GPU Web 2019-05-06

Chair: Corentin
Scribe: Ken
Location: Google Meet

TL;DR

- Agenda for the F2F
    - David will make a programming model agenda.
    - Feedback from early users of WebGPU
    - Google will talk about WebGPU at it's I/O conference.
        - Will direct users to give feedback on public-gpu
- Command buffer reuse
    - Concern that it could lead to non-portable performance.
    - Would be nice to have, will need investigation + proposal.
    - Will need two proposals: one for bundle style, one for reuse style.
- Clear operation API #276
    - Biggest concern is that clearing should be encouraged, which isn't the defaults there.
- PR burndown
    - #281 "default entryPoint to main" closed.
    - #268 default values for attachments, conflicts with #276
    - #277 move primitiveTopology to rasterization state, closed.

Tentative agenda

- Agenda for the F2F
- Early proposal
- Clear operation API - #276
- Agenda for next meeting

Attendance

- Apple
    - Dean Jackson
    - Justin Fan
    - Myles C. Maxfield
- Google
    - Austin Eng
    - Corentin Wallez

    - Dan Sinclair
    - Kai Ninomiya
    - Ken Russell
    - Shrek Shao
- Intel
    - Yunchao He
- Microsoft
    - Chas Boyd
    - Rafael Cintron
- Mozilla
    - Dzmitry Malyshau
- Mehmet Oguz Derin
- Timo de Kort

Agenda for the F2F

- CW: Programming model discussion
    - RC: is this an alias for “which shading language should the API ingest” or something else?
    - CW: something different. Constraints in terms of security, bit-ness, etc. on all languages WebGPU ingests. Memory model, kinds of images you can sample, etc. Clarifies what’s the difference between programming language issues and other things.
    - CW: David Neto will make an agenda for this part.
    - DS: we can ask him to.
- CW: have been partnering with some early customers to do WebGPU ports. Babylon.js is one. (One of the big 3D JS frameworks, open-source, developed by Microsoft.) Tried WebGPU and came up with a number of issues.
- CW: For example, GenerateMipmaps. (DZ smiling.)
    - DZ: suggest letting user libraries do this.
    - CW: maybe, but boilerplate everyone will need. We can find early pain points with this.
    - YH: Intel folks porting Aquarium to Dawn, also had GenerateMipmap issues.
- CW: RowPitch=256 seems arbitrary to people. More painful than WebGL. Now we have data and people actually complaining about this, and not completely free / easy to do. (Google was in favor of the 256 byte row pitch limitation, but now not sure we should require this or paper over it.)
- CW: we’ll talk about WebGPU at the Google I/O conference coming. Where should we solicit feedback? Register on public-gpu@ and give feedback there?
    - RC: didn’t we set up a partner mailing list?
    - CW: yes but nobody uses it.
    - DJ: guess the public list. They could also file bugs about specific issues.
    - MM: is there an intended audience of developers?

    - CW: will be developer-y people.
    - DM: we don't want them to file 1000 issues on webgpu. Maybe make a sub-project on the organization.
    - MM: really? WebKit, Chromium, etc have issue trackers
    - KR: this isn't going to be focused feedback yet. Will be pretty general.
    - CW: we probably don't want this open discussion on the issue tracker.
    - KN: would prefer to use the mailing list.
    - DJ: agree. People may have to join the list, but it's not that difficult.
    - CW: ok, we'll direct feedback to the mailing list.
    - MD: we could also use a Discourse forum.
    - DM: think it's a good idea. Often useful questions there. Rust uses it and it's useful.
    - KN: like the idea.
    - MM: if someone sets it up then we could publicize it.
    - DJ: when is the talk?
    - CW: will add a link. Thursday.
    - RC: W3C has its own Discourse server.
    - MM: think the WICG has its own server.
    - DM: sounds like there'll be a lot of feedback to discuss.
    - CW: could potentially cover a whole day, maybe not. Maybe half a day. Do you have early feedback?
    - DM: not from the web. Will ask around.
- CW: snapshotting, and a CTS. Didn't succeed on these last time around.
    - KN: we don't yet have a CTS that we can add tests to.

Early proposal

- CW: Command Buffer Reuse
- MM: would like to discuss topics from NVIDIA's DX12 Do's and Don'ts article
- DM: Texel buffer views
- CW: roundup of Github issues.
- KR: is it feasible to discuss performance issues in-browsers at this point?
    - CW: right, we found some perf issues with the Babylon.js port - JS bottlenecks. WASM could help. But command buffer reuse is intended to fix that.
- KN: reusable secondary command buffers?
- MM: Metal doesn't have this, discarded this due to non-portable performance. Has this changed?
    - KN: no. It's avoiding JS overhead in the construction of the same command buffer over and over.
    - CW: for example, the background of a scene might be constant.
    - MM: wouldn't this require all impls to not record at recording time?
    - CW: discussed with DM. Could split reusable command buffer in many chunks, then stitch together at submit time.

    - MM: right now our implementation, when the web app says “bind this bind group”, etc., we put it in a command buffer and we later submit. We don’t remember anything about it - no software mirror of the Metal command buffer. Would have to change if we want to support this idea.
    - CW: we could have a command buffer creation attribute “reusable”.
    - MM: might help. Impl burden - not sure how much that’ll be - and runtime burden of recording into 2 things.
    - CW: app would have to say, I want to reuse this command buffer. If no attribute, it can’t reuse the cmdbuf. If flag not set, WebKit can do what it’s doing now.
    - MM: true, but if it is set, then maybe the Metal impl falls off a perf cliff. Have to see.
    - MM: ultimately we should have some internal discussion before the F2F.
    - AE: MM is just saying they don’t serialize before submission, so they don’t know the perf implications entailed.
    - DM: you’d see unusually high submit time costs, which would be non-portable.
- MM: fundamentally we agree with you, would be great to have. Should open internal feature request with the Metal team to add this.
- DM: there’s a window of opportunity with D3D12-style bundles. Then user would say “use this bundle during recording”. No surprises during submission time; costs are during recording.
    - RC: will you use D3D12 bundles in your impl?
    - DM: talking about exposing the concept. Won’t see unusually slow submission.
    - KN: because you unpack the bundle during cmdbuf record time rather than submit. That’s what I was getting at; would prefer bundle style vs. reusable cmdbufs.
    - CW: almost the same.
    - MM: what’s the difference?
    - CW: in D3D12, you can set bind group, vertex buffer, and draw/dispatch. But you can’t SetRenderTarget, or copies.
    - DM: you have some sub-piece of work inside the rendering pass.
    - MM: didn’t WebGL just add something similar?
    - AE: it’s sort of like that but more flexible. More than just drawing.
    - CW: WebGL extension is transparent in that you write binary format to specify your commands, but the bundles are opaque.
    - KN: the WebGL version is literally just multiple draw calls.
    - MM: sounds like we have two competing proposals but neither is written down. Earlier suggested that Apple would like some time internally to discuss this. Maybe we can postpone the rest of this discussion.
    - CW: I’ll write something down, can’t promise before Friday. Idea before writing it was to be able to reuse whole command buffers but also “bundle” style things - e.g. containing both approaches.
    - DM: sounds like it should be two proposals. They barely interact.
    - MM: would probably pick one or the other. Think they compete.

    - CW: I can make it into two proposals.
- MM: Multiple queues?
    - DM: I'll remind JG about this, promised to put a proposal together.

Clear operation API - #276

- DM: the PR suggests an API that's most concise, but the problem is that it doesn't encourage people to use the Clear operation.
- CW: do we expect to have any more load ops?
- MM: moving this out of an enum makes it less extensible.
- CW: the only other one in native APIs is DONT_CARE, and we probably don't want to add that.
- KN: could imagine adding it if it weren't actually used.
- CW: Metal has tiled_image (for compute inside render passes). Does that use specific new load ops?
    - MM: don't think so.
- CW: the way DM wrote this it looks really nice for JS. Doesn't encourage people to think about it which is the biggest issue.
- MM: this patch also removes some default values, default clear color and stencil. Should I consider these two separate concepts or are they linked?
- DM: linked. If you want to clear then you have to specify the color.
- KN: previously if you didn't say clear color it was opaque back. Now if you don't it says "load".
- MM: the default values would encourage people to use the clear colors.
- KN: I like the thing I proposed, either string "LOAD" or the clear color, and the default could be a clear color value. Little funky type-wise but makes sense to me.
    - DJ: I like that too.
    - MM: I like it too but thought Dean didn't like it. :)
- DM: great, so there's consensus?
- RC: SGTM
- CW: SGTM too. Little tricky for a C API but we'll think of something.
- MM: so it'll be required, either pass a string or a color?
- KN: would like to keep default clear color.
- MM: optional then?
- CW: no, then the default does something entirely different. You render black and you don't know why.
- KN: good point.
- MM: the LoadOp was required before. Even before this change the programmer has to think about the loading behavior they want.
- KN: the thing I proposed, if you omit it it'll clear.
- CW: would be RGBA with zeros.
- KN: would like it to be [0, 0, 0, 1].
- MM: want to avoid micro-syntaxes.

- DJ: could also say “clear-black” or similar.
- KN: yes, could have shortcuts for common ones.
- CW: could you make a PR for this for the F2F? GpuLoadOp / GpuColor.
    - KN: yes, will add the special ones.
    - RC: so the one we’re agreeing on is the one Kai commented on ~10 days ago?
    - CW: yes, with some more stuff.
    - RC: not sure DM’s proposal can be done in JS.
    - KN: right, that can’t be done, would be the ideal but don’t know how to do it.
- CW: also ClearBlack should be ClearZero.

PR Burndown

- #281
    - MM: probably not good. People putting vertex / fragment shaders in same source code lets them share helper functions, speed up compilation. Not possible if they both need to have a default name of “main”.
    - CW: it’s sort of a GLSL-ism. People we’re working with have GLSL and expect their entry point to be “main”.
    - KN: think we should continue to require the entry point.
    - MM: can we hear from the other two stakeholders?
    - RC: fine requiring entry point.
    - DM: (nodded)
    -
- #268
    - CW: conflicts with the Clear ops we just discussed
- CW: everything else seems to be there.
- KN: I need to update the ImageBitmap one.
- #277
    - CW: I can close this - no longer think this is a good idea.

Agenda for next meeting

-
-
-
-