WebGPU

Compute kernels in your browser

Dániel Berényi

GPU Day 2025

Graphics technologies on the web

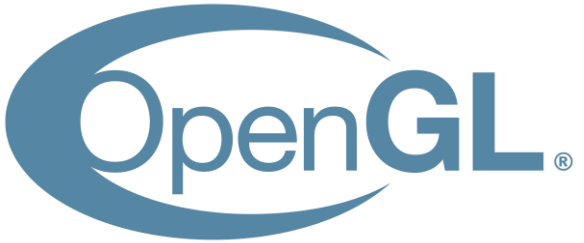
- Images (raster, vector)
- CSS ~~tricks~~ hacks
- HTML 5 canvas (2D drawing commands)

- WebGL and WebGL 2 for 3D

A long list of attempts that did not really gain traction.

(VRML/X3D, O3D, WebCL, OpenVG, …)

Why do we need another API on the web?



Since 1993…

We already have WebGL and WebGL 2…
but, they:

… are built on the legacy OpenGL abstraction
implicit state machine

… only support vertex and fragment shaders, nothing else

(so stuck at features from 15+ years ago...)

So, we have the same problem as desktop graphics had 10 years ago:

- Driver overhead, cannot support newer hw features

The solution is also the same: we need to go to lower-levels

WebGPU

WebGPU recognizes that we need to address modern requirements:

- Games, Advanced Visualization, AI
  should use the mobile GPU better to improve energy efficiency
- This requires lower-level access, more modern API model
- AI and other neural workloads cry for general compute capabilities
- Need to squeeze this into a web-first framework (stability, security)

WebGPU

Yes, but how low-level? How this is going to be implemented?

Currently developed native, low-level graphics technologies:

- Windows: DirectX 12
- Apple: Metal
- Everything non-Apple: Vulkan

They are very similar; it is entirely possible to abstract them reasonably well!

WebGPU

WebGPU hits a middle ground between the complexities of WebGL (OpenGL) and DX12/Metal/Vulkan:

- It hides some of the very low-level synchronization-, queue-, memory management and presentation details
- It exposes the pipeline object that explicitly manages the states connected to the rendering setup
- It exposes the pre-construction of commands to decrease CPU usage at execution time

WebGPU

Ballpark numbers for Lines-of-Code for illustration:

Task	WebGL	WebGPU	Vulkan
Draw a simple triangle	≈120	≈150	≈1000
Vector add computation	N/A	≈150	≈470

"I'm not developing for the web, why should I care?"

Getting some cross platform graphics working is notoriously hard

- You either learn at least 2 different APIs or
- Use some very large and complicated frameworks

However, WebGPU has bindings for the desktop too!

- You can link to it as a library
- And then suddenly you have a graphics solution working everywhere!

WebGPU has bindings for the following languages:

- JavaScript on the web
- C++, rust on the desktop

You can compile C++ to JavaScript with emscripten that supports WebGPU!

"I'm not developing for the web, why should I care?"

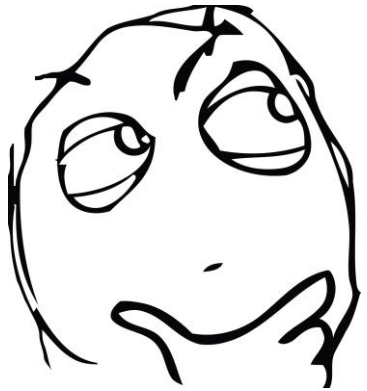
Furthermore:

Today, there is no good entry point to learn modern 3D graphics APIs

- starting with any of the low-level APIs is extremely challenging
- starting with OpenGL requires you to unlearn many things later

WebGPU solves that by hitting a middle ground

Also, it requires 0 setup: it *just works*™ everywhere,
all you need is a browser and a text editor to get started



"I do work with web technologies, whats new in it for me?"

- Compute Shaders
- Faster rendering
- More GPU features compared to WebGL
- Active development

Challenges on the web

There is no free lunch however, not everything is as easy, as with native technologies:

- Security: undefined behaviour, invalid memory accesses, sandboxing: these are very hard with GPUs
- Security is a trade-off against performance, and the web *does* take security seriously
- Much wider scope of devices than with native, large variety in capabilities and driver readiness thus, heavy compromises in features
- Privacy: limit fingerprinting opportunities

"I'm a scientist, should I care?"

- No double precision support yet, not planned for 1.0 :(

However:

- More efficient visualization of millions of datapoints
- Draw very complex geometries in a platform independent way
  -> scientific / interactive visualization for large data
- Added value in educational / later AR /VR settings
- Nice particle simulations,
  advanced client side data processing ☺

WebGPU – The API

At the surface it is very similar to other GPU APIs:

- A device that represents the physical GPU
  - Has a command queue to issue operations to
- Buffers, Textures for data storage
- Shader objects from text files (WGSL) to represent GPU shaders/kernels
- Most "new" things are around pipelines, resource bindings and command encoders

WGSL

WebGPU decided on its own shading language: WGSL

- Portable across all platforms(!), plain text format
- It is a bit rust like
- Somewhat restricted compared to other shading languages, but functionally similar
- You need to always explicitly manage the bindings compared to GLSL

WGSL – vector add


```
struct Vector
{ // Comma is the separator, the last one is optional
    data: array<f32>
};

@group(0) @binding(0) var<storage, read>       A : Vector;
@group(0) @binding(1) var<storage, read>       B : Vector;
@group(0) @binding(2) var<storage, read_write> C : Vector;

@compute @workgroup_size(128)
fn main(@builtin(global_invocation_id) idx : vec3<u32>)
{
  let i = idx.x;
  if (i >= 1024) { return; }
  C.data[i] = A.data[i] + B.data[i];
}
```

"So, is it really heaven on Earth?"

Not yet

- The specification is not yet finalized, changes are still happening
- It is in a W3C candidate recommendation status
- Support:
  Chrome and chromium-based browsers enabled support for WebGPU
- but other browsers have only experimental support
- Examples, explainers, docs are somewhat rare or out of date,
  it is a new technology with all its shortcomings

Will it stay?

The question is reasonable; many attempts have failed before.

- We think so, because of large demand for AI and games
- Otherwise, deploying a GPU accelerated neural network / LLM solution is a nightmare
- Google is pushing this for a very long time.
- WebGPU is already getting adopted in many places!

Recommended reading

- Mozilla WebGPU API Docs
- WebGPU Specification
- WebGPU Shading Langauge Specification

Some Demos at:
https://gpgpu.hu/GPUDay2025

Other useful links:

- https://webgpu.rocks/
- https://gpuweb.github.io/gpuweb/explainer/
- https://webgpufundamentals.org/webgpu/lessons/webgpu-from-webgl.html
- https://github.com/beaufortfrancois/webgpu-cross-platform-app

Hungarian GPGPU Community on Discord

- Meeting place for all who are involved in or would like to learn more about GPGPU technologies
- All fields: Graphics, Compute, Vision, …
- All APIs: CUDA, OpenCL, Vulkan, WebGPU, …
- All vendors: AMD, Intel, NVIDIA, …
- All levels: Students, Teachers, Professionals, Scientists, …
- All topics: Programming, Hardware, Bugs, Optimizations, Teaching, Trends, Events, News, …

Magyar GPGPU Közösség
Discord szerver
https://discord.gg/eX8uHsmUwd