Porting Babylon.js for WebGPU

Alexis, @sebavanjs, @thomlucc
Origins

First experimentation in 2019
- First version of a WebGPU engine
- Render Bundle demo [Forest](#)

Babylon.js agnostic by design

Backward compatibility as a key objective

Main goal: no code change for users
Porting the beast

Ported 90% of the engine during 2020
• Render Target
• Postprocesses
• Shadows (incl. PCSS – Contact Hardening Shadows)
• Compressed Textures (KTX2)
• Stencil Buffer
• Effect Layers (glow, highlight etc….)
• And many more….

Merged to main branch in December 2020
• First version of the Babylon.js Playground

Learning: essential collab with teams designing and implementing WebGPU in browsers
Available today

Start to experiment with it in **Babylon.js Playground**

Via WebGPU flag in Chrome/Edge Canary

Via [Origin Trial](#) in Chrome/Edge

Learning: no simple migration from GLSL

- [ShaderMaterial](#) for new shaders in WGSL
- GLSLang/Tint/Naga for existing shaders in GLSL?
Simply update the engine creation

Replace

```javascript
const engine = new BABYLON.Engine(canvas);
```

By

```javascript
const engine = new BABYLON.WebGPUEngine(canvas);
await engine.initAsync();
```

(Initialization of WebGPU is asynchronous)

Breaking changes: [documentation](#)

Feedback: contact us on the [forum](#)
What’s new: **Compute Shaders**

WebGPU only feature (ported in early 2021)

Non-graphics parallel processing (blur, computer vision, simulations)

Demos: Image Blur, Compute Boids, Hydraulic Erosion, Slime Simulation, Ocean Simulation

Learnings: much easier to add a new feature than porting existing features 😊
## What’s new: Fast Path

Implemented a Fast Path with Render Bundle

### Snapshot Recording
- Records draw calls during one frame
- Replays for all subsequent frames

More than x10 perf increase
- For mostly-static scenes (no pipeline change) such as eCommerce
- Possibility to update snapshot (e.g. when adding meshes or adding transparency)

Learning: backward compatibility and performance

---

### WebGL

<table>
<thead>
<tr>
<th>Metric</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frame total</td>
<td>5.09 ms</td>
</tr>
<tr>
<td>Virtual fps</td>
<td>196 fps</td>
</tr>
</tbody>
</table>

### WebGPU – Compatibility Mode

<table>
<thead>
<tr>
<th>Metric</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frame total</td>
<td>5.89 ms</td>
</tr>
<tr>
<td>Virtual fps</td>
<td>170 fps</td>
</tr>
</tbody>
</table>

### WebGPU – Snapshot Standard Mode

<table>
<thead>
<tr>
<th>Metric</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frame total</td>
<td>3.75 ms</td>
</tr>
<tr>
<td>Virtual fps</td>
<td>267 fps</td>
</tr>
</tbody>
</table>

### WebGPU – Snapshot Fast Mode

<table>
<thead>
<tr>
<th>Metric</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frame total</td>
<td>0.36 ms</td>
</tr>
<tr>
<td>Virtual fps</td>
<td>2778 fps</td>
</tr>
</tbody>
</table>
More Fast Path optimizations (increase granularity)

External textures (Video texture/2D Canvas)

Node Material Editor support for new WGSL?