Vulkanised 2026

The 8th Vulkan Developer Conference
San Diego, USA | February 9-11, 2026

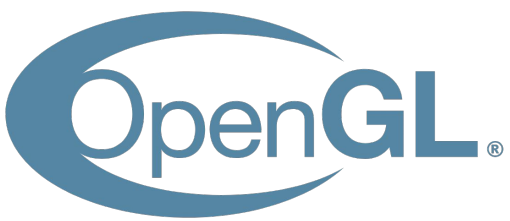
Vulkan Now and Then (for Hobbyists)

Sascha Willems, Independent

Vulkan as a hobby developer?

- Why not!
- Realtime graphics might be a niche
- But it it’s an interesting field
- Both technical and creative
- Games the reason for many to get into graphics programming
- Often starts as a hobby (out of curiosity)
- Learning a low-level graphics api can help landing a professional job
- **And the hobbyist of today might be the professional of tomorrow**
- …so we shouldn’t neglect that audience
- ***Spoiler***
  - We don’t😉
  - Wanting to use Vulkan for your hobby is fine (more than ever in 2026)

Graphics API landscape 2026



Vulkan in 2016

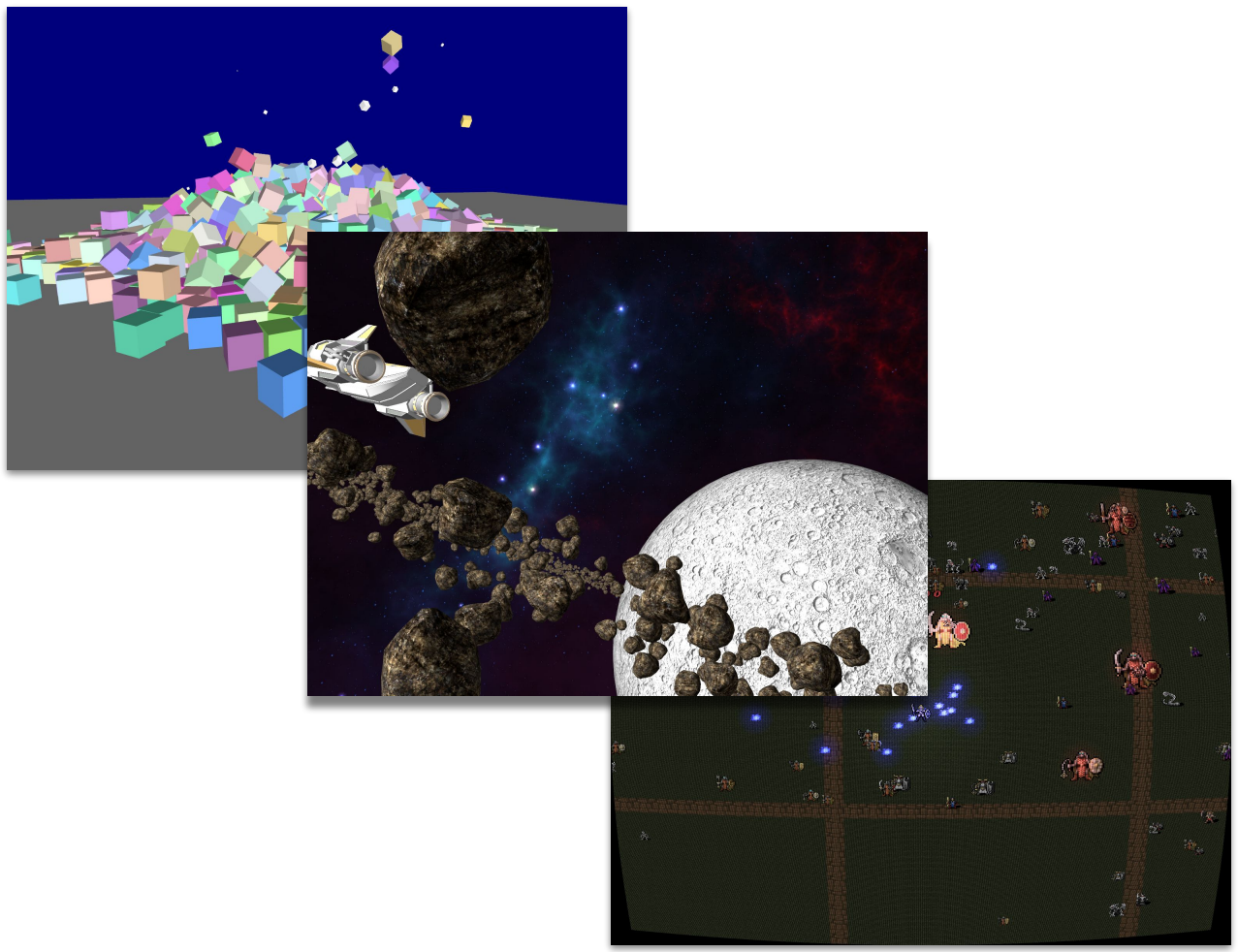
- Vulkan 1.0 was a compromise
- (Some) People expected an evolution of OpenGL (glNext)
  - But got something that felt more like a hardware abstraction layer
- Very little learning material
- No helper libraries
- Limited tooling support
- Varying driver quality
- Result: Off-putting to (not only) hobbyists
  - “I just want to draw something on screen, why is this so complicated!”
  - “I have converted my GL renderer to Vulkan and it isn’t any faster!”
  - “None of my GL knowledge works here”
  - “My stuff randomly crashes on other people’s system”
  - “Screw this, I’ll go back to GL or DX11…”

Vulkan in 2026

- 10 Years of continuous improvements
- Vulkan 1.3 as a pivotal release
  - “This is what Vulkan initially should’ve been”
- Lots of ecosystem improvements
  - Helper libraries (VMA, Volk, vk-bootstrap)
  - New shading language options (Slang, HLSL, etc.)
  - Improved tooling (Vulkan SDK, RenderDoc, vendor profilers)
  - Wide library support (SDL, SFML, GLFW)
  - Vulkan Profiles (and toolset)
  - Active communities (Reddit, Discord)
  - Vulkan support on mobile improving
- Lots of learning material and documentation aimed at developers
- **Worth giving Vulkan a second chance**

Vulkan Hobby Starter Pack

- Vulkan 1.3
- Vulkan memory allocator
- volk Meta Loader
- SDL, SFML, GLFW
- GLM math library
- Dear ImGui
- Slang for shaders
- ktx-software for Textures
- tinyglTF/tinyObj for 3D models
- Jolt for Physics
- Vulkan SDK
- RenderDoc for debugging



How to Vulkan in 2026

- Lack of easy to follow Tutorials
- Written from scratch
- Show how to use Vulkan for rasterization in 2026
- Vulkan 1.3 as a baseline
- Use common libraries (SDL, VMA, Volk, GLM)
- Do more than just a single colored triangle
- Doable in a single day
- Use C headers and very little “modern” C++ (designated initializers)
- Slang for shaders
- Incorporate (my) Vulkan experience
- Lots of tips and background info and pointers to additional resources
- < 700 LoC for rendering multiple textured 3D meshes
- See www.howtovulkan.com

Closing words

- Vulkan has evolved a lot
- But still feels like a full-time job
- Keeping up with changes still a challenge
  - More than 400 device extensions (incl. 15 maintenance extensions)
    - Some are very niche and only work on few devices
    - **Hard to judge when or if to use one of these**
- Making sure your stuff works everywhere isn't easy
  - Not everyone can afford hardware from all vendors/platforms
  - Validation layers help, but not a 100% guarantee
  - Driver stability still an issue, even on desktop
- Asking AI for help can be problematic
- But using it *can* be very rewarding

Links to get you started

- 🎓Learning
  - Vulkan Documentation
  - Simple Game Engine part of the Khronos Tutorial
  - Vulkan Guide
  - How to Vulkan in 2026
  - Learn at Vulkan.org
- 🤝Communities
  - Vulkan Reddit
  - Vulkan Discord

Thanks for your attention

The Joy of Vulkan