

# Vulkan SDK Update and Community Feedback

**SIGGRAPH 2019**

# Who is LunarG?

- **3D Graphics Software Consulting Company**
  - Based in Colorado
  - Vulkan, OpenGL, OpenXR, SPIR-V, ...
- **Sponsored by Valve and Google to deliver critical pieces of the Vulkan Ecosystem**
  - Vulkan Loader & Validation Layers
  - Vulkan tools (GFX Reconstruct, apidump, Assistant Layer, ...)
  - **Vulkan SDK**
  - Close collaboration with the Khronos Vulkan Working Group
- **Come learn more about Vulkan at the Khronos BoF day**
  - Wednesday, July 31st
  - J.W Marriott hotel LA Live, Diamond Ballroom 7-10
  - Vulkan sessions beginning at 2PM
  - Networking session with refreshments starts at 5:30PM
    - Visit the LunarG table to **get a FREE GIFT!**



# Agenda

- **We really want this session to be interactive**
  - but will start with some presentation material for context
- **Presentation**
  - What is the Vulkan SDK
    - Intended users
    - Supported platforms
- **Demo**
  - Navigate [vulkan.lunarg.com](https://vulkan.lunarg.com) and SDK content
  - Using vkConfig to configure your validation layers
- **Q&A - we are here to answer your questions.**

These slides are posted at:

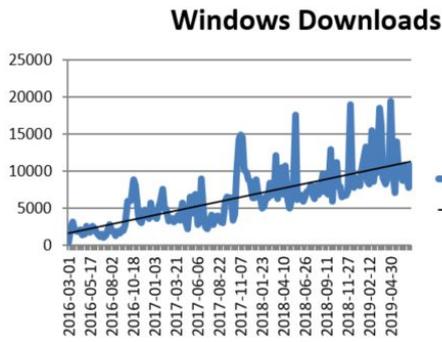
<https://www.lunarg.com/siggraph-2019-lunarg-presents-vulkan-ecosystem-topics>

# Audience Poll

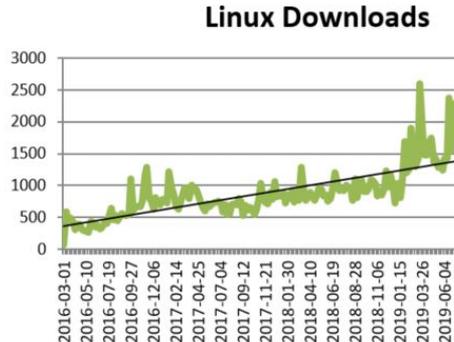
- Who has used the SDK?
- Who has specific questions or feedback for the SDK?

# What is the Vulkan SDK?

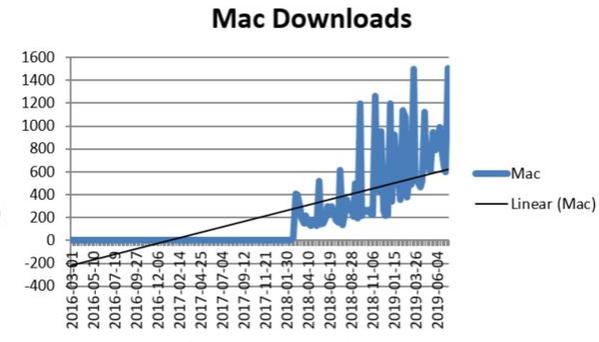
- If you are doing Vulkan application development, the Vulkan SDK is your friend!
- Key layers and tools to aid in the development of your Vulkan application
- Has been available since the initial public launch of Vulkan



~10K/week



~1.5K/week



~0.6K/week

- LunarG recently donated the SDK packaging technologies to Khronos
  - Enables Vulkan WG collaboration

# Where do I get the Vulkan SDK?



Download SDK at: [vulkan.lunarg.com](https://vulkan.lunarg.com) (Windows, Linux - Ubuntu packages, Linux- Tarball, macOS):

The screenshot shows the Vulkan website's developer tools page. The left sidebar contains navigation links: SDK (highlighted), Issues, Docs, Khronos, and a Valve logo. The main content area is titled 'DOWNLOAD DEVELOPER TOOLS FOR' and features icons for Windows, Linux, Mac, and Android. Below these are sections for each platform, each with a 'Latest SDK' or 'Latest SDK Tarball' download button and a table of versions.

**Windows**

Download buttons: Latest SDK, Latest Runtime

Version / Released	File / SHA 256
1.1.108.0 14-Jun-2019	VulkanSDK-1.1.108.0-Installer.exe (467MB) c2fb84632bd6056f92f91a92d4b3710bc51956280a163365be2cb2fa7eb41a9b
	VulkanRT-1.1.108.0-Installer.exe (0MB) 69417ed67917f86325b71146dfcb5acf113284de1709656cdee3f117161cd0dfc8
1.1.106.0 16-Apr-2019	VulkanSDK-1.1.106.0-Installer.exe (465MB) 24b5c9d415912c0fb071973f101671a488b0e33ca991409bcabf1d4beb0b804

**Linux**

Download button: Latest SDK Tarball

Version / Released	File / SHA 256
1.1.108.0 14-Jun-2019	vulkansdk-linux-x86_64-1.1.108.0.tar.gz (167MB) 686bca4d02875f40e0e88fedfb5ba0f31ddff6403709fe41e15b2d94914d0ea
1.1.106.0 16-Apr-2019	vulkansdk-linux-x86_64-1.1.106.0.tar.gz (175MB) 78739f6418f10bc9784743ab3d297b278106663256fe8b7482edfea6c65c7ec3

**Mac**

Download button: Latest SDK

Version / Released	File / SHA 256
--------------------	----------------

**Android**

© 2019 LunarG, Inc. Privacy Policy

# Who are intended users of the Vulkan SDK?

IS:

- Desktop Vulkan Application developers
- Utility layer developers (use the VLF - Vulkan Layer Factory)
- Those learning Vulkan - there is a tutorial

IS NOT:

- End users just needing a Vulkan loader
  - Windows: IHVs deliver the loader with their drivers
  - Most major Linux distros include a Vulkan Loader
  - But you can get the most recent loader from the SDK if you want
- Developing and debugging the layers and tools included in the SDK
  - Use github development process

# SDK Usage Model

IS:

- Be able to configure & run layers and tools included in the SDK
- Ability to see symbols in the layers and tools (good for stack traces)

IS NOT:

- Ability to make modifications to layers/tools and upstream changes
- Ability to debug “into” the validation layers, loader, and SDK tools

# SDK Releases

- About every 4-6 weeks
- Newer Vulkan header and specification
- Latest loader and validation layers
- Continued enhancements to additional developer tools

# SDK contents/docs viewable at [vulkan.lunarg.com](https://vulkan.lunarg.com)



The screenshot shows the Vulkan SDK website. On the left, a dark navigation sidebar contains the following items: SDK, Issues, Docs (highlighted with a red box), Khronos, and logos for Valve and LunarG. A red box also highlights the version dropdown menu showing '1.1.108.0 (Latest)' and a list of links including 'Getting Started', 'Release Notes', 'Loader and Layers', 'Loader', 'Layers Overview and Configuration', 'Validation Layers', 'GPU Assisted Validation', 'Utility Layers', 'API Dump', 'Device Simulation', 'Assistant Layer', 'Monitor', 'Screenshot', 'Tools', 'Vulkan Tools', 'Framework', 'vkconfig', 'Layer Factory', 'VIA', 'vulkaninfo', 'Trace and Replay', 'SPIR-V Toolchain', 'Vulkan Samples', 'Vulkan Tutorial', and 'Build/Run the'. A red arrow points from the text 'Full set of SDK contents and associated documentation' to the version dropdown menu. The main content area features the Vulkan logo, the title 'Getting Started with the Vulkan SDK', a Creative Commons BY-ND license icon, and a section titled 'Version for Windows' with introductory text.

Full set of SDK contents and associated documentation

## Getting Started with the Vulkan SDK

### Version for Windows

This guide describes the requirements and procedure for installing the Vulkan SDK for Windows. It also includes compilation and runtime instructions for demo Vulkan applications. Refer to the Vulkan SDK, Documentation, and Known Issues at the [Vulkan SDK Download Site](#) for the most up to date SDK information.

The Vulkan API is a low-overhead, explicit, cross-platform graphics API that provides applications with direct control over the GPU, maximizing application performance. For more information on the Vulkan specification and API, refer to [Khronos.org](#). For tutorial-level information, refer to the Vulkan tutorial, which can be found in the SDK in the `Documentation\Tutorial\html` directory and at the [Vulkan SDK Download Site](#).

This SDK does NOT include a Vulkan driver. Please contact your GPU hardware vendor for a Vulkan Installable Client Driver (ICD). This SDK will allow you to build Vulkan applications but you will need a Vulkan ICD to execute them.

© 2019 LunarG, Inc.

# GFX Reconstruct (<https://github.com/LunarG/gfxreconstruct>)

- MUCH improved capture/replay tool
- Currently in Beta mode
- Performance Benefits (relative to vktrace/vkreplay)
  - Up to 2X FPS improvement during capture replay
  - Capture file size reduced up to 50%
- vktrace/vkreplay will be deprecated in favor of GFX Reconstruct
  - Fall 2019

# GFX Reconstruct Benefits

- Android is given same priority as desktop in features and support
- Automatic code generation to accommodate evolving API
- Reliable trimming
- Increased portability
  - X86 vs. x64 differences
  - Cross OS portability (i.e. capture on windows, replay on linux).
  - *Cross vendor GPU support (capture on one GPU, replay on another)*
- LZ4 compression for capture data
- *Future valuable plug-ins with minimal code changes*
  - Generate C code program
  - Data mining utilities (search for feature usage)
  - Extract/replace shaders

*\*Items in Italics may not be ready until after vktrace/vkreplay deprecation*

# What is the Vulkan Configurator? (vkconfig)

- GUI tool, included in the Vulkan SDK
- “Dashboard” for developing Vulkan apps
  - (for those who prefer UI)
- VulkanInfo tab, Layer Manager tab, VIA tab
- Layer Manager
  - Enable explicit layers (e.g. KHRONOS\_validation, LUNARG\_api\_dump)
  - Disable implicit layers (e.g. Optimus)
  - Change layer order
  - Change layer settings

# Vulkan Configurator demo



# Vulkan Configurator

- **Make more layers use layer\_settings.txt**
- **UI improvements are under consideration**
- **More intelligence for typical development workflow**
  - Presets for Vulkan API Validation
  - Presets for API dumping
  - Presets for Shader Validation
  - Etc
- **Health tab (distilled version of VulkanInfo and VIA)**
- **Show validation error log directly in Vulkan Configurator with filtering**
- **Target only one executable (currently layer override is global)**
- **Selectable exceptions (a la Visual Studio)**

**File bugs against [github.com/LunarG/VulkanTools](https://github.com/LunarG/VulkanTools)!**

# Backup

# If your first time, be sure to start here



The screenshot shows the Vulkan website's navigation menu on the left. The 'Getting Started' link is highlighted with a red box. The main content area displays the Vulkan logo, the title 'Getting Started with the Vulkan SDK', a Creative Commons BY-ND license icon, and the section 'Version for Windows'. The text under 'Version for Windows' provides instructions for installing the Vulkan SDK on Windows, including references to the Vulkan SDK Download Site, Khronos.org, and the Vulkan tutorial. At the bottom right of the page, there is a copyright notice for LunarG, Inc. © 2019.

# VIA - Vulkan Installation Analyzer



The screenshot shows the Vulkan website's navigation menu on the left. The 'Tools' section is expanded, and 'VIA' is highlighted with a red box. The main content area displays a banner for 'LUNAR VIA Vulkan Installation Analyzer' with a starry background. Below the banner is a Creative Commons license icon (CC BY-ND) and the text 'Copyright © 2015-2019 LunarG, Inc.'. The text states: 'This document is an overview of how to use the LunarG Vulkan Installation Analyzer (VIA). VIA is a tool that can:' followed by a numbered list of three items: 1. Determine the state of Vulkan components on your system, 2. Validate that your Vulkan Loader and drivers are installed properly, 3. Capture your system state in a form that can be used as an attachment when submitting bugs. At the bottom, it says 'This document describes where to find the source for VIA, building it, running it, and how to understand the resulting' and '© 2019 LunarG, Inc.'

# Validation Layers - Fundamental

SDK

Issues

Docs

Khronos

Sponsored by

VALVE

Developed by

LUNAR C

1.1.108.0 (Latest)

- Getting Started
- Release Notes
- Loader and Layers
  - Loader
  - Layers Overview and Configuration
  - Validation Layers**
  - CPU Assisted Validation
- Utility Layers
  - API Dump
  - Device Simulation
  - Assistant Layer
  - Monitor
  - Screenshot
- Tools
  - Vulkan Tools Framework
  - vkconfig
  - Layer Factory
  - VIA
  - vulkaninfo
  - Trace and Replay
  - SPIR-V Toolchain
- Vulkan Samples
  - Vulkan Tutorial
  - Build/Run the

## Vulkan

### Vulkan Validation Layers

#### Validation Layers Included in the SDK

The Vulkan SDK includes the following validation layers:

Layer Name	Layer Type	Description
<a href="#">VK_LAYER_KHRONOS_validation</a>	validation	the main, comprehensive Khronos validation layer -- this layer encompasses the entire functionality of the layers listed below, and supercedes them. As the other layers are deprecated this layer should be used for all validation going forward.
		validate the descriptor set, pipeline state, and dynamic state; © 2019 LunarG, Inc.

# Vulkan Configurator (vkconfig)

SDK

Issues

Docs

Khronos

Sponsored by

VALVE

Developed by

LUNAR G

1.1.108.0 (Latest)

- Getting Started
- Release Notes
- Loader and Layers
  - Loader
  - Layers Overview and Configuration
  - Validation Layers
  - GPU Assisted Validation
- Utility Layers
  - API Dump
  - Device Simulation
  - Assistant Layer
  - Monitor
  - Screenshot
- Tools
  - Vulkan Tools Framework
  - vkconfig**
  - Layer Factory
  - VIA
  - vulkaninfo
  - Trace and Replay
  - SPIR-V Toolchain
- Vulkan Samples
  - Vulkan Tutorial
  - Build/Run the

LUNAR G

Copyright © 2015-2019 LunarG, Inc.

## Vulkan Configurator (vkconfig)

The Vulkan Configurator is a graphical application that allows a user to specify which layers will be loaded by Vulkan applications at runtime. It provides an alternative to setting layers through environment variables or an application's layer selection. In addition, it allows using layers from non-standard locations, selecting the ordering for implicit layers, and specifying settings for layers that Vulkan Configurator supports.

### Using the Vulkan Configurator

The Vulkan Configurator is a graphical user interface (GUI), and does not support any functionality through the system console. It may be launched from the console (as `vkconfig`), but no further functionality will be available from the console.

# Assistant Layer

The screenshot shows the Vulkan website's navigation menu. The 'Assistant Layer' item is highlighted with a red box. The website header includes the Vulkan logo, a '+ Signup' button, and a 'Signin' button. The left sidebar contains links for SDK, Issues, Docs, Khronos, and sponsors like Valve. The main content area features the LunarG logo, a Creative Commons BY-ND license icon, and a copyright notice for 2015-2019 LunarG, Inc.

## VK\_LAYER\_LUNARG\_assistant\_layer

The `VK_LAYER_LUNARG_assistant_layer` functions as a Vulkan 'best practices' layer, and is intended to highlight potential performance issues, questionable usage patterns, common mistakes, and items not specifically prohibited by the Vulkan specification but that may lead to application problems.

This layer will ideally be run periodically along with normal validation checks so that issues may be addressed in early stages of development.

The specific areas covered by this layer are currently tracked in [Github Issue #24: Best Practices/Assistant/Performance Layer](#). Requests for additional checks can be posted through the same issue, or by creating a new Github issue, and current status is tracked there as well.

# SPIR-V Tool Chain

**Vulkan**

+ Signup    Signin

SDK    1.1.108.0 (Latest)

- Getting Started
- Release Notes
- Loader and Layers
  - Loader
  - Layers Overview and Configuration
  - Validation Layers
  - GPU Assisted
  - Validation
- Utility Layers
  - API Dump
  - Device Simulation
  - Assistant Layer
  - Monitor
  - Screenshot
- Tools
  - Vulkan Tools Framework
  - vkconfig
  - Layer Factory
  - VIA
  - vulkaninfo
  - Trace and Replay
  - SPIR-V Toolchain**
  - Vulkan Samples
  - Vulkan Tutorial
  - Build/Run the

Sponsored by **VALVE**

Developed by **LUNAR C**

**Vulkan**

info@lunarg.com

## Vulkan

## SPIR-V Toolchain

The Vulkan graphics API requires that all shaders be presented in SPIR-V format. SPIR-V is a binary intermediate representation for graphical shaders and compute kernels. The advantages and a high-level description of the representation are given in the [SPIR-V White Paper](#). A more detailed description of the SPIR-V supported by this SDK can be found in [The SPIR-V Specification](#).

The Vulkan SDK includes tools for SPIR-V shader generation, inspection, editing, validation, optimization, constant manipulation, and compression improvement. For the Linux SDK, some of these tools are not included as binaries for possible compatibility issues. They can be built with an included script like so `vulkansdk spirvtools`.

The chief member of the SPIR-V toolchain is `glslangValidator`, which can be used to create SPIR-V shaders from equivalent GLSL or HLSL shaders.

The SDK also includes `spirv-opt` which can be used (along with `spirv-remap`, described below) to reduce the size of SPIR-V code by more than 60% through classic optimization transformations. `spirv-opt` can also be used to set and optimize a

© 2019 LunarG, Inc.

# vktrace/vkreplay



1.1.108.0 (Latest)

- Getting Started
- Release Notes
- Loader and Layers
  - Loader
  - Layers Overview and Configuration
  - Validation Layers
  - GPU Assisted Validation
- Utility Layers
  - API Dump
  - Device Simulation
  - Assistant Layer
  - Monitor
  - Screenshot
- Tools
  - Vulkan Tools Framework
  - vkconfig
  - Layer Factory
  - VIA
  - vktrace
  - Trace and Replay**
  - SPRV TOUJIAN
- Vulkan Samples
  - Vulkan Tutorial
  - Build/Run the

**LUNAR C**

Copyright © 2015-2019 LunarG, Inc.

## Trace and Replay Tools

This document describes the vktrace and vkreplay tools which are used for tracing and replaying Vulkan API calls.

### Tracing and Trace Files

The trace program is named `vktrace`. It is used to record an application's Vulkan API calls to a trace file. The call information is stored in the trace file in a compact binary format. The trace files normally have a `.vktrace` suffix. The application can be either a local or remote application.

Options for the `vktrace` command are:

© 2019 LunarG, Inc.

The screenshot shows the Vulkan website interface. On the left, a navigation sidebar contains links for SDK, Issues, Docs, Khronos, and sponsored/developed by sections. The 'API Dump' link is highlighted with a red box. The main content area features the LunarG logo, copyright information (© 2015-2019 LunarG, Inc.), a Creative Commons BY-ND license icon, and the title 'VK\_LAYER\_LUNARG\_api\_dump'. Below the title, a paragraph describes the utility layer's function: 'The `vk_layer_LUNARG_api_dump` utility layer prints API calls, parameters, and values to the identified output stream. It has several settings that can be adjusted by either environment variables or by using the `vk_layer_settings.txt` file.' A section titled 'Enabling the Layer' is followed by a sub-section 'Desktop (Linux/Windows/MacOS)'.

© 2019 LunarG, Inc.

# Screenshot Layer

The screenshot shows the Vulkan website interface. On the left is a navigation sidebar with categories like SDK, Issues, Docs, Khronos, and a list of layers. The 'Screenshot' layer is highlighted with a red box. The main content area displays the LunarG logo, a Creative Commons license icon, and the title 'VK\_LAYER\_LUNARG\_screenshot'. Below the title is a paragraph explaining that the layer records frames to image files based on the `VK_SCREENSHOT_FRAMES` environment variable. A code block at the bottom shows the command to enable the layer on Android.

**1.1.108.0 (Latest)**

- Getting Started
- Release Notes
- Loader and Layers
  - Loader
  - Layers Overview and Configuration
  - Validation Layers
  - GPU Assisted Validation
- Utility Layers
  - API Dump
  - Device Simulation
  - Assistant Layer
  - Monitor
  - Screenshot**
- Vulkan Tools Framework
- vkconfig
- Layer Factory
- VIA
- vulkaninfo
- Trace and Replay
- SPIR-V Toolchain
- Vulkan Samples
  - Vulkan Tutorial
  - Build/Run the

**Vulkan**

+ Signup | Signin

SDK

Issues

Docs

Khronos

Sponsored by

**VALVE**

Developed by

LUNAR G

info@lunarg.com

**LUNAR G**

CC BY ND

Copyright © 2015-2019 LunarG, Inc.

## VK\_LAYER\_LUNARG\_screenshot

The `VK_LAYER_LUNARG_screenshot` layer records frames to image files. The environment variable `VK_SCREENSHOT_FRAMES` can be set to a comma-separated list of frame numbers. When the frames corresponding to these numbers are presented, the screenshot layer will record the image buffer to PPM files in the working directory. For example, if `VK_SCREENSHOT_FRAMES` is set to "4,8,15,16,23,42", the files created will be: 4.ppm, 8.ppm, 15.ppm, etc.

### Android

To enable, set a property that contains target frame:

```
adb shell setprop debug.vulkan.screenshot <framenumber>
```

© 2019 LunarG, Inc.

# Vulkan Tutorial

# Vulkan Specification



**Vulkan.** + Signup Signin

SDK Issues Docs Khronos Sponsored by VALVE Developed by LUNAR C

1.1.108.0 (Latest) Search:

- vkconfig
- Layer Factory
- VIA
- vulkaninfo
- Trace and Replay
- SPIR-V Toolchain
- Vulkan Samples
  - Vulkan Tutorial
  - Build/Run the Samples
  - Samples Index
  - Vulkan Spec**
  - Chunked Spec
  - Full Vulkan Spec
  - Vulkan Man Pages
- White Papers
  - Unified Validation Layer
  - GPU Assisted Validation
  - SPIR-V Shader Size Reduction
  - Vulkan 1.1 vs 1.0
  - Debug Utilities
  - Subgroup Tutorial
  - Pipeline Barriers
  - 3rd Party Resources

0. Preamble

1. Introduction

- 1.1. Document Conventions

2. Fundamentals

- 2.1. Host and Device Environment
- 2.2. Execution Model
- 2.3. Object Model
- 2.4. Application Binary Interface
- 2.5. Command Syntax and Duration
- 2.6. Threading Behavior
- 2.7. Errors
- 2.8. Numeric Representation and Computation
- 2.9. Fixed-Point Data Conversions
- 2.10. Common Object Types

3. Initialization

- 3.1. Command Function Pointers

## Vulkan® 1.1.108 - A Specification (with all published extensions)

The Khronos® Vulkan Working Group – Version 1.1.108, 2019-06-14 19:37:29Z  
| from git branch: [Git branch information not available commit: 40ba6e9a4468915aa3aef2ccc153d60e7988193f](#)

Copyright 2014-2019 The Khronos Group Inc.

This Specification is protected by copyright laws and contains material proprietary to Khronos. Except as described by these terms, it or any components may not be reproduced, republished, distributed, transmitted, displayed, broadcast or otherwise exploited in any manner without the express prior written permission of Khronos. Khronos grants a conditional copyright license to use and reproduce the unmodified Specification for any purpose, without fee or royalty, EXCEPT no licenses to any patent, trademark or other intellectual property rights are granted under these terms.

Khronos makes no, and expressly disclaims any, representations or warranties, express or implied, regarding this Specification, including, without limitation: merchantability, fitness for a particular purpose, non-infringement of any intellectual property, correctness, accuracy, completeness, timeliness, and reliability. Under no circumstances will Khronos, or any of its Promoters, Contributors or Members, or their respective partners, officers, directors, employees, agents or representatives be liable for any damages, whether direct, indirect, special or consequential damages for lost revenues, lost profits, or otherwise, arising from or in connection with these materials.

This Specification has been created under the Khronos Intellectual Property Rights Policy, which is Attachment A of the Khronos Group Membership Agreement available at [https://www.khronos.org/files/member\\_agreement.pdf](https://www.khronos.org/files/member_agreement.pdf), and which defines the terms 'Scope', 'Compliant Portion', and 'Necessary Patent Claims'. Parties desiring to implement the Specification and

# White Papers & Third Party Resources

The screenshot shows the Vulkan website interface. The top navigation bar includes the Vulkan logo, a '+ Signup' button, and a 'Signin' button. The left sidebar contains navigation links for 'SDK', 'Issues', 'Docs', 'Khronos', and 'Sponsored by VALVE'. Below this, it says 'Developed by LUNAR C'. The main content area is titled 'Vulkan® 1.1.108 - A Specification (with all published extensions)'. It includes a search bar, a table of contents, and the main text of the specification. The 'White Papers' section in the sidebar is highlighted with a red box, listing various documents like 'Unified Validation Layer', 'GPU Assisted Validation', etc.

**White Papers**

- Unified Validation Layer
- GPU Assisted Validation
- SPIR-V Shader Size Reduction
- Vulkan 1.1 vs 1.0
- Debug Utilities
- Subgroup Tutorial
- Pipeline Barriers
- 3rd Party Resources
- RenderDoc
- Volk - Meta-Loader for Vulkan API
- V-EZ
- DXC
- Memory Allocator
- gfx-rs
- Vulkano

## Vulkan® 1.1.108 - A Specification (with all published extensions)

The Khronos® Vulkan Working Group – Version 1.1.108, 2019-06-14 19:37:29Z  
| from git branch: Git branch information not available commit: 40ba6e9a4468915aa3aef2ccc153d60e7988193f

Copyright 2014-2019 The Khronos Group Inc.

This Specification is protected by copyright laws and contains material proprietary to Khronos. Except as described by these terms, it or any components may not be reproduced, republished, distributed, transmitted, displayed, broadcast or otherwise exploited in any manner without the express prior written permission of Khronos. Khronos grants a conditional copyright license to use and reproduce the unmodified Specification for any purpose, without fee or royalty, EXCEPT no licenses to any patent, trademark or other intellectual property rights are granted under these terms.

Khronos makes no, and expressly disclaims any, representations or warranties, express or implied, regarding this Specification, including, without limitation: merchantability, fitness for a particular purpose, non-infringement of any intellectual property, correctness, accuracy, completeness, timeliness, and reliability. Under no circumstances will Khronos, or any of its Promoters, Contributors or Members, or their respective partners, officers, directors, employees, agents or representatives be liable for any damages, whether direct, indirect, special or consequential damages for lost revenues, lost profits, or otherwise, arising from or in connection with these materials.

This Specification has been created under the Khronos Intellectual Property Rights Policy, which is Attachment A of the Khronos Group Membership Agreement available at [https://www.khronos.org/files/member\\_agreement.pdf](https://www.khronos.org/files/member_agreement.pdf), and which defines the terms 'Scope', 'Compliant Portion', and 'Necessary Patent Claims'. Parties desiring to implement the Specification and

# What is Included in the Vulkan SDK?



<b>Getting Started</b>	First time user? Definitely start here...
<b>VIA - Vulkan Installation Analyzer</b>	Verifies your Vulkan installation
<b>Vulkaninfo</b>	For each GPU, identifies available extensions, properties, formats, etc. Also indicates available Vulkan Layers
<b>Vulkan Loader</b>	Application interfaces with the Loader which does the discovery of ICD's and available Vulkan Layers
<b>Vulkan Validation Layers</b>	Validate correct Vulkan API usage by your application
<b>Vulkan Configurator (vkconfig)</b>	GUI Configurator for Vulkan Layers
<b>Assistant Layer</b>	Identifies bad Vulkan API usage (that still may be conformant)
<b>SPIR-V Tool Chain</b>	Tools for SPIR-V shader generation, inspection, editing, validation, optimization, constant manipulation, and compression improvement.
<b>api_dump</b>	Dump the api calls and parameters used by your application in human readable form

# What is Included in the Vulkan SDK?



<b>vktrace/vkreplay</b>  (to be replaced with GFX Reconstruct)	Vulkan API trace and replay tool. Very useful for creating trace files for sharing issues, regression testing, and debugging with IHVs.
<b>Screenshot Layer</b>	Convert frames to image files by specifying a frame number, or list of frames
<b>Vulkan Layer Factory</b>	Want to create your own layer? Based on the canonical Vulkan layer model, the VLF facilitates the creation of Vulkan Layers. The layer factory hides the majority of the loader-layer interface, layer boilerplate, setup and initialization, and complexities of layer development.
<b>Vulkan Tutorial</b>	Our tutorial is very basic and will walk you through the steps to create a very simple Vulkan program
<b>Vulkan Specification and Vulkan Headers</b>	With each Vulkan SDK release, we include the appropriate version of the Vulkan Specification and headers.
<b>White Papers and links to 3rd Party resources</b>	We have collected some white papers and links to 3rd party resources that are useful