

Software Project Summary

Sherman Studio Toolbox

Company: Free Software Foundation
Language: Python/SQL
Contribution: Original author
URL: <http://sourceforge.net/projects/shermanstudio/>

Frustrated by nonstandard reinventions-of-the-wheel at various studios, I've been developing a set of useful scripts and utilities for visual effects and CG film production called the Sherman Studio Toolbox. The largest of these tools is a fully functional renderfarm with advanced prioritization, resource allocation, and reporting features. The project is a work in progress - studios are encouraged to improve the codebase and share their improvements.

Skycap

Company: ESC
Language: Python
Contribution: Original author

Skycap is a renderfarm script submission system designed to address the shortcomings in the existing scripting system. Skycap introduced a new jobfile format to avoid submission errors, allowed queueing of dependent and independent tasks within a job, supported chunking of frames to reduce farm allocation overhead, and allowed scripts to be written in any scripting language instead of only Windows .bat.

Gluegrep

Company: ESC
Language: Python
Contribution: Original author

Mental Ray files are assembled from with a program called "Gluer", which uses file specifying the MI file components to use. Gluegrep allows a user to search all files referenced by the gluer template files - handy for discovering which component is including the wrong material or referencing missing texture files.

MiCombine

Company: ESC
Language: Python
Contribution: Original author

An improvement on (or alternative to) the Gluer, eliminates the need for a .gt file. miCombine combines several .mi files and concatenates the elements to preserve dependency ordering. As a side effect, miCombine also garbage collects all irrelevant and duplicate elements. MiCombine does not keep the content of its .mi files in memory: instead it keeps file name and line numbers delineating MI element boundaries and assembles the final MI file by reading only from the parts of the files it needs. miCombine is modular, written to be used as either a library or standalone program, and its functions are used as modules in other programs.

Lightjack

Company: ESC
Language: Python
Contribution: Original author

A stopgap hack to allow TDs to position lights in scenes with low-res models and see their lighting system applied to high-res models within ESC's integrated Maya/Mental Ray "Rayjack" system. Lightjack uses MiCombine as a module to extract the lights from and adds them to the .mi file specified on the command line, and passes the new file to Mental Ray for rendering.

MiEdit

Company: ESC
Language: Python
Contribution: Original author

MiCombine's other half. Reads .mi files from stdin (miCombine), writes to stdout (Mental Ray). Allows in-stream editing of .mi files without additional files being saved anywhere. Objects can be deleted, phenomena reassigned, and shader parameters edited, etc.

Jeb

Company: Tippett Studio
Language: Python/Tkinter
Contribution: Original author

The Jeb GUI enables users to build and edit job render scripts. Separate show-specific, user-specific and department-specific script libraries allowed Jeb to be adopted as the standard renderfarm interface by TD, Matchmove, Compositor, and FX animation departments.

Ez2Shake

Company: Tippett Studio
Language: Python/Tkinter
Contribution: Original author

Tippett Studio's gradual transition from Irix to Linux left behind several key programs including the IRIX-only "EzComp" compositing system. Ez2Shake provided a workaround by translating EzComp scripts into Shake scripts which could be rendered on Linux. Ez2Shake proved so effective that the Linux EzComp render engine was never necessary.

LightMix

Company: Tippett Studio
Language: Perl/Tk, Python, Renderman SL
Contribution: Maintenance, Linux porting, new features

LightMix allows lighting TDs to render individual light contributions in a scene and change the color and intensity of the contributions in 2D. My responsibility included maintenance of both the rendering side (special shaders for simultaneous multichannel rendering, scripts to substitute shaders and add parameters in a RIB, scripts to render the passes) and the interactive side (programs to change brightness and hue of the elements and combine and display them).