smedge for integration into even more sophisticated pipeline environments.

Engine -AllowMultiple switch has new purpose

The switch now tells the engine to start up as a pure test-mode multiple instance "virtual machine". Note that this is a live instance of the production engine running, with a simulation of 64 cores and a unique name. You can supply a name for the engine as parameters to the -AllowMultiple switch. For example, -AllowMultiple Test-Machine would start a new engine component with a unique ID and the name "Test-Machine". These are useful for simulating larger machines or networks, to test and debug custom modules and components.

Licenses could get lost over time requiring refreshing the master to get it back

On occasion, Engines could lose connectivity without giving up their license, which could cause starvation of licenses. The underlying cause of the issue has been resolved and the licensing system has been improved to be more resilient to future issues that may arise.

Smedge on Mac could hang starting child processes

A race condition could cause a thread deadlock starting a child process on Macs. It generally did not affect the parent process, but could result in stale hung instances of the parent process from failed forks left resident on the system, and some operations could fail in strange ways. The cause of the deadlock is fixed.

Some usage was not resetting the Engine Mode timeout

The Engine Mode timer was originally based on input via menus or selecting items. It is now extended to other possible usages where the mode may have kicked even though the user was operating the GUI. Modal dialogs will block engine mode, so it cannot happen while one is open. Submitting jobs and engine settings from open modeless windows also resets the time. These and similar other changes should help avoid the GUI going to Engine Mode while being used.

History Control horizontal scrolling is wrong

The control was not updating itself correctly as it loaded the column widths at startup. This difference could lead to an annoying lack of scroll bar. It now correctly updates and scrolls.

JobHistory incorrectly reset finished work count

The history would throw away the count when copying from another object. This manifested in weird ways, notably by showing 0 packets done for finished jobs. The underlying issue is resolved, fixing this display (which is now also modified to show the total range count from above).

Event command errors did not always set the job's LastError value

The error would get reported in the note, but not show up if you queried the $(LastError) parameter. This has been corrected so it should be easier to find errors of this type.

Master did not always reset the job accumulated real and engine time counts

If a job had completed some work when it was copied to make a new job, that new job would start out with the same amount of elapsed real and engine time that the original job had, instead of starting at 0. This could make statistics inaccurate. These counters are now properly reset when a new job is created, even if that job was copied from another job that had already accumulated some elapsed time.

New GUI buttons were not showing enabled/disabled state

Several attempts to resolve this had not completely eradicated the issue on all platforms. Updating the underlying GUI library to be consistent across all platforms has allowed what should be (hopefully) a final resolution to this.

Failure to start the info server at engine startup no longer crashes the Engine

If the machine information server failed to start correctly, the Engine component would crash. Now the engine works normally even if this system fails to start, and it keeps trying to start the system, in case the cause of the failure is temporary.

Submit command line tool could hang without ever submitting the job

A rare thread race condition could cause the process to think it had sent the job data before it connected to the master, leaving it hanging around doing nothing. This is resolved so that it is now impossible for that to happen, and Submit is smarter about retrying to submit if things go wrong.

Processor affinity setting could get set unintentionally

If the environment variable to override processor affinity was not supplied, it was possible when checking for it to accidentally set up a value that would get an invalid affinity setting, causing erratic behavior. This unusual situation is now corrected.

The bar history control clipped values to 16 bit signed max limit

If a machine had more than 64GB of RAM, the graph and tool tip would show incorrect values. Now it correctly reads the value and shows consistently on all platforms.

Conspectus was not respecting the color setting

Now it does.

Some windows did not restore their position correctly

Inconsistencies in the wxWidgets GUI library changed how the window options were loaded, causing some windows to no longer restore themselves to their last position. This should be corrected with all components on all platforms now.