WebGL and WebGPU Updates

On Behalf of the WebGL WG and WebGPU CG
WebGL+WebGPU Meetup, October 2021

Agenda

Join WebGL & WebGPU Communities

WebGL 2.0 in Safari Updates

Upcoming WebGL Extensions

WebGPU Updates

WebGPU Origin Trial

WebGPU Samples

Presentations

Join WebGL & WebGPU Communities

- The WebGL and WebGPU APIs are supported by vibrant online communities!
- If you're developing with these APIs, we would like to hear from you!
- On the WebGL side:
  - Please consider joining the WebGL Dev List: announcements of products, demos, new tools, job postings, questions, discussions - all are welcome!
  - Khronos' public_webgl mailing list hosts lower-traffic announcements
  - The WebGL Matrix chatroom offers a way to talk with browser implementers and other developers
  - You can find a lot of cool stuff by searching #webgl on Twitter 😂

Join WebGL & WebGPU Communities

- On the WebGPU side:
  - If you have feedback on the API, please see the main WebGPU repository for options to communicate it to the community group
  - The WebGPU Matrix chatroom also offers a way to talk with browser implementers and other developers
  - There's an increasing amount of cool stuff showing up on #webgpu on Twitter 😊
- We all look forward to hearing from you!

WebGL 2.0 In Safari Updates

- WebGL 2.0 has shipped in Safari 15 on both macOS and iOS! 🎉🎉🎉
- The culmination of a 2+ year journey started in June 2019
- Many positive outcomes from this joint project

WebGL 2.0 In Safari Updates

- WebGL 2.0 can now be considered universally available across browsers, operating systems and devices
- As an application author, you can target WebGL 2.0 with confidence
- WebGL 2.0 has resolved many corner cases and behavioral differences compared to the combination of WebGL 1.0 + its many extensions
- We encourage you to migrate to WebGL 2.0
- It's no longer necessary to maintain a WebGL 1.0 fallback path unless you need to reach absolutely every device
    - In particular, older Windows machines and Android devices

WebGL 2.0 In Safari Updates

- Apple has adopted ANGLE as the basis for Safari's WebGL implementation
- Apple's team made dramatic contributions to ANGLE's Metal backend over a period of 1+ year
- Safari 15 runs WebGL on top of Metal on recent iOS and macOS devices
- Apple and Google engineering teams are collaborating on:
  - Upstreaming Apple's work to the ANGLE repository
  - Passing the underlying OpenGL ES 2.0 and 3.0 conformance tests which affect WebGL
  - Addressing key functionality issues
  - Adopting top-of-tree ANGLE into WebKit, to have a common codebase for development going forward
  - Switching Chrome to use ANGLE's Metal backend
- As always, file any bugs you see in WebGL in Safari 15 on bugs.webkit.org, component "WebGL"
- (For other browsers' bugs, consult "How to get a WebGL Implementation")

Upcoming WebGL Extensions

OES_draw_buffers_indexed

- Enhances multiple draw buffer functionality
- This extension provides the ability to:
  - enable or disable blending
  - set the blend equations
  - set the blend functions
  - set the color write masks
  - all per color output!
- This extension was specifically requested by the 3D Formats working group to implement advanced materials (e.g., that use dual depth peeling) more efficiently
- Can be tested in Chrome today by enabling WebGL Draft Extensions in `about:flags`
  - Please file any bugs on crbug.com
- Will come to all browsers shortly after community approval

Upcoming WebGL Extensions

Base Vertex/Base Instance & Multi-Draw Variation

- Provide control of BaseVertex, for indexed draw calls, and BaseInstance, for instanced draw calls
- Multi-draw variants are provided as well
- Allow reuse of index buffers to draw multiple different geometries from the same set of vertex buffers
- Reduce CPU and memory overhead in certain scenarios
- If you've needed these draw parameters, please try the extensions and provide your feedback
- Can be tested in Chrome today by enabling WebGL Draft Extensions in `about:flags`
  - Please file any bugs on crbug.com
- Will come to all browsers shortly after community approval

WebGPU Updates

- Specification discussions among browser implementers are converging
- Aiming for a 1.0 version of the specification early in 2022
- Today, you can try the API, and all of the graphics & compute functionality it offers, in multiple browsers
- Chrome:
  - In Canary, enable unsafe WebGPU in `about:flags`
- Firefox:
  - In Nightly, set `dom.webgpu.enabled` in `about:config`
- These are intended for local development
- Do not browse the open web with these flags enabled
- Keep in mind that implementations are still evolving quickly
- Content may not be portable among browsers yet
- Suggest always targeting the latest version of the WebGPU specification in your own applications, and polyfill if you find your desired browser hasn't caught up

WebGPU Origin Trial

- The WebGPU specification has advanced to the point where it's ready for broader testing!
- If you're developing a WebGPU application, you can now publish it in a way that users can access it without needing to enable flags in their browser - via a Chrome Origin Trial
- Instructions are in this web.dev article
- The Origin Trial is running from now (Chrome 94) to Chrome 97 (~Jan 2022)
- Please keep in mind that the API **will change incompatibly** during the Origin Trial - by design
  - Only publish content, and advertise it, if you plan to keep it up to date

WebGPU Samples

Several WebGPU samples have been published to help you get started learning both the API and shading language (WGSL)!

- Austin Eng's WebGPU Samples (Github)
  - Journeys from your first triangle, to real-world compute & graphics examples
- Brandon Jones' Clustered Shading and Metaballs (Github) demos
  - Real-world usage of the WebGPU API and compute shaders
- Shrek Shao's WebGPU Deferred Renderer
  - Shows how to do deferred shading in WebGPU, complete with debug views

Additionally, Babylon.js and Three.js have WebGPU renderers well underway!

We'll hear more details from the Babylon.js team later in this Meetup!

Presentations

Great group of presenters today!

- Ivan Popelyshev, Crazy Panda: stroked lines in WebGL with top performance (no MSAA) and quality equal to 2D Canvas
- Thomas Lucchini, Microsoft: journey of porting the Babylon.js engine to WebGPU, including several demos
- Tam Belayneh, Esri: pushing 3D geospatial boundaries with WebGL

Feel free to type your questions into the Q&A panel at any time!

We'll answer them live at the end of the session.