WebGL in Internet Explorer

Ben Constable

Principal SDE – Internet Explorer

Why WebGL in IE

- Strong developer feedback asking for it
- The standard has matured
  - Security
  - Recoverability
- 3D is cool for the web

Goals

- Security
- Interoperability
  - What end users want
  - What developers want
- Breadth of hardware
  - IE spans from ARM tablet to gaming desktop
  - Which means D3D9 FL to D3D11 FL

How does WebGL in IE work?

- We are built on D3D11
  - Same as other rendering code in IE
  - Need resource domain for images, etc
- GL to D3D call translation layer
  - Translate from JS
  - Translate from GL
- GLSL to HLSL converter
  - Built in-house

Security

- IE11 focuses on a safe, interoperable subset
- Shaders are restricted to reduce DOS attack surface
  - Limited instruction count
  - No loops that cannot unroll
- Windows provides TDR as defense in depth for DOS
  - IE will also preemptively drop back to SW
  - WARP used instead of hardware in these cases
- Driver stack vital to security
  - Worked with GPU vendors verifying drivers (source code review)
  - Older hardware (most DX9) replaced with WARP
- Cross domain images
  - We are using CORS

My WebGL does not work in IE!

- We are trying to do most used APIs first
  - Ordering is a huge challenge for us!
  - We have hundreds of sites we analyze for this
  - Real world use very important for our bootstrap
- API coverage in preview is not final for release
- But there are some features not in plan
- We love to hear feedback to tune ordering

Why not do some features?

- Mapping gap between D3D11 / OpenGL ES
  - Point size and line width good examples
- Emulation sucks performance out
- Example: Vertex*
  - No equivalent in D3D11
  - Emulating with vertex buffers adds layer of code
  - Heavy JS calls will also take time
  - Especially on low end hardware, will not be useful
  - No real world usage observed so far

Interoperability

- Breadth of hardware new to most web devs
  - People are used to dealing with different perf
  - But not used to different capability in same UA
- Our initial implementation tries to reduce this
  - Site written for IE11 should work everywhere
  - Desktop, Laptop, Tablet
- Ecosystem will evolve
  - Hopefully less caps than D3D9, but not zero
- Potential to expose this better in the API
  - Coarse vs fine grained caps

Looking forward

- Our implementation is evolving
  - More work done for 8.1 release
- How do we evolve the standard?
  - Immediate / retained mode issues
  - Better integration with HTML / CSS?
  - Higher level APIs?
  - Caps measurement
- We welcome feedback
  - bencon@microsoft.com