KHRONOS GROUP

Slang™

Shader Productivity through Open Source Collaboration

November 2024

Image courtesy of NVIDIA

Slang is Now Under Khronos Open Governance

- **3D developers need a modern, responsive domain-specific shading language**
  - To boost productivity and rapidly expose new technologies and techniques
- **Slang is a fast-moving language leveraging 15 years of R&D and deployment experience**
  - Hosted at NVIDIA since 2017
- **Transition to Khronos hosting to foster industry-wide collaboration and innovation**
  - Step beyond open source, giving all an equal chance to influence and decide Slang's evolution
- **Slang Initiative organized to preserve and enhance open-source project responsiveness**

Slang™

Technical work under streamlined open-source project

Welcome contributors from any engaged company

Community-driven project structure and best practices suited to a shading language

NVIDIA is increasing resources to sustain project velocity

Working Group for logistical, outreach, and funding support

No detailed engineering interaction with the open-source project

Enable the open-source project to focus on technical forward progress

Explore and leverage synergy between Slang, Vulkan and SPIR-V

Slang is Now Under Khronos Open Governance

- **3D developers need a modern, responsive domain-specific shading language**
  - To boost productivity and rapidly expose new technologies and techniques
- **Slang is a fast-moving language leveraging 15 years of R&D and deployment experience**
  - Hosted at NVIDIA since 2017
- **Transition to Khronos hosting to foster industry-wide collaboration and innovation**
  - Step beyond open source, giving all an equal chance to influence and decide Slang's evolution
- **Slang Initiative organized to preserve and enhance open-source project responsiveness**

Slang™

Technical work under streamlined open-source project

Welcome contributors from any engaged company

Community-driven project structure and best practices suited to a shading language

NVIDIA is increasing resources to sustain project velocity

Working Group for logistical, outreach, and funding support

No detailed engineering interaction with the open-source project

Enable the open-source project to focus on technical forward progress

Explore and leverage synergy between Slang, Vulkan and SPIR-V

Urgent Need for Shading Language Innovation

- **Handle increasing scale and complexity of shader codebases**
  - Languages designed for 10s of LOC are being used for 10s of *thousands* of LOC
- **Streamline cross-platform shader portability**
  - Today's multi-target compiler toolchains are often complex, ad-hoc, and fragile
- **Solve the "shader combinatorics" problem**
  - Number of compiled variants seriously degrades compile times, binary sizes, and load times
  - A decades-old problem
- **Unleash significant language innovations**
  - Especially as approach the neural graphics discontinuity
- **Enable an evolutionary path for Vulkan developers**
  - GLSL is no longer innovating new language features

Open-Source, Cross-Platform Compiler

Slang compiler ingests Slang shaders AND provides onramps for HLSL and GLSL codebases

Slang

HLSL 2020

GLSL

Slang™

Open source

https://github.com/shader-slang

SPIR → Vulkan

HLSL → Microsoft DirectX

GLSL → OpenGL

MSL → Metal

WGSL → WebGPU

PyTorch

C (CPU)

CUDA

Optix

Diverse backend targets enable shader code to be written once to run almost anywhere

Try Slang in your Browser!

https://try.shader-slang.org/ (https://try.shader-slang.com/ before launch)

Slang™ | ShaderToy: Ocean | ▶ Run

SPIR-V | Compile

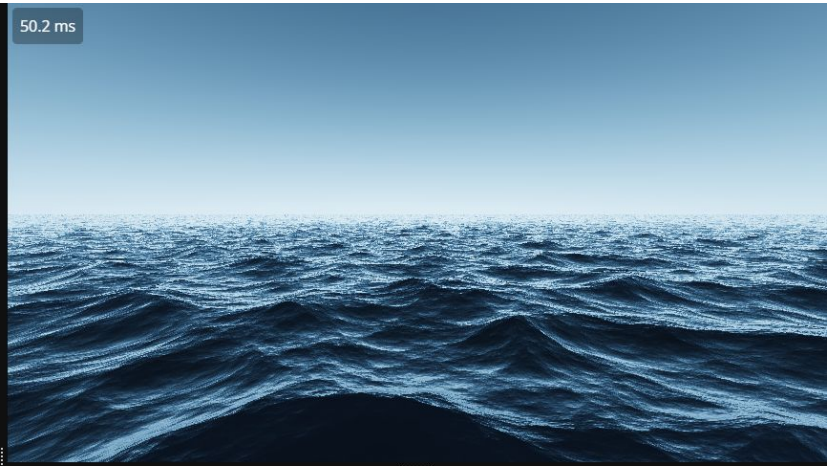
```
// with regard to `position`. See `normal` function for how autodiff is used.

import playground;

static const float DRAG_MULT = 0.38; // changes how much waves pull on the water
static const float WATER_DEPTH = 1.0; // how deep is the water
static const float CAMERA_HEIGHT = 1.5; // how high the camera should be
static const int ITERATIONS_RAYMARCH = 12; // waves iterations of raymarching
static const int ITERATIONS_NORMAL = 37; // waves iterations when calculating normals

// Calculates wave value and its derivative,
// for the wave direction, position in space, wave frequency and time
[Differentiable]
[PreferCheckpoint]
float2 wavedx(float2 position, float2 direction, float frequency, float timeshift) {
  float x = dot(direction, position) * frequency + timeshift;
  float wave = exp(sin(x) - 1.0);
  float dx = wave * cos(x);
  return float2(wave, -dx);
}

// Calculates waves by summing octaves of various waves with various parameters.
// This function is a generic function that can be specialized by different number of iterat
[Differentiable]
float getwaves<int iterations>(float2 position) {
  float wavePhaseShift = length(position) * 0.1; // this is to avoid every octave having exa
  float iter = 0.0; // this will help generating well distributed wave directions
  float frequency = 1.0; // frequency of the wave, this will change every iteration
  float timeMultiplier = 0.0; // time multiplier for the wave, this will change every iterat
  float weight = 1.0;// weight in final sum for the wave, this will change every iteration
  float sumOfValues = 0.0; // will store final sum of values
  float sumOfWeights = 0.0; // will store final sum of weights
```



Target Code | Reflection

```
@binding(3) @group(0) var outputTexture_0 : texture_storage_2d<rgba8unorm, write>;

struct UniformInput_std140_0
{
    @align(16) mousePosition_0 : vec4<f32>,
    @align(16) time_0 : f32,
};

struct GlobalParams_std140_0
{
    @align(16) uniformInput_0 : UniformInput_std140_0,
};
```

Why Another Shading Language?

	GLSL	MSL	WGSL	HLSL	Slang
Actively Evolving	NO	YES	YES	YES	YES
Modular Code Management	NO	NO	NO	NO	YES
Converging with C++	NO	YES	NO	YES	NO*
Auto-diff / Neural Shading	NO	NO	NO	NO	YES
Diverse Backend Targets	NO	NO	NO	DXIL and SPIR-V	YES
Open-Source Compiler(s)	YES	NO	YES	YES	YES
Open Governance	YES	NO	YES	NO	YES

* Slang and HLSL are taking complementary evolutionary paths
HLSL will remain and evolve as a critically important shading language for many developers
Language diversity and choice is good for the graphics ecosystem!

Slang's Benefits for Shader Developers

Slang is an innovative language designed specifically for graphics

A better fit than C++ e.g., C++ Templates do not solve shader combinatorics problem

Onramp Existing Codebases

Enables incremental Slang's adoption
Support for GLSL 4.6 and HLSL 2020

Write Once - Run 'Everywhere'

Multiple, diverse, compiler backends
Platform-specific features managed via semantic checking

Manage Large-scale Code Bases

Generics & Interfaces reduce shader combinations
Modular code through Modules with visibility control

Reduce Shader Compile and Load Times

Modules, Generics and Interfaces avoid re-checking common code
Early type checking via generics speeds front-end compilation

Language Expressivity

Flexible dispatch through generics and interfaces
Application controlled compile-time code generation
Polymorphism via powerful type system

Powerful Tooling and Debugging

IntelliSense support via Visual Studio & VSCode extensions
GPU-assisted validation

Leverage Machine Learning

Deeply integrated, first-class automatic differentiation
Inferencing and *training* in neural shaders

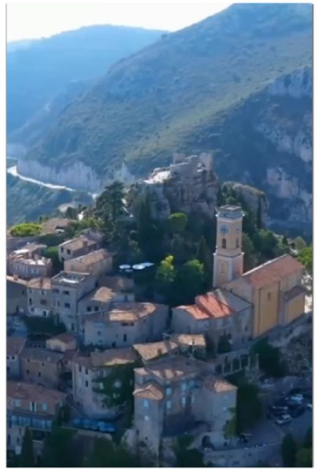
Slang™

The Neural Rendering Revolution

- **Intuitive use of machine learning in real-time shaders**
  - Generate geometry/texture/material LODs, compression, approximations, parameter tuning
  - Embed training inside the renderer e.g., accelerate learned Neural Materials
- **Deeply integrated, first-class automatic differentiation**
  - Eliminates need for additional, separate differential version of every shader

Compression
16x more Texels than D3D/Vulkan textures in same footprint

Neural Materials
Movie-quality photoreal materials in real-time with appearance-based LOD



Neural Shapes
New content creation paradigms and LOD solution

Neural Lighting
Neural Radiance
Neural Path Sampling

Modern Language Design

- **Manage complexity and compile/load times with modules, generics, and interfaces**
  - Modules enable separate compilation with visibility control
  - Interfaces for explicit requirements (similar to Rust traits, Swift protocols, Haskell type classes)
  - Generics for improved code reusability, maintainability and diagnostics without impacting code efficiency
- **Application controlled code generation**
  - Multiple compile-time tradeoffs from same shader code
- **Early type checking via generics**
  - Faster front-end compilation time, better IntelliSense assistance
- **Dynamic dispatch via interfaces**
  - Polymorphism with flexible decoupling and maintainability
- **And many additional goodies…**
  - Associated types & associated constants
  - Namespaces
  - Properties
  - Extensions
  - Operator overloading
  - Automatic type inference
  - …

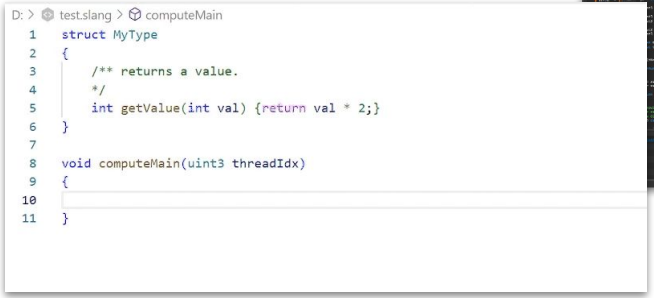
Defining and using an interface - and getting assistance via IntelliSense

Slang Productivity & Tooling

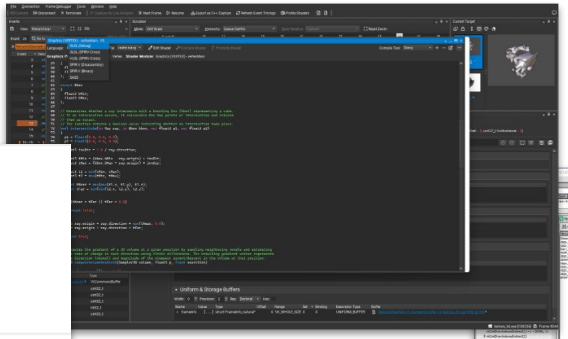
- **Slang Productivity**
  - Debuggability: code structure and identifier names kept close to original form
  - Maintainability: semantic checking of cross-platform code with platform-specific behaviors
  - Backwards compatibility: structured versioning communicates thresholds for breaking changes
  - GPU-assisted validation: enhanced insights into GPU-side run-time behavior

Inspecting & profiling a Slang shader in NVIDIA Nsight

Visual Studio

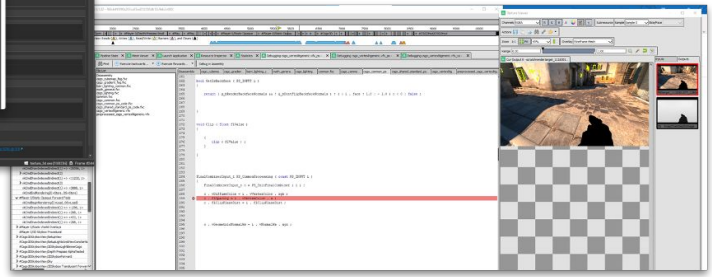


Slang IntelliSense in Visual Studio / VSCode



Slang Tooling Support

RenderDoc



Step-through Slang debugging in RenderDoc

Slang Adopters and Support

VALVE source 2

- Valve migrated entire Source 2 HLSL codebase to Slang
- Minimal changes (~10 lines) needed to compile existing shaders with Slang
- Slang version is in production

AUTODESK

- Slang is used by Aurora path tracing renderer, enables single-source ray tracing codebase
- Slang produces GLSL at runtime when Vulkan backend is used

"Slang builds on the accomplishments of existing languages, while offering novel solutions to the challenges we face in the realm of modern graphics development."

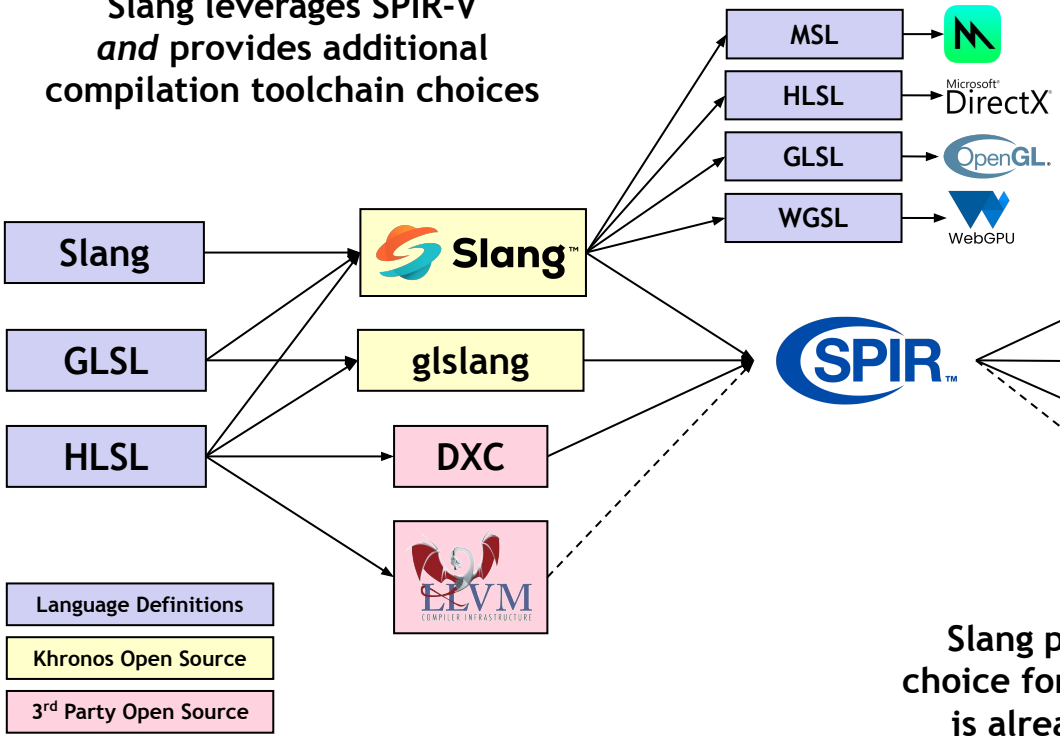
Billy Khan, Director of Engine Technology, id Software

id

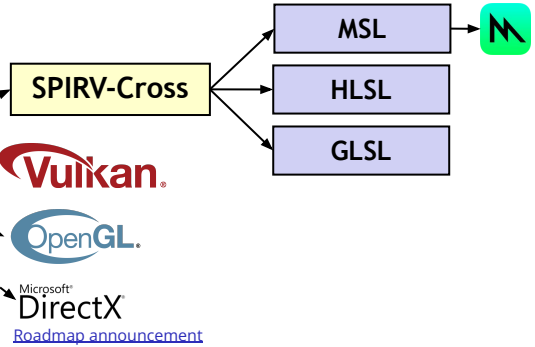
KHRONOS GROUP

Slang and SPIR-V for 3D Graphics

Slang leverages SPIR-V *and* provides additional compilation toolchain choices



Slang is an additional SPIR-V 'client' language



Slang provides an additional language choice for many APIs, including Vulkan, and is already included in the Vulkan SDK

Slang Initiative Organization

Working Group and the Open Source project kept at 'arm's length' to retain responsiveness of language development under open-source best practices

Slang Working Group
Logistical Support

Slang Working Group
Open to any Khronos member

Funding and Legal Support
RFPs, grants, contracts and provisioning of online resources

Marketing
Outreach planning & execution with Khronos marketing

Khronos Ecosystem Coordination
Vulkan SDK/CTS releases
Coordination with other Khronos specifications

Snapshot Ratifications of Language Specification
Specification patent licensing coverage
Manages outbound patent and trademark licensing

NDA GitLab Repo
Forks for experimental hardware features or unreleased SPIR-V extensions

Khronos NDA

Contributions from Khronos members and non-members
RFCs for design proposals
PRs for submissions
Bug Reports

Slang GitHub Repo(s)
Short and friendly CLA
To maximize community engagement

Headers, libraries, test suite and samples
Outbound Apache 2.0 with LLVM Exception

Language specification and documentation
Outbound CC BY 4.0

Committers Channel
Meet in public, private discussions only when necessary
Set technical direction and resolve conflicts
Add or remove Committers
Modify project processes and structure

Committers
Responsive approval process
Commit rights gained through participation
Track record of non-trivial patches and multiple endorsements

Discussions
Logistics | Pre-socialization of PRs
Use cases and feature requests

Slang Open Source Project
Hosts all technical discussions

Discord Channels
Public GitHub Forums

Slang™

Get Involved!

- **Slang resources**
  - https://shader-slang.org/
- **Open-source Slang Repo**
  - Accepting design proposal RFCs, Pull Requests, and Bug Reports
  - https://github.com/shader-slang
- **Discord Discussion Channels**
  - https://khr.io/slangdiscord
- **SIGGRAPH Asia Presentation – Wednesday December 4th**
  - Slang and the 3D Shading Language Landscape
- **Playground - try Slang in your browser**
  - https://try.shader-slang.org/

SIGGRAPH ASIA 2024 TOKYO 東京

Slang™

KHRONOS® GROUP