open shading language

TAC Annual Project Review

October 29, 2025

#ASWF

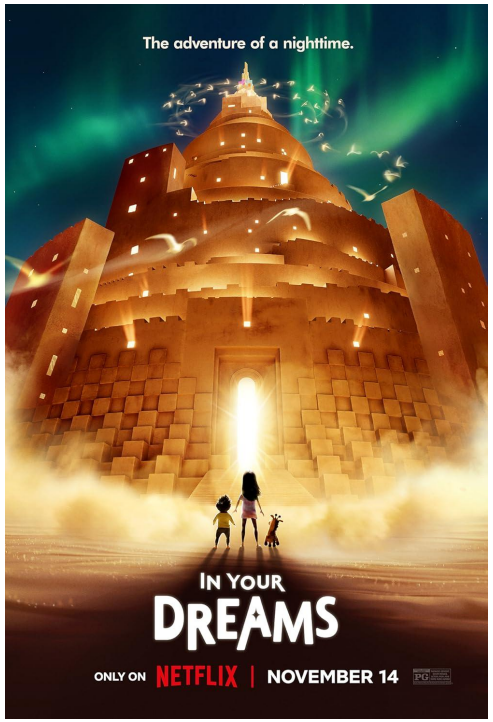
What is Open Shading Language?

- Shading language for modern production renderers
- A compiler and efficient runtime for the language
- Leverages LLVM for execution on CPU and GPU
- Project just turned 17 years old (first commit Sept 3, 2008)
- ASWF member since 2020

What is Open Shading Language?

- Software supporting OSL:
  - 3ds Max, Arnold, Blender/Cycles, 3delight, Renderman, V-Ray, Octane, Redshift, …
- Studio renderers:
  - Animal Logic's Glimpse, Sony Imageworks' SPEAR/Arnold, Illumination Labs
- Ties to other ASWF projects:
  - Dependencies: OpenImageIO, OpenColorIO, OpenEXR, OpenVDB
  - Used by: MaterialX, OpenColorIO
  - Collaborate with: OpenImageIO, MaterialX, OpenColorIO
- At least 200 films (that we know of… please tell us!)

Some recent films using OSL



What is different about OSL?

- Shaders can be assembled into networks
  - Networks are evaluated lazily (unused sections are not evaluated)
- Entire graph is runtime optimized
  - Works across nodes in the shading graph
  - Scales to very large production shaders that have lots of features
  - Optimizer can remove 98% of instructions in such cases
- Automatic Differentiation for filtering and bump mapping
  - More accurate and more efficient than finite differences
- Shaders return *closures* instead of colors
  - Renderer decides how to integrate the lighting
  - Complex layering can be achieved by building closure trees

OSL Technical Steering Committee

- Current members
  - Chris Kulla, Epic Games (TSC Chair)
  - Larry Gritz, Sony Pictures Imageworks (Chief Architect)
  - Adrien Herubel, Autodesk
  - Alex Wells, Intel
  - Brecht Van Lommel, Blender
  - Clint Hanson, DNEG
  - Eric Enderton, NVIDIA
  - Lee Kerley, Apple
  - Luke Emrose, Netflix Animation
  - Mitch Prater, Laika
  - Stephen Friedman, Pixar

Code contributors in the last year

- Aidan Welch
- Alejandro Conty (SPI)
- Alex Fuller
- Alex Wells (Intel)
- Alexey Smolenchuk (DNEG)
- Brecht Van Lommel (Blender)
- Chris Hellmuth (SPI)
- Christopher Kulla (Epic)
- Cliff Stein (SPI)
- Curtis Black (Netflix)
- Jean-Francois Panisset (CIWG)
- John Haddon (Image Engine)
- Jonathan Stone (Lucasfilm)
- Larry Gritz (SPI)
- Lukas Stockner
- Mitch Prater (LAIKA)
- Sergey Sharybin (Blender)
- Sparsh Nair
- Sven-Hendrik Haase
- Tim Grant (NVIDIA)

What's new in OSL 1.14 this year?

Released OSL v1.14 in April, 2025 (1.13 was Feb 2024)

- New dependency & toolchain minimums:
  - gcc 6 -> 9, C++ 14 -> 17, Python 2.7 -> 3.7, CMake 3.19
  - Imath 2.4 -> 3.1, OIIO 2.5, LLVM 11
- Eliminated Boost dependency
- Support for LLVM 18 (also 19, 20, 21 in patches)
- Support for OIIO 3.0 (also 3.1 in patches)
- Support for gcc14 & C++20 (as of 1.14.6.0)

- Documentation has moved to RTD via https://docs.openshadinglanguage.org/
- OptiX now fully tested in CI (as of 1.14.6.0)

What's new in OSL 1.14 this year?

- Texture calls accept "colorspace" optional keyword/argument
- testrender feature improvements:
  - triangle meshes, displacement, full OptiX support
- Renderer interface changes:
  - ustringhash nearly everywhere (ustring hard on GPUs)
  - API for building getattribute getter "free functions"
  - Extend free functions to: texturing, point clouds, trace
- Facilities for caching generated PTX
- Many improvements to OptiX and SIMD batch back ends

OSL 1.15 Highlights – Coming soon!

- Raise some dependency minimums:
  - Python 3.7 -> 3.9, LLVM 11 -> 14
  - maybe OIIO 2.5 -> 3.0? maybe raise CMake?
- BSDL library
  - SPI open sourcing their production BSDFs
- More improvements to free functions
  - rs_allocate_closure
- Aiming for full CI testing on Windows

Future roadmap grab bag

- Full GPU texture parity with OIIO TextureSystem
- Dependency auto-build capabilities (similarly to OpenImageIO)
- Continued transition from RendererServices to “free functions” provided as LLVM bitcode
- Continued GPU improvements
- Improved guidelines on standard attributes and UI metadata
- Lightweight **`oslcomp`** (new preprocessor, no more libclang)
- Evolutions to language syntax
- SPIR-V back end?

Dev Days

- Participated in May & Sept 2025 Dev Days
- May: no participants
- Sept: 3 participants
  - 1 patch submitted, pending minor revision
  - 2 participants claimed issues but did not submit a PR
- Reasons why it's hard for us:
  - Tricky to find "good first issues"
  - C++, LLVM & compiler tech, GPU, rendering → most parts of OSL require lots of specialized knowledge
  - Lack of developer bandwidth
  - **Not good excuses**

Current Status

- OpenSSF Best Practices
  - Passing 100%
  - Silver 84%
  - Gold 78%
- Most outstanding items are security related
  - Some low hanging fruit around docs, signing, etc …
  - Need help to add fuzzing to the project
  - OSL lets you execute arbitrary user code *by design*
  - Should not format your hard-drive or let you gain elevated privileges, but could crash or loop forever
  - How OSL behaves is highly dependent on how it is integrated into a renderer

How to Get Involved

- TSC meetings are every other Thursday at 2pm Pacific Time
  - https://calendar.openshadinglanguage.org/
- Slack channel
  - https://academysoftwarefdn.slack.com/
  - #openshadinglanguage
- Github
  - https://openshadinglanguage.org