WebGL and WebGPU Updates

Ken Russell and Corentin Wallez, Google
On Behalf of the WebGL WG and WebGPU CG
WebGL+WebGPU Meetup, July 12, 2022

Agenda

Join WebGL & WebGPU Communities

WebGL Updates

- Firefox, WebKit and Chromium Updates
- WebGL 2.0 In Safari / Chrome Updates
- Conformance Improvements
- New & Upcoming WebGL Extensions

WebGPU Updates

- Standardization
- Implementations
- Resources and Contributing

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 spec announcements
  - The WebGL Matrix chat room 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 chat room also offers a great way to talk directly 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!

Firefox WebGL Updates

- Some progress implementing WebGL extensions, in particular OES_draw_buffers_indexed
- display-p3 support for WebGL canvases is implemented on macOS and Windows

WebKit WebGL Updates

- Several more extensions hooked up, including EXT_texture_norm16, EXT_texture_compression_bptc, and OES_draw_buffers_indexed
- Base Vertex/Base Instance & Multi-Draw Variation available behind a flag
- Progress on GPU process implementation
- Cleanup and bug fixing in extensions-related code, automatic clearing of back buffer after presentation, transform feedback, and Canvas developer tools
  - Thanks to WebGL WG member Alexey Knyazev for these, and for exposing the extensions mentioned above

Chromium WebGL Updates

- Work on WebGL extensions, in particular a performance test with Ivan Popelyshev to motivate base_vertex_base_instance/multi
- Working on shipment of ANGLE's Metal backend on macOS
  - Ongoing collaboration with Apple, who have fixed several key bugs
  - Working on dual-GPU support in Chrome with the Metal backend
  - Raised several limits which were lower than in the OpenGL driver
  - Diagnosing and addressing GPU resource consumption and performance issues
- display-p3 support per Canvas Color Space Proposal
  - Thanks to Chris Cameron
- Customer reported issues

WebGL 2.0 In Safari / Chrome Updates

- You can test Chrome on top of ANGLE's Metal backend today:
  - Go to `about:flags`
  - Set "Choose ANGLE graphics backend" to "Metal"
  - (please test it in Chrome Canary - fixes are actively going in)
- and compare its behavior to the OpenGL backend
- Some regressions exist, like low-power/high-performance GPU selection not working yet
- Please file any bugs you find with the Metal backend on ANGLE's issue tracker
- Please test Safari Technology Preview and file any bugs you see in WebGL on bugs.webkit.org, component "WebGL"
- (For other browsers' bugs, consult "How to get a WebGL Implementation")

WebGL 2.0 In Safari / Chrome 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

Conformance Improvements

- Many new tests are flowing into Khronos' WebGL repository
- Discovered through many means - code inspection, investigation of failed tests, work on ANGLE's Metal backend, and others
- Thanks in particular to Gregg Tavares (Google) and Alexey Knyazev (Independent) for these improvements
- These will enable a more portable WebGL 2.0 ecosystem across browsers

New 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
- Shipping in Chrome; coming to other browsers
  - Please file any bugs on crbug.com, WebGL component

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
  - Have customer feedback that these can improve performance as well
- If you've needed these draw parameters, please try the extensions and provide your feedback
- Can be tested in Chrome and WebKit Nightly 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

EXT_provoking_vertex

- When using the `flat` interpolation qualifier in shaders, it's necessary to precisely define the initiating, or provoking, vertex for a given primitive like a triangle or line
- Unextended OpenGL ES 3.0 / WebGL 2.0 require OpenGL's convention of the "last" vertex as the provoking vertex
- Emulating this is expensive on some platforms - Direct3D and Metal especially
- This extension allows the provoking vertex to be configured by the application, specifically to the first vertex
- Significantly improves performance of flat shaded geometry on some platforms
- Extension is in draft form now; expect rapid progress to community approval

WebGPU

An upcoming "modern" graphics API for the Web:

- A successor to WebGL, not a replacement.
- Compute shaders on the Web!
- Lower overhead API
- Foundation for future features (bindless, ray tracing, multithreading …)

Development happens on GitHub and at the W3C

- Anybody can join and participate in the development.
- Thanks to Khronos for hosting us here!

WebGPU standardization updates

Standardization of v1.0 is reaching completion. Blockers are polishing the spec and reaching a decent amount of conformance testing. V1.0 2022Q3 hopefully, maybe Q4.

WGSL standardization is in the polishing phase:

- Addition of shorthand vector types `vec4f vec2i vec3u` etc.
- Improvements to the uniformity analysis based on developer feedback.
- Lots and lots of clarifications (semantic of divides by zero, writes of structure with padding, …)

API standardization is also in the polishing phase:

- WebGPU `<canvas>` API is now complete (with color-space, resizing, etc)
- Addition of `GPUAdapterInfo` to query the GPU vendor / architecture (under control of the browser for privacy)
- Reflection of creation parameters for container objects.
- Lots of polish as well :)

WebGPU - Implementation status

Firefox

- In Nightly set dom.webgpu.enabled to true in about:config
- Not yet suitable for browsing securely with this flag enabled.

Chromium

- Windows, ChromeOS and Mac (Linux and Android later)
- The WebGPU Origin Trial allows publishing WebGPU apps on the Web today!
  - Until M105, a request to extend the Origin Trial again is expected.
  - Breaking API / shading languages are happening by design. You must fix warnings surfaced in the devtools.
  - web.dev/gpu
- Aiming for release close to the v1.0 release of the standard this year.

Implementations are mostly interoperable already!

WebGPU - Uniformity analysis feedback

Uniformity analysis ensures it can prove gradients and barriers are done in uniform control flow.

Previously the semantic of **discard** was that invocations are terminated.

Because some invocations may be terminated inside the if, the control flow is non-uniform after the if.

```
// In uniform control flow

// non    - uniform value
var albedo = sampleAlbedo();

if albedo.a < 0.5 {
        // Non    - uniform control flow
        discard;
}

// Non    - uniform control flow

//    Error!      Implicit gradient in texture
// sampling is not allowed here.
sampleSpecular();
```

WebGPU - Uniformity analysis feedback

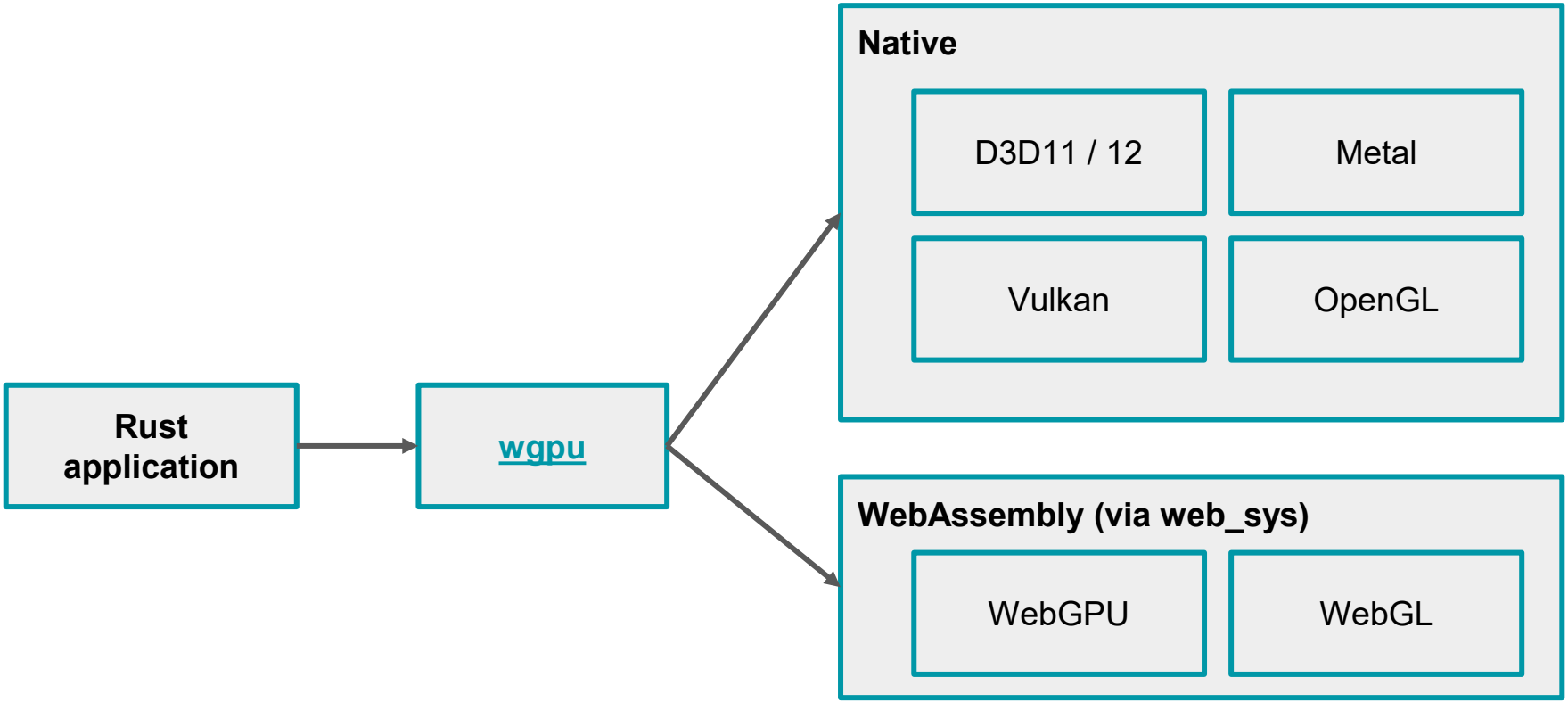
Feedback was that it was almost impossible to make uniformity happy for alpha testing.

The semantic of **discard** was changed to be "demote to helper", invocations that still execute but don't have side effects.

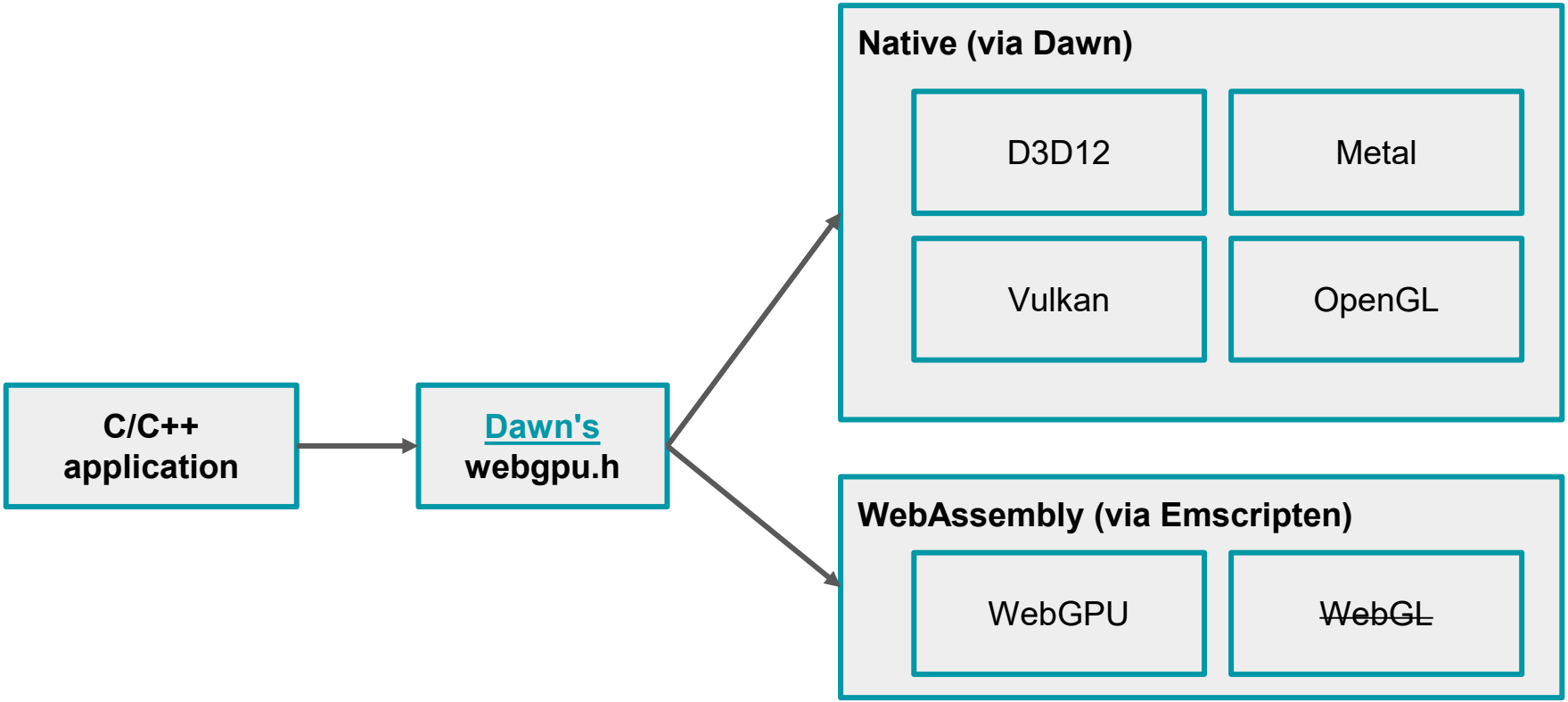
Alpha testing written as usual is now valid for the uniformity analysis!

```
// In uniform control flow

// non    - uniform value
var albedo = sampleAlbedo();

if albedo.a < 0.5 {
        // Non    - uniform control flow
        discard;
}

//     Still in uniform control flow

// This is now valid, yay!
sampleSpecular();
```

WebGPU - WebAssembly ecosystem (wgpu)



WebGPU - WebAssembly ecosystem (Dawn)



WebGPU - Using it in JS without a browser!

Many reasons to use WebGPU outside of a browser:

- Automated testing.
- Offline rendering using the same tech stack.
- "Native" frameworks like Electron, BabylonNative, etc.

Deno:

- Deno is a Javascript runtime with built-in WebGPU support
- Uses wgpu under the hood.

Node.js

- Dawn has a dawn.node Node.js module.
- In a WIP but fairly good state (99% on par with Chromium for tests)

WebGPU - Chromium partnerships

Steady progress on WebGPU backends for popular web 3D libraries

- Three.js, Babylon.js

Ongoing partnerships with teams including Intel, TensorFlow.js, Google Meet, MediaPipe, and more

PlayCanvas has undertaken a major refactor of their engine in support of WebGPU

- Tracking bug: https://github.com/playcanvas/engine/issues/3986
- Nearly ready for beta testing - follow this bug if you'd like to test

WebGPU - Resources

Tutorials:

- Get started with GPU Compute on the web by Francois
- WebGPU - All of the cores, none of the canvas by Surma
- Raw WebGPU by Alain
- WebGPU Best Practices by Brandon

Samples

- The WebGPU Samples repo (Github) by Austin
- Clustered Shading, Metaballs, and Spookyball demos by Brandon
- WebGPU Deferred Renderer by Shrek

WebGPU - Contributing!

Many ways to engage!

- Try the API and provide feedback on any channel
- Publish sites using WebGPU using Chrome's WebGPU Origin Trial
  - Could use WebGPU support in popular frameworks like Three.js Babylon.js and TF.js
- Help with conformance testing
- Contribute sample / demos / articles using WebGPU
- Join the conversations on the Matrix chat!