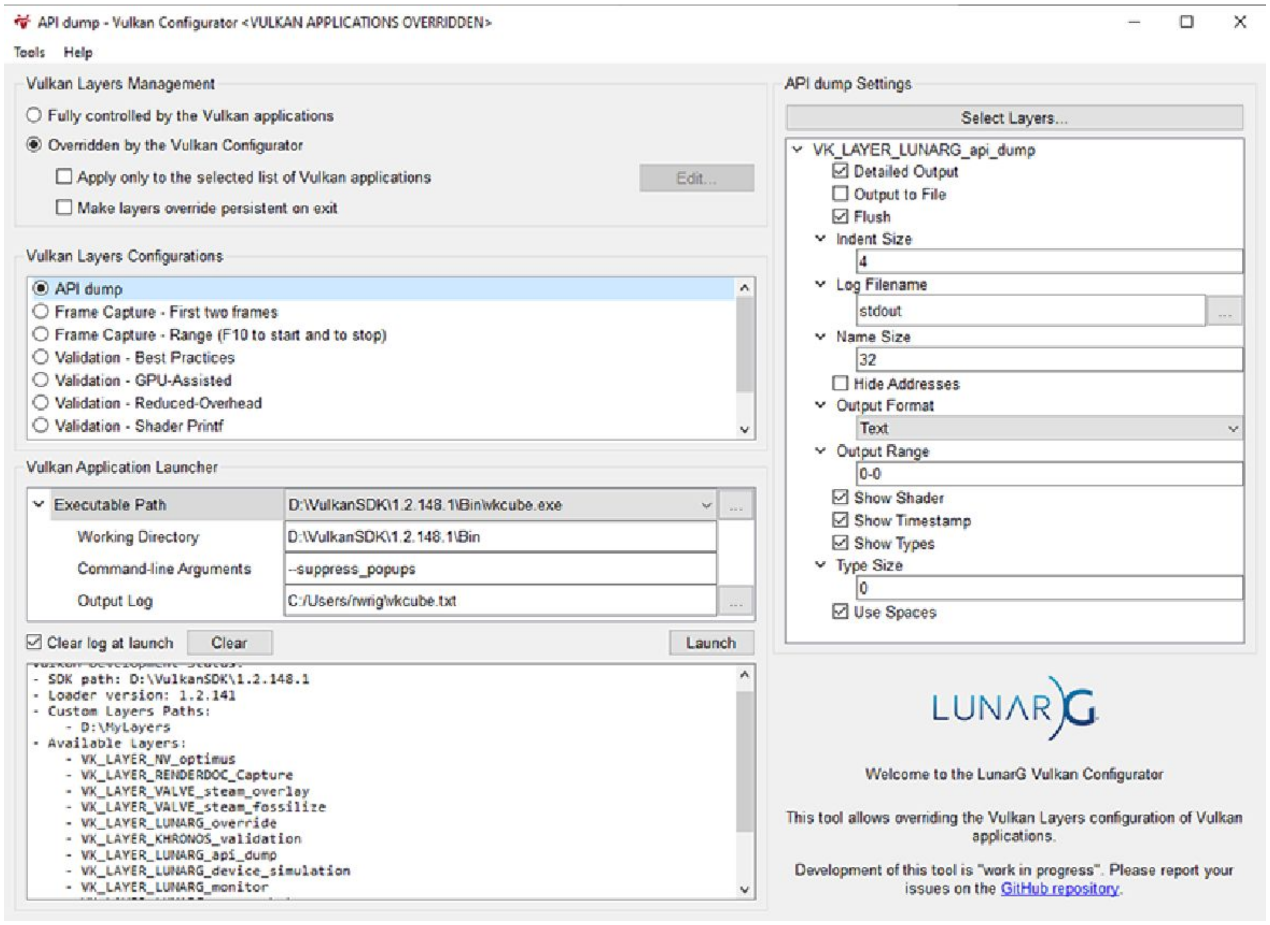
Introducing the new Vulkan Configurator

Richard Wright, LunarG
August 2020

With the August 2020 release of the Vulkan SDK, LunarG has introduced a new, reimagined, version of vkconfig, also called the Vulkan Configurator. We are very excited about this new release and plan to evolve vkconfig to become a central part of your Vulkan toolset.

What is the Vulkan Configurator and what are the benefits?

The Vulkan Configurator is a powerful, yet easy to use, tool that manages your system's *implicit* (automatically loaded) layer environment; it essentially puts you in charge of your system's Vulkan implicit layer configuration.


API dump - Vulkan Configurator <VULKAN APPLICATIONS OVERRIDDEN>
Tools
Help
Edit...
Select Layers...
Clear
Launch
...

Vulkan Layers Management
- Fully controlled by the Vulkan applications
- [✓ Overridden by the Vulkan Configurator]
  - ☐ Apply only to the selected list of Vulkan applications
  - ☐ Make layers override persistent on exit

Vulkan Layers Configurations
- [✓ API dump]
- Frame Capture - First two frames
- Frame Capture - Range (F10 to start and to stop)
- Validation - Best Practices
- Validation - GPU-Assisted
- Validation - Reduced-Overhead
- Validation - Shader Printf

Vulkan Application Launcher

Executable Path	D:\VulkanSDK\1.2.148.1\Bin\vkcube.exe
Working Directory	D:\VulkanSDK\1.2.148.1\Bin
Command-line Arguments	--suppress_popups
Output Log	C:/Users/rwrig\vkcube.txt

☑ Clear log at launch

```
- SDK path: D:\VulkanSDK\1.2.148.1
- Loader version: 1.2.141
- Custom Layers Paths:
    - D:\MyLayers
- Available Layers:
    - VK_LAYER_NV_optimus
    - VK_LAYER_RENDERDOC_Capture
    - VK_LAYER_VALVE_steam_overlay
    - VK_LAYER_VALVE_steam_fossilize
    - VK_LAYER_LUNARG_override
    - VK_LAYER_KHRONOS_validation
    - VK_LAYER_LUNARG_api_dump
    - VK_LAYER_LUNARG_device_simulation
    - VK_LAYER_LUNARG_monitor
```

API dump Settings

VK_LAYER_LUNARG_api_dump
- ☑ Detailed Output
- ☐ Output to File
- ☑ Flush
- Indent Size: 4
- Log Filename: stdout
- Name Size: 32
- ☐ Hide Addresses
- Output Format: Text
- Output Range: 0-0
- ☑ Show Shader
- ☑ Show Timestamp
- ☑ Show Types
- Type Size: 0
- ☑ Use Spaces

Welcome to the LunarG Vulkan Configurator

This tool allows overriding the Vulkan Layers configuration of Vulkan applications.

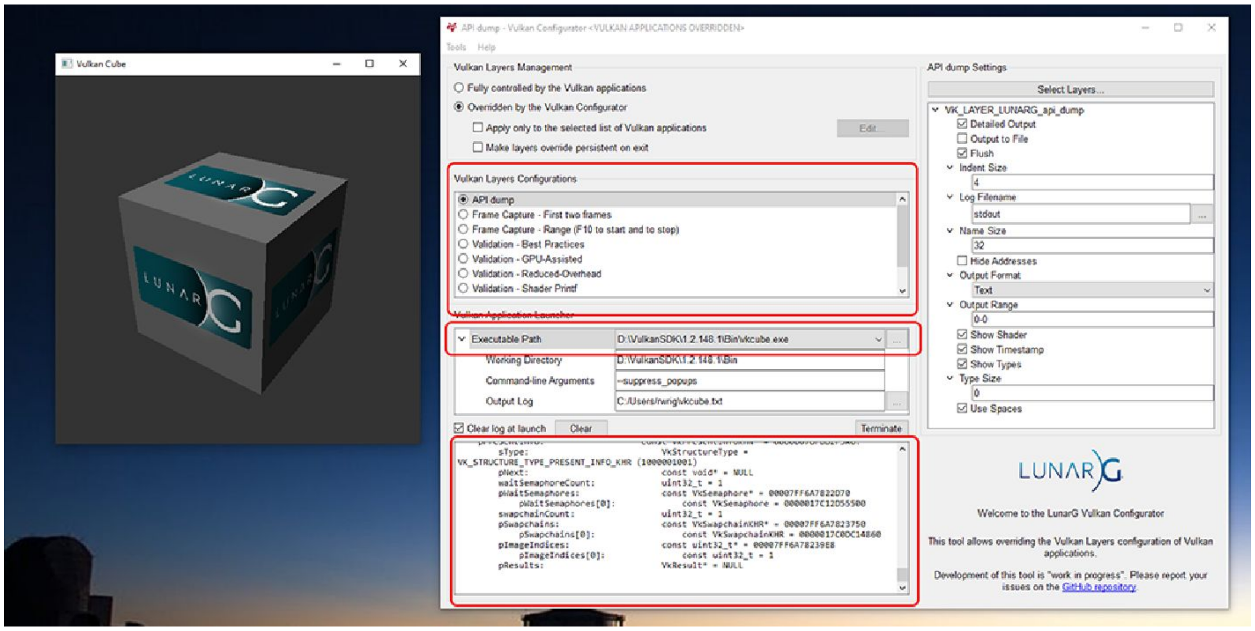
Development of this tool is "work in progress". Please report your issues on the GitHub repository.

Figure 1. The new Vulkan Configurator gives you control over your systems implicit layer configuration.

Vulkan Layers are a powerful and flexible feature of Vulkan, with some providing validation services or extended features to your Vulkan applications. Some layers are also an integral part of your graphics hardware's Vulkan Driver package.

Vulkan Configurator will discover all the layers you have installed on your system, and will allow you to select any number of them to be loaded automatically by Vulkan applications on your computer. This overrides the normal loader behavior and gives you more control over the layers that are loaded with your programs. This means you can turn layers on and off, and examine their output without having to recompile your or even someone else's Vulkan code. This is really useful for all kinds of run time diagnostics, catching Vulkan usage errors early, and sometimes just figuring out why your Vulkan code is not behaving as you expected.

On the first startup, the Vulkan Configurator searches for all the layers installed on your computer. It sets up the initial set of layer configurations and if any configurations don't have the layers required to support them, they will be disabled. It also searches for the vkcube sample from your SDK and automatically adds that to the application launcher as a usable demonstration of using vkconfig. With the API dump selected for example, you can launch vkcube directly and see the API dump writing continuously to your log window.


Vulkan Cube
API dump - Vulkan Configurator <VULKAN APPLICATIONS OVERRIDDEN>
Tools
Help
Edit...
Clear
Terminate
Select Layers...
LUNARG

Vulkan Layers Management
- ( ) Fully controlled by the Vulkan applications
- (●) Overridden by the Vulkan Configurator
  - ☐ Apply only to the selected list of Vulkan applications
  - ☐ Make layers override persistent on exit

Vulkan Layers Configurations
- (●) API dump
- ( ) Frame Capture - First two frames
- ( ) Frame Capture - Range (F10 to start and to stop)
- ( ) Validation - Best Practices
- ( ) Validation - GPU-Assisted
- ( ) Validation - Reduced-Overhead
- ( ) Validation - Shader Printf

Vulkan Application Launcher

Executable Path	D:\VulkanSDK\1.2.148.1\Bin\vkcube.exe
Working Directory	D:\VulkanSDK\1.2.148.1\Bin
Command-line Arguments	--suppress_popups
Output Log	C:/Users/rwrig/vkcube.txt

☑ Clear log at launch

```
        sType:                           VkStructureType =
VK_STRUCTURE_TYPE_PRESENT_INFO_KHR (1000001001)
        pNext:                           const void* = NULL
        waitSemaphoreCount:              uint32_t = 1
        pWaitSemaphores:                 const VkSemaphore* = 00007FF6A7822D70
            pWaitSemaphores[0]:              const VkSemaphore = 0000017C12D55500
        swapchainCount:                  uint32_t = 1
        pSwapchains:                     const VkSwapchainKHR* = 00007FF6A7823750
            pSwapchains[0]:                  const VkSwapchainKHR = 0000017C0DC14860
        pImageIndices:                   const uint32_t* = 00007FF6A78239E8
            pImageIndices[0]:                const uint32_t = 1
        pResults:                        VkResult* = NULL
```

API dump Settings
- VK_LAYER_LUNARG_api_dump
  - ☑ Detailed Output
  - ☐ Output to File
  - ☑ Flush
  - Indent Size: 4
  - Log Filename: stdout
  - Name Size: 32
  - ☐ Hide Addresses
  - Output Format: Text
  - Output Range: 0-0
  - ☑ Show Shader
  - ☑ Show Timestamp
  - ☑ Show Types
  - Type Size: 0
  - ☑ Use Spaces

Welcome to the LunarG Vulkan Configurator

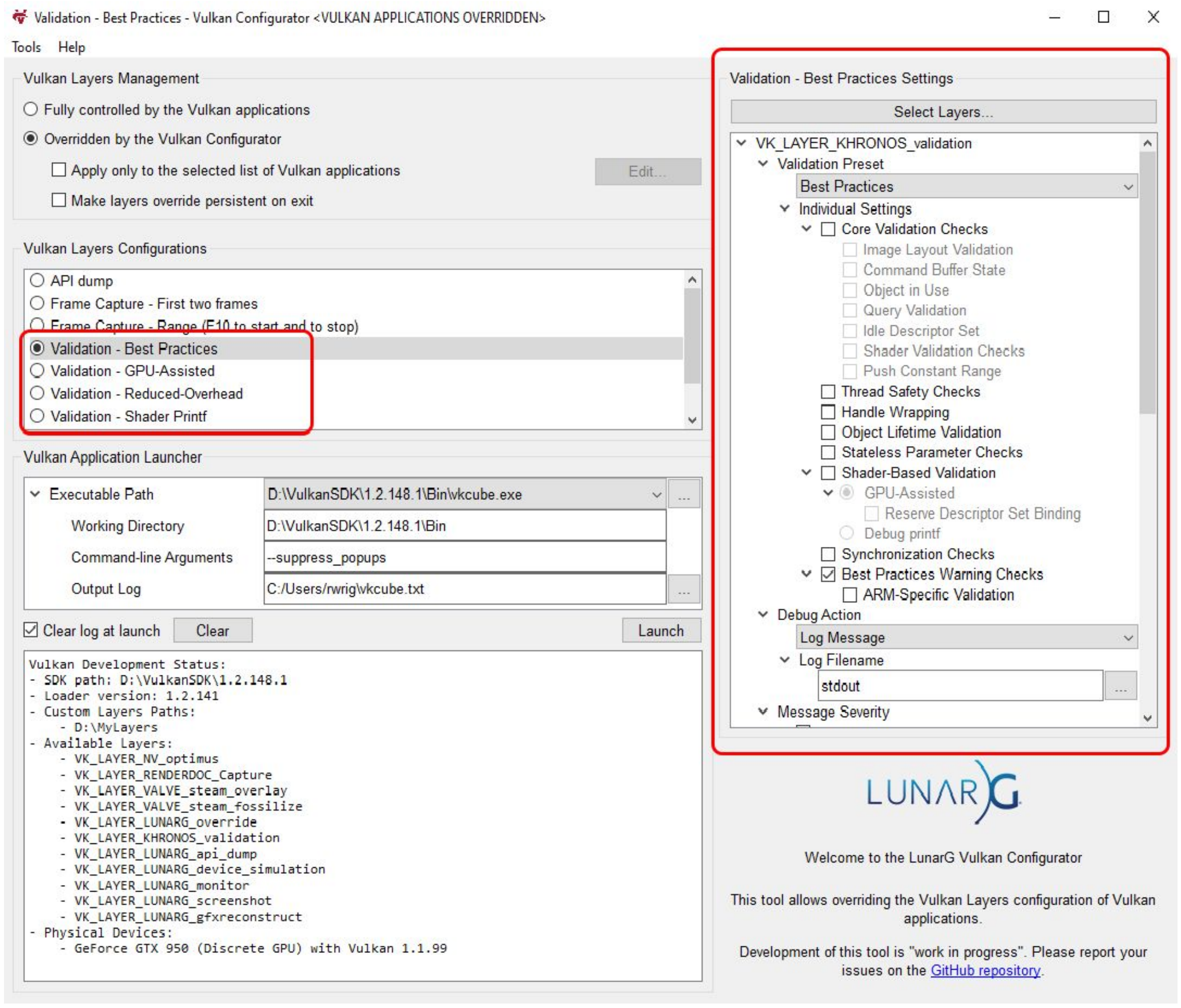
This tool allows overriding the Vulkan Layers configuration of Vulkan applications.

Development of this tool is "work in progress". Please report your issues on the GitHub repository.

Figure 2. Multiple layer configurations are available, and can be created by the developer. Layer output can also be monitored directly from the Vulkan Configurator.

You have the option to leave the override active after vkconfig terminates, and you can specify only a particular list of applications that are to be affected. This is handy as you can be working on a set of your own Vulkan programs, but not worry that these layers are going to impact performance anywhere if you need a quick break to play vkDoom. This is a significant advancement over the last vkconfig and it required changes to the Vulkan loader to support this. Making use of this capability means you never need to worry about custom override configurations affecting any other Vulkan applications on your system that you didn't explicitly intend. You can also temporarily turn off the override layer and it will have no effect on Vulkan applications anywhere on your system.

Each configuration consists of one or more Vulkan Layers, and some of these layers may have their own individual settings that you may want to edit. These settings are saved as part of the configuration and configurations that use the same layer can be set up in different ways. A good example are the Validation configurations that all use the Khronos Validation Layer. Each configuration however is set up differently to enable a given type of validation output.



Validation - Best Practices - Vulkan Configurator <VULKAN APPLICATIONS OVERRIDDEN>

Tools Help

Vulkan Layers Management

- ○ Fully controlled by the Vulkan applications
- ● Overridden by the Vulkan Configurator
  - ☐ Apply only to the selected list of Vulkan applications
  - ☐ Make layers override persistent on exit

Vulkan Layers Configurations

- ○ API dump
- ○ Frame Capture - First two frames
- ○ Frame Capture - Range (F10 to start and to stop)
- ● Validation - Best Practices
- ○ Validation - GPU-Assisted
- ○ Validation - Reduced-Overhead
- ○ Validation - Shader Printf

Vulkan Application Launcher

Executable Path	D:\VulkanSDK\1.2.148.1\Bin\vkcube.exe
Working Directory	D:\VulkanSDK\1.2.148.1\Bin
Command-line Arguments	--suppress_popups
Output Log	C:/Users/rwrig/vkcube.txt

☑ Clear log at launch

```
Vulkan Development Status:
- SDK path: D:\VulkanSDK\1.2.148.1
- Loader version: 1.2.141
- Custom Layers Paths:
    - D:\MyLayers
- Available Layers:
    - VK_LAYER_NV_optimus
    - VK_LAYER_RENDERDOC_Capture
    - VK_LAYER_VALVE_steam_overlay
    - VK_LAYER_VALVE_steam_fossilize
    - VK_LAYER_LUNARG_override
    - VK_LAYER_KHRONOS_validation
    - VK_LAYER_LUNARG_api_dump
    - VK_LAYER_LUNARG_device_simulation
    - VK_LAYER_LUNARG_monitor
    - VK_LAYER_LUNARG_screenshot
    - VK_LAYER_LUNARG_gfxreconstruct
- Physical Devices:
    - GeForce GTX 950 (Discrete GPU) with Vulkan 1.1.99
```

Validation - Best Practices Settings

- VK_LAYER_KHRONOS_validation
  - Validation Preset: Best Practices
  - Individual Settings
    - ☐ Core Validation Checks
      - ☐ Image Layout Validation
      - ☐ Command Buffer State
      - ☐ Object in Use
      - ☐ Query Validation
      - ☐ Idle Descriptor Set
      - ☐ Shader Validation Checks
      - ☐ Push Constant Range
    - ☐ Thread Safety Checks
    - ☐ Handle Wrapping
    - ☐ Object Lifetime Validation
    - ☐ Stateless Parameter Checks
    - ☐ Shader-Based Validation
      - ● GPU-Assisted
        - ☐ Reserve Descriptor Set Binding
      - ○ Debug printf
    - ☐ Synchronization Checks
    - ☑ Best Practices Warning Checks
      - ☐ ARM-Specific Validation
  - Debug Action: Log Message
    - Log Filename: stdout
  - Message Severity

Welcome to the LunarG Vulkan Configurator

This tool allows overriding the Vulkan Layers configuration of Vulkan applications.

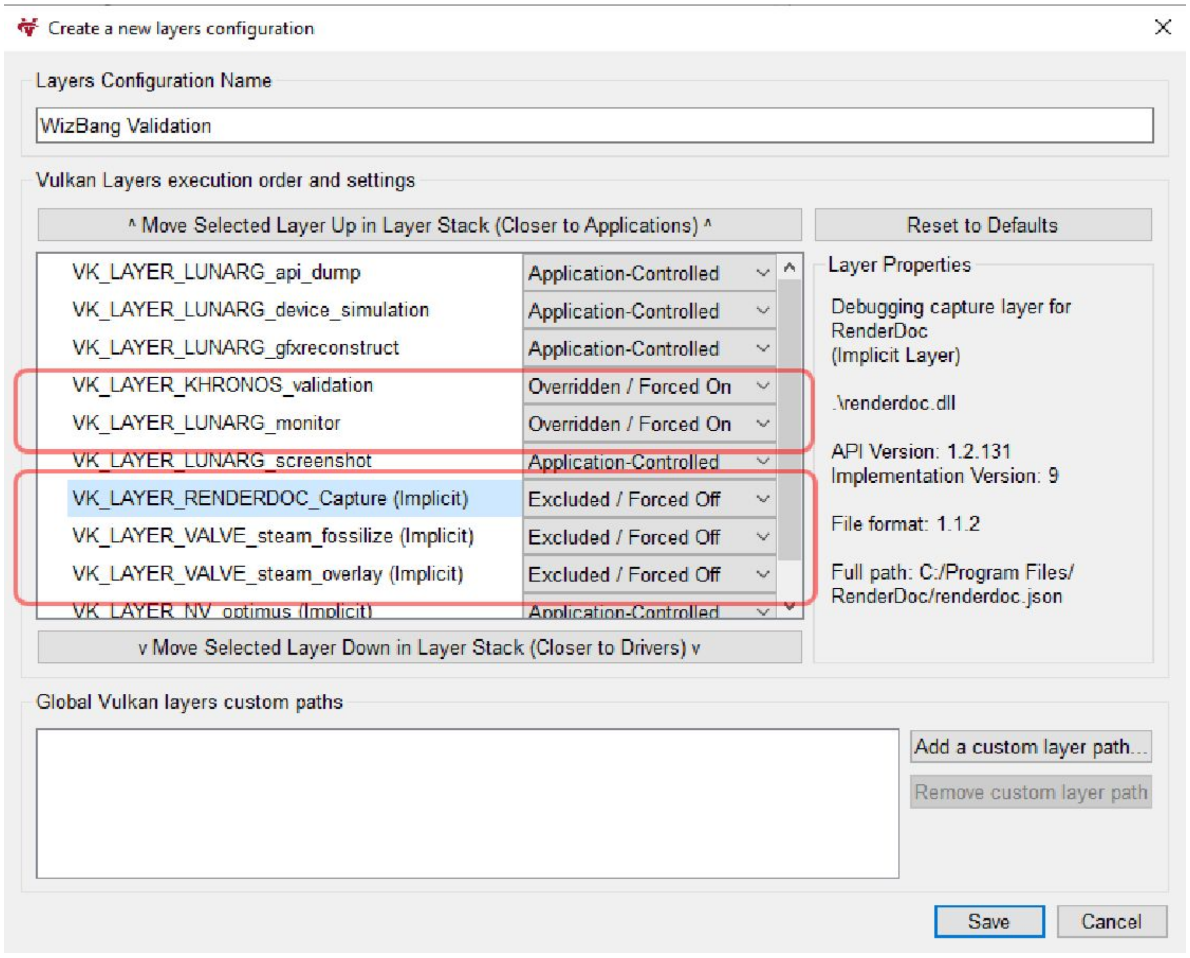
Development of this tool is "work in progress". Please report your issues on the GitHub repository.

Figure 3. Many layers have individual settings to customize how they behave and the services they perform.

Not only can you edit the layers settings, you can add and remove layers from any given configuration, or even make your own. Right click on any configuration and click “Select Layers” to add or remove layers from the configuration. You can also select “New…” to create a brand-new custom layer configuration.

For example, to create a new configuration that contains the Khronos Validation Layer, but also displays the frame rate of our Vulkan application, we could combine the Khronos layer with the LunarG Monitor layer. We could even explicitly exclude the three implicit layers found on our system that might be loaded as well to keep them out of the way.

By default, the layers are marked as “Application Controlled,” which means we keep our hands off and the application can choose whether or not to load these layers. If we select ”Overridden / Forced On,” then the layer becomes part of our override configuration and will be automatically loaded by the Vulkan loader -- whether the application wants it or not.


Create a new layers configuration
Layers Configuration Name
Vulkan Layers execution order and settings
^ Move Selected Layer Up in Layer Stack (Closer to Applications) ^
Reset to Defaults
Layer Properties
v Move Selected Layer Down in Layer Stack (Closer to Drivers) v
Global Vulkan layers custom paths
Add a custom layer path...
Remove custom layer path
Save
Cancel

Layers Configuration Name: WizBang Validation

VK_LAYER_LUNARG_api_dump	Application-Controlled
VK_LAYER_LUNARG_device_simulation	Application-Controlled
VK_LAYER_LUNARG_gfxreconstruct	Application-Controlled
VK_LAYER_KHRONOS_validation	Overridden / Forced On
VK_LAYER_LUNARG_monitor	Overridden / Forced On
VK_LAYER_LUNARG_screenshot	Application-Controlled
VK_LAYER_RENDERDOC_Capture (Implicit)	Excluded / Forced Off
VK_LAYER_VALVE_steam_fossilize (Implicit)	Excluded / Forced Off
VK_LAYER_VALVE_steam_overlay (Implicit)	Excluded / Forced Off
VK_LAYER_NV_optimus (Implicit)	Application-Controlled

Debugging capture layer for RenderDoc
(Implicit Layer)

.\renderdoc.dll

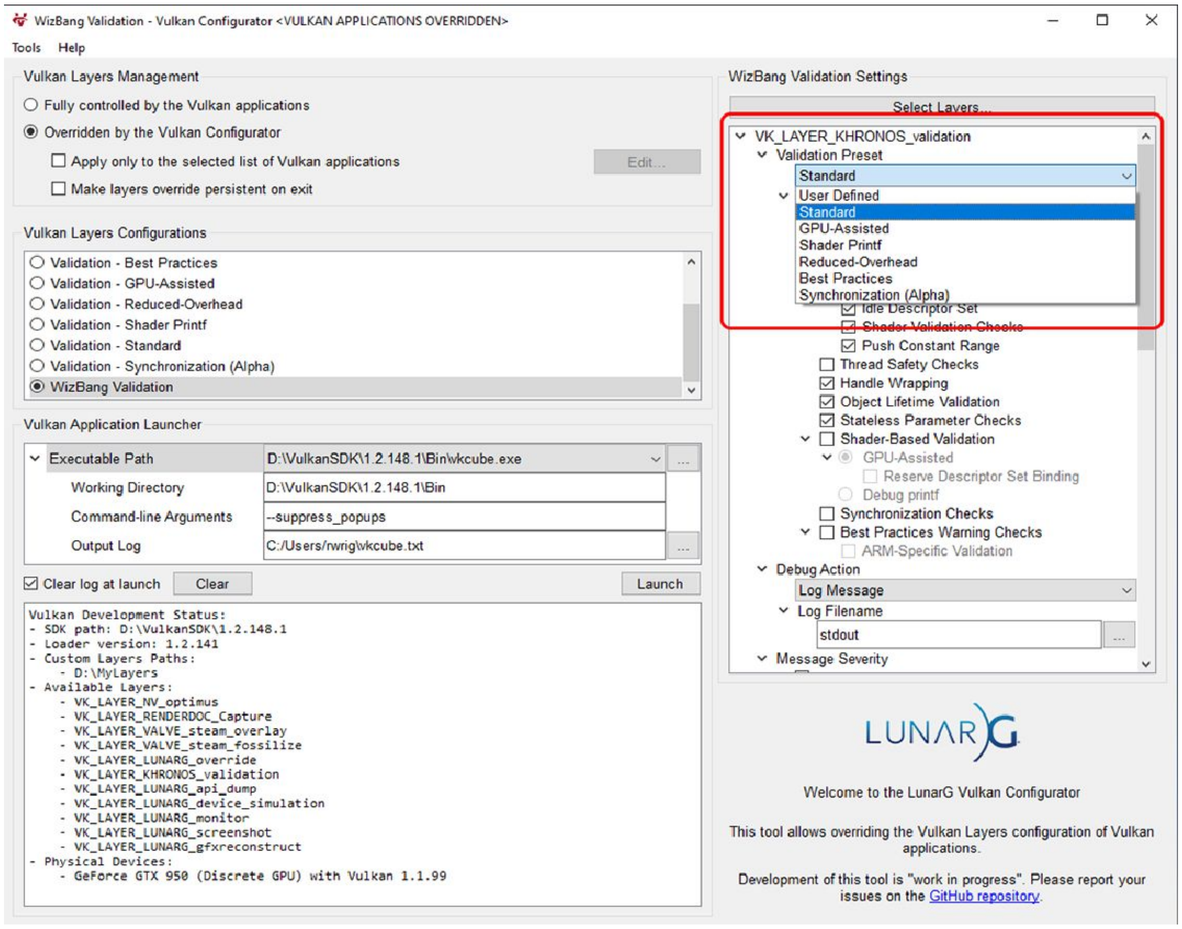
API Version: 1.2.131
Implementation Version: 9

File format: 1.1.2

Full path: C:/Program Files/RenderDoc/renderdoc.json

Figure 4. You have complete control over the layers that are included or excluded from your Vulkan programs.

For layer settings for the Khronos Validation Layer, we also have some presets for the myriad of settings to give you a little help in setting up a well-defined validation scenario. Here I'll just select the "Standard Validation" settings.


WizBang Validation - Vulkan Configurator <VULKAN APPLICATIONS OVERRIDDEN>

Vulkan Layers Management

- ○ Fully controlled by the Vulkan applications
- ● Overridden by the Vulkan Configurator
  - ☐ Apply only to the selected list of Vulkan applications
  - ☐ Make layers override persistent on exit

Vulkan Layers Configurations

- ○ Validation - Best Practices
- ○ Validation - GPU-Assisted
- ○ Validation - Reduced-Overhead
- ○ Validation - Shader Printf
- ○ Validation - Standard
- ○ Validation - Synchronization (Alpha)
- ● WizBang Validation

Vulkan Application Launcher

Executable Path	D:\VulkanSDK\1.2.148.1\Bin\vkcube.exe
Working Directory	D:\VulkanSDK\1.2.148.1\Bin
Command-line Arguments	--suppress_popups
Output Log	C:/Users/rwrig/vkcube.txt

☑ Clear log at launch

```
Vulkan Development Status:
- SDK path: D:\VulkanSDK\1.2.148.1
- Loader version: 1.2.141
- Custom Layers Paths:
    - D:\MyLayers
- Available Layers:
    - VK_LAYER_NV_optimus
    - VK_LAYER_RENDERDOC_Capture
    - VK_LAYER_VALVE_steam_overlay
    - VK_LAYER_VALVE_steam_fossilize
    - VK_LAYER_LUNARG_override
    - VK_LAYER_KHRONOS_validation
    - VK_LAYER_LUNARG_api_dump
    - VK_LAYER_LUNARG_device_simulation
    - VK_LAYER_LUNARG_monitor
    - VK_LAYER_LUNARG_screenshot
    - VK_LAYER_LUNARG_gfxreconstruct
- Physical Devices:
    - GeForce GTX 950 (Discrete GPU) with Vulkan 1.1.99
```

WizBang Validation Settings

- VK_LAYER_KHRONOS_validation
  - Validation Preset: Standard
    - User Defined
    - Standard
    - GPU-Assisted
    - Shader Printf
    - Reduced-Overhead
    - Best Practices
    - Synchronization (Alpha)
  - ☑ Idle Descriptor Set
  - ☑ Shader Validation Checks
  - ☑ Push Constant Range
  - ☐ Thread Safety Checks
  - ☑ Handle Wrapping
  - ☑ Object Lifetime Validation
  - ☑ Stateless Parameter Checks
  - ☐ Shader-Based Validation
    - ◉ GPU-Assisted
      - ☐ Reserve Descriptor Set Binding
    - ○ Debug printf
  - ☐ Synchronization Checks
  - ☐ Best Practices Warning Checks
    - ☐ ARM-Specific Validation
  - Debug Action: Log Message
    - Log Filename: stdout
  - Message Severity

Welcome to the LunarG Vulkan Configurator

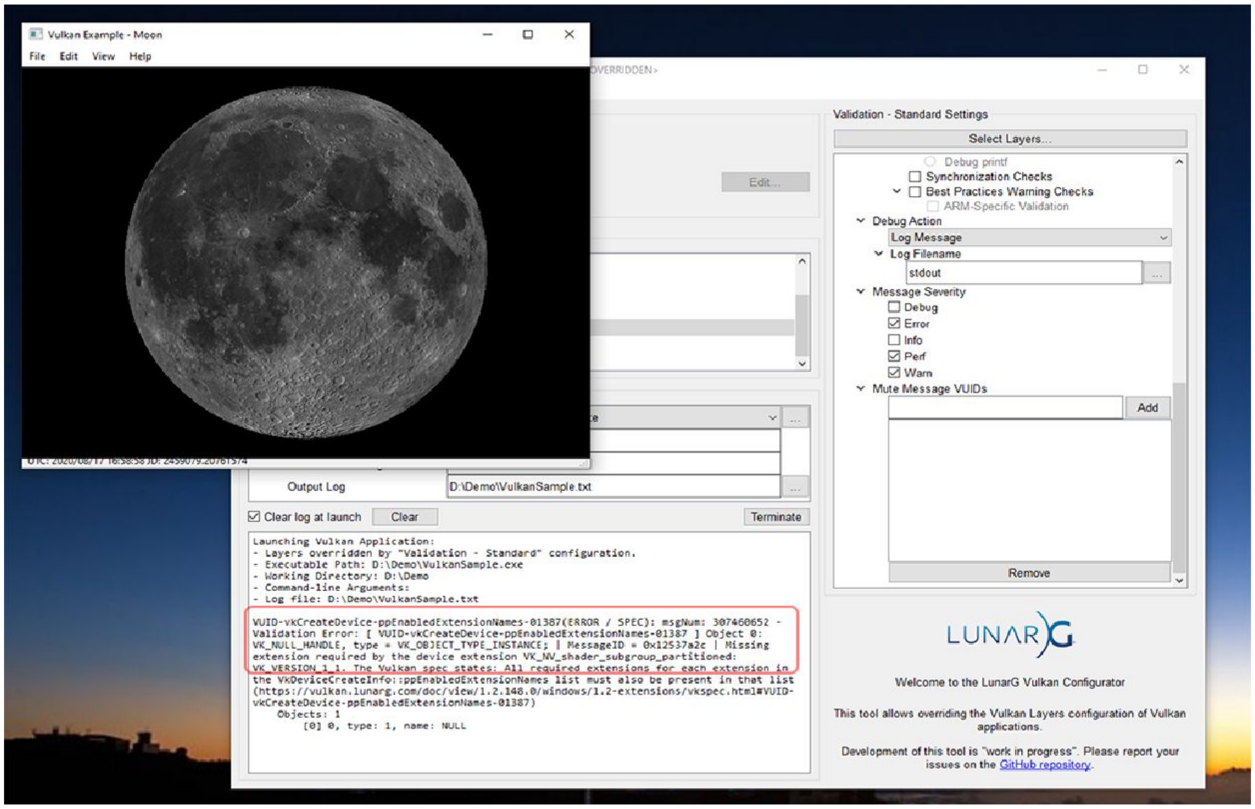
This tool allows overriding the Vulkan Layers configuration of Vulkan applications.

Development of this tool is "work in progress". Please report your issues on the GitHub repository.

Figure 5. Presets for the Khronos Validation layer are available.

Another new feature making its debut is VUID filtering. This capability is part of the Khronos Validation Layer, and allows you to mute any layer output based on a message's VUID.

Let's take for example my amazing Vulkan "I'm tired of cubes" project, the great textured sphere. Let's add the project to our application launcher and fire it off using the Standard Validation configuration.



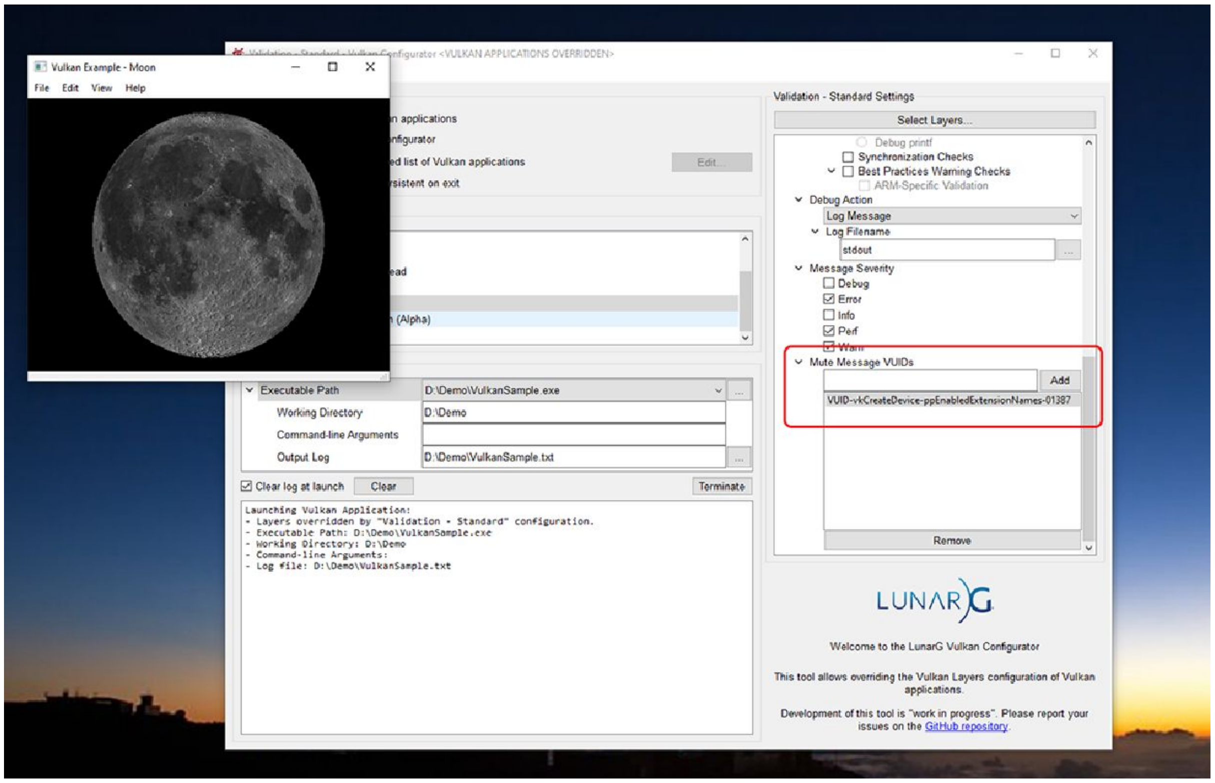
```
Launching Vulkan Application:
- Layers overridden by "Validation - Standard" configuration.
- Executable Path: D:\Demo\VulkanSample.exe
- Working Directory: D:\Demo
- Command-line Arguments:
- Log file: D:\Demo\VulkanSample.txt

VUID-vkCreateDevice-ppEnabledExtensionNames-01387(ERROR / SPEC): msgNum: 307460652 -
Validation Error: [ VUID-vkCreateDevice-ppEnabledExtensionNames-01387 ] Object 0:
VK_NULL_HANDLE, type = VK_OBJECT_TYPE_INSTANCE; | MessageID = 0x12537a2c | Missing
extension required by the device extension VK_NV_shader_subgroup_partitioned:
VK_VERSION_1_1. The Vulkan spec states: All required extensions for each extension in
the VkDeviceCreateInfo::ppEnabledExtensionNames list must also be present in that list
(https://vulkan.lunarg.com/doc/view/1.2.148.0/windows/1.2-extensions/vkspec.html#VUID-
vkCreateDevice-ppEnabledExtensionNames-01387)
    Objects: 1
        [0] 0, type: 1, name: NULL
```

Figure 6. There are often going to be validation warnings that we wish to get out of our way.

I decided to just enable all the detected extensions when creating my Vulkan instance, because I know that I'll need every single one of them without exception at some point… right? Okay, not really, but for illustration purposes it is not going to be uncommon for you to get warnings in your own projects that you have your own reasons for understanding and letting them slide. You also don't want a great number of warnings that you are ignoring because something important may show up as your project progresses, and you could miss it in all the "noise."

On the right-hand side at the end of the Khronos Validation Layers settings you'll find a "Mute Message VUID's" box. You can use the autocomplete feature to type in the VUID you wish to silence, or just copy and paste the whole VUID identifier into the edit control. Now on subsequent runs you'll find that particular message has been muted.



Validation - Standard Settings

- Debug printf
- [ ] Synchronization Checks
- [ ] Best Practices Warning Checks
  - [ ] ARM-Specific Validation
- Debug Action: Log Message
  - Log Filename: stdout
- Message Severity
  - [ ] Debug
  - [✓] Error
  - [ ] Info
  - [✓] Perf
  - [✓] Warn
- Mute Message VUIDs
  - VUID-vkCreateDevice-ppEnabledExtensionNames-01387

Executable Path	D:\Demo\VulkanSample.exe
Working Directory	D:\Demo
Command-line Arguments	
Output Log	D:\Demo\VulkanSample.txt

[✓] Clear log at launch

```
Launching Vulkan Application:
- Layers overridden by "Validation - Standard" configuration.
- Executable Path: D:\Demo\VulkanSample.exe
- Working Directory: D:\Demo
- Command-line Arguments:
- Log file: D:\Demo\VulkanSample.txt
```

Welcome to the LunarG Vulkan Configurator

This tool allows overriding the Vulkan Layers configuration of Vulkan applications.

Development of this tool is "work in progress". Please report your issues on the GitHub repository.

Figure 7. Tada! Now, the warning is out of my way and I can more easily spot anything new that comes along.

What’s next for the Vulkan Configurator?

We started our vkconfig redesign from scratch with the future in mind -- making the interface easier and more intuitive, and added some very powerful and time saving features along the way. Hopefully this tool will save you a lot of time, and be a great resource for your Vulkan projects.

We plan to enhance vkconfig moving forward with better integration and support for 3rd party tools and layers, and hope it will become a central tool for developing Vulkan applications by making layer management much easier. If you’re developing custom layers yourself, we want this to be a great way to expose that layer’s features and capabilities to other developers.

Please contact LunarG to let us know what you think of the new Vulkan Configurator or to offer suggestions for future releases. The best way to reach us is via our GitHub repository using the link shown below:

https://github.com/LunarG/VulkanTools/issues

Check out the new Vulkan Configurator Demo!

Check out our short demo of the new Vulkan Configurator that is now available on our LunarG YouTube channel.