Intel’s Updates on Web Graphics

Yang Gu
Intel Advanced Web Technology Team
August 15, 2018 (SIGGRAPH 2018) - July 31, 2019 (SIGGRAPH 2019)
About Team

● History
  ○ Nov 2011: Began to focus on web
  ○ Nov 2012: Began to work on Chrome, mostly on performance tuning
  ○ Nov 2014 - Jan 2017: Implemented WebGL 2.0 in Chrome
  ○ Jan 2017 - now: WebGL 2.0 Compute
  ○ July 2018: Began to contribute to WebGPU
  ○ Contributed 1700+ patches to Chrome and related projects

● Now
  ○ Intel rep of Khronos WebGL Working Group
  ○ Intel rep of W3C WebGPU Community Group
Quality Gate Keeper on Intel Platforms

- Conformance
  - WebGL CTS has been integrated with Windows driver CI since Sep 2018
  - WebGL CTS has been integrated with Intel Mesa driver CI since Feb 2019
  - Now, any changes to Intel’s graphics driver that affect or break WebGL conformance are caught before they ship, even before they are checked in!
  - Thanks the great support from Google Chrome team on test framework improvements

- Performance
  - Monthly performance test with most recent hardware, OSes, browsers and selected benchmarks
WebGL 2.0 Compute

- **Desc**
  - Bring OpenGL ES 3.1 Compute Shader to web

- **Status**
  - Started the effort at the beginning of 2017
  - Draft specification is online
  - Native dEQP tests have 100% pass rate on both OpenGL and D3D backends
  - Available in Chrome on Windows and Linux
    - Windows: --use-cmd-decoder=passthrough --enable-webgl2-compute-context
    - Linux: --use-cmd-decoder=passthrough --enable-webgl2-compute-context --use-gl=angle
  - Emscripten support to convert native apps to WebGL 2.0 Compute is under review
  - Cool WebGL 2.0 Compute Samples
    - TFJS WebGL 2.0 Compute backend by us (Just a fork, not in TFJS repo, WIP)
    - Matrix multiplication in WebGL2-compute by Evgeny Demidov
    - WebGL 2.0 Compute Demos by 9ballsynrome

- **Plan**
  - TFJS backend, with more critical features like subgroup and FP16
  - Expose more features via Chrome
  - Port native dEQP OpenGL ES 3.1 Compute Shader cases to web
Parallel Shader Compile

- **Desc**
  - Implement KHR_parallel_shader_compile extension so that shaders can be compiled in background and in parallel, to mitigate long-standing pain about slow shader compiles

- **Status**
  - **Specification** is in Community Approved stage
  - Available in Chrome on Windows and Linux
    - Windows: no flag is needed
    - Linux: --use-cmd-decoder=passthrough --use-gl=angle
    - Main thread is no longer blocked by shader compiles, so that other tasks can fit in to hide the latency.
    - Shaders can be compiled in parallel, and more performance boost can be seen with more CPU threads.
      - Some examples showed improvements of 35% (2 threads), 55% (4 threads) and 63% (8 threads) on average.
  - Babylon.js and Three.js already have the integration
  - A performance workload can be found in Khronos WebGL repo.

- **Try it out today!**
WEBGL_video_texture

- **Desc**
  - A new extension to support zero-copy video uploads into WebGL textures

- **Status**
  - Extension Spec draft
  - First Implementation is already in Chrome
  - Seeing up to 47% speedups on some contents

- **Plan**
  - Complete the feature
  - Mature the spec
Aquarium

- **Desc**
  - Measure the performance diff between WebGL and native
  - Drive the web platform optimizations by minimizing general overheads (like JS engine, security and D3D conversion) and proposing new features

- **Status**
  - Project became open source a week ago!
  - Already supported various backends, including OpenGL, D3D12, ANGLE and Dawn
  - Many options, like GPU selection, specific features, window mode and so on, can be easily toggled for performance comparison
  - Worked closely with ANGLE and Dawn developers
  - Helped to find many missing features and issues in ANGLE and Dawn

- **Plan**
  - Port to more backends
    - Web: WebGL 2.0, WebGPU
    - Native: D3D11, Vulkan, Metal
  - Experiment new features that may be proposed to web
Questions or Suggestions?

Email: Yang Gu (yang.gu@intel.com)