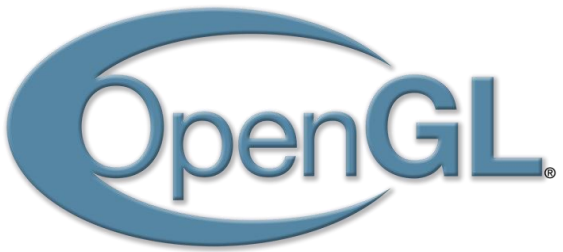


Vancouver SIGGRAPH 2014

Outline

- **Ecosystem Overview**
  - Neil Trevett (NVIDIA)
- **What's new in OpenGL**
  - Barthold Lichtenbelt (NVIDIA)
  - Bill Licea-Kane (Qualcomm)
- **What's new in OpenGL ES**
  - Tom Olson (ARM)
  - Demos (Rightware, NVIDIA, Kishonti)
- **Next Generation OpenGL Initiative**
  - Tom Olson (ARM)
  - Special guests
- **Party!**

Neil Trevett
VP Mobile Ecosystem, NVIDIA | President, Khronos
Vancouver SIGGRAPH 2014

Khronos Connects Software to Silicon

Open Consortium creating ROYALTY-FREE, OPEN STANDARD APIs for hardware acceleration

Defining the roadmap for low-level silicon interfaces needed on every platform

Graphics, compute, rich media, vision, sensor and camera processing

Rigorous specifications AND conformance tests for cross-vendor portability

Acceleration APIs
BY the Industry
FOR the Industry

Well over a *BILLION* people use Khronos APIs *Every Day…*

Access to 3D on Over 2 BILLION Devices

OpenGL|ES

1.9B Mobiles / year

OpenGL

300M Desktops / year

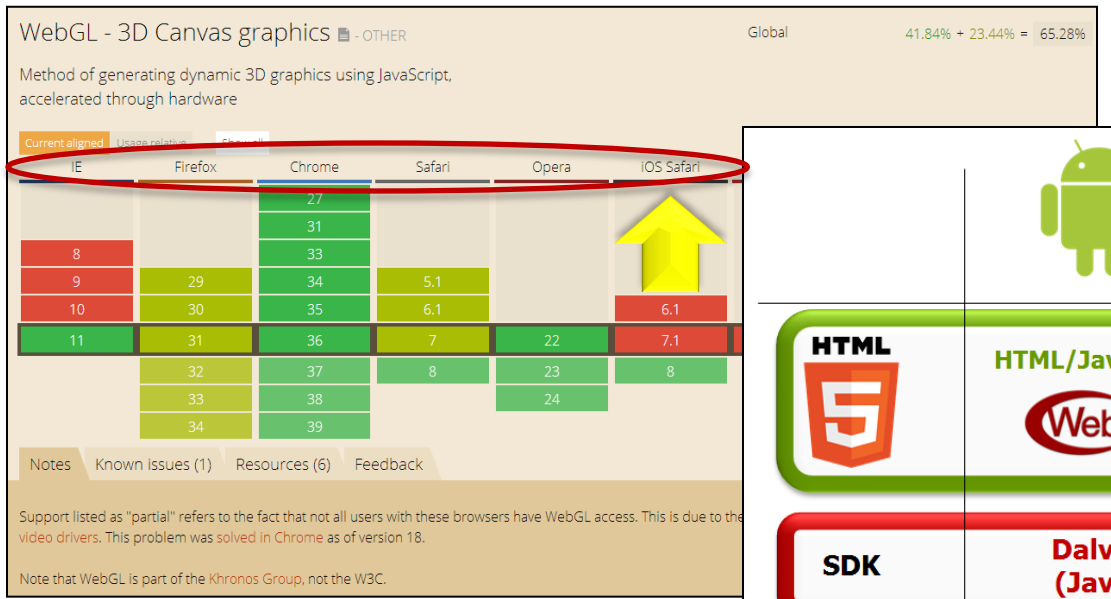
WebGL

1B Browsers / year

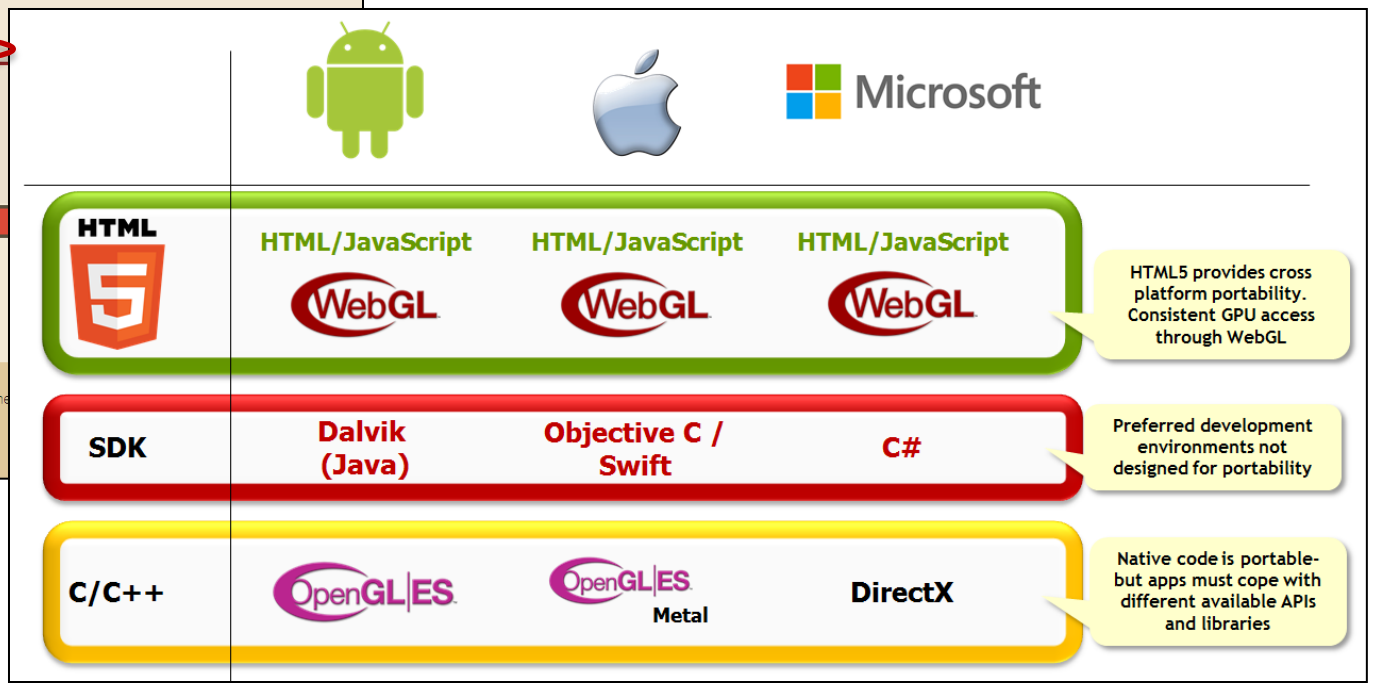
Source: Gartner (December 2013)

Pervasive WebGL

- **WebGL on EVERY major desktop browser**
  - And coming to all major mobile browsers
- **Portable (NO source change) 3D applications are possible for the first time**

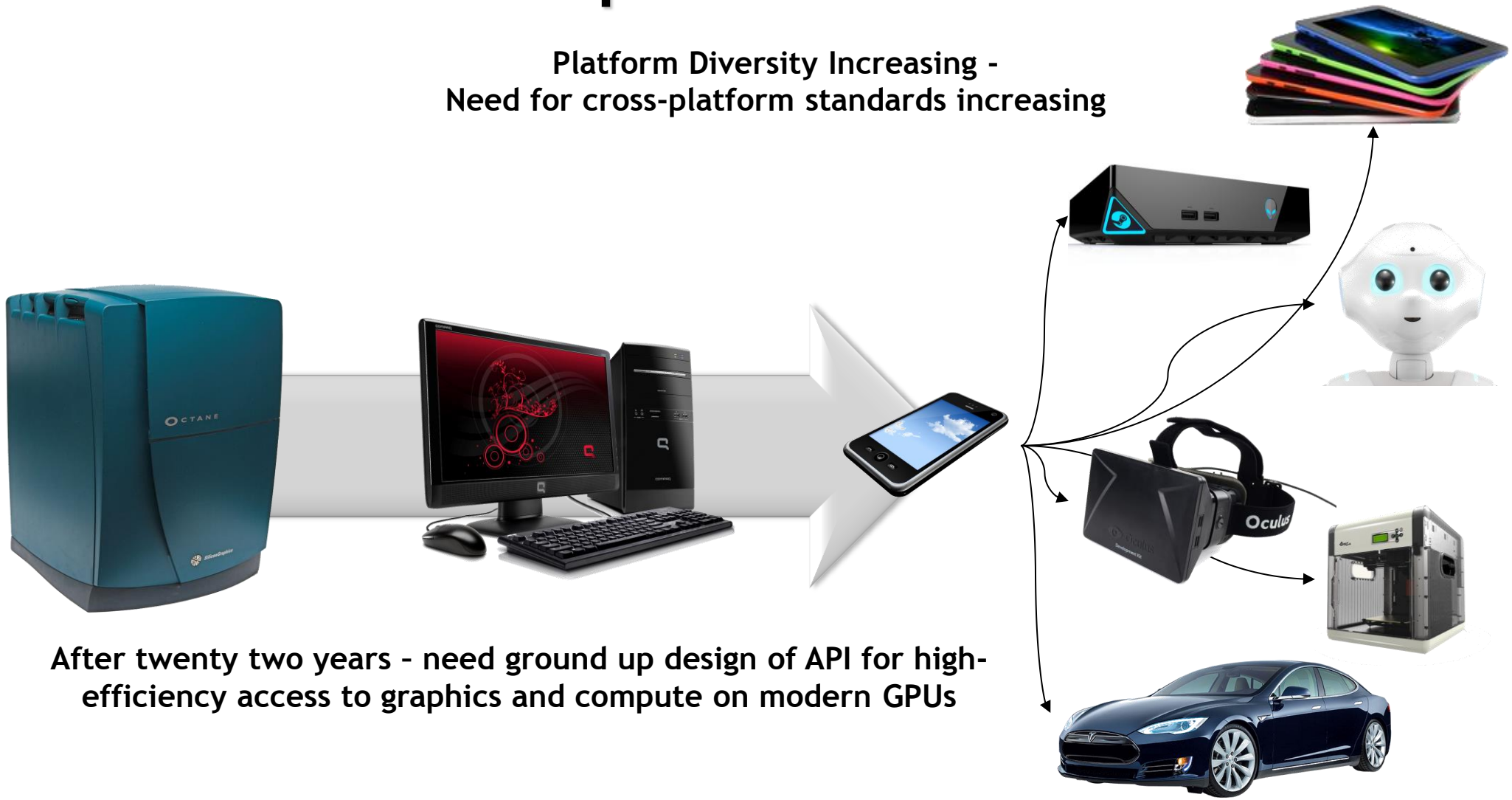


http://caniuse.com/#feat=webgl



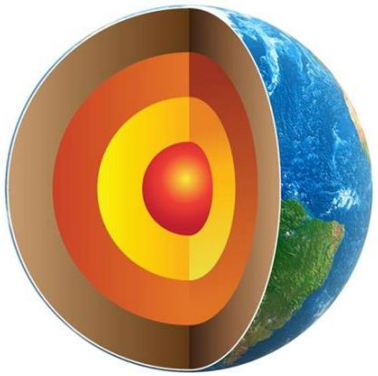
Next Generation OpenGL Initiative

Platform Diversity Increasing -
Need for cross-platform standards increasing



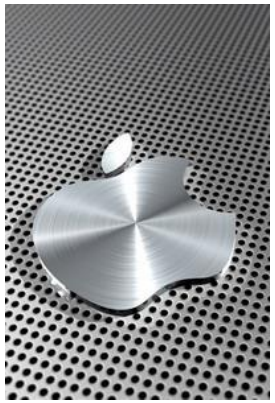
After twenty two years - need ground up design of API for high-efficiency access to graphics and compute on modern GPUs

Cross Platform Challenge



One family of GPUs

One OS

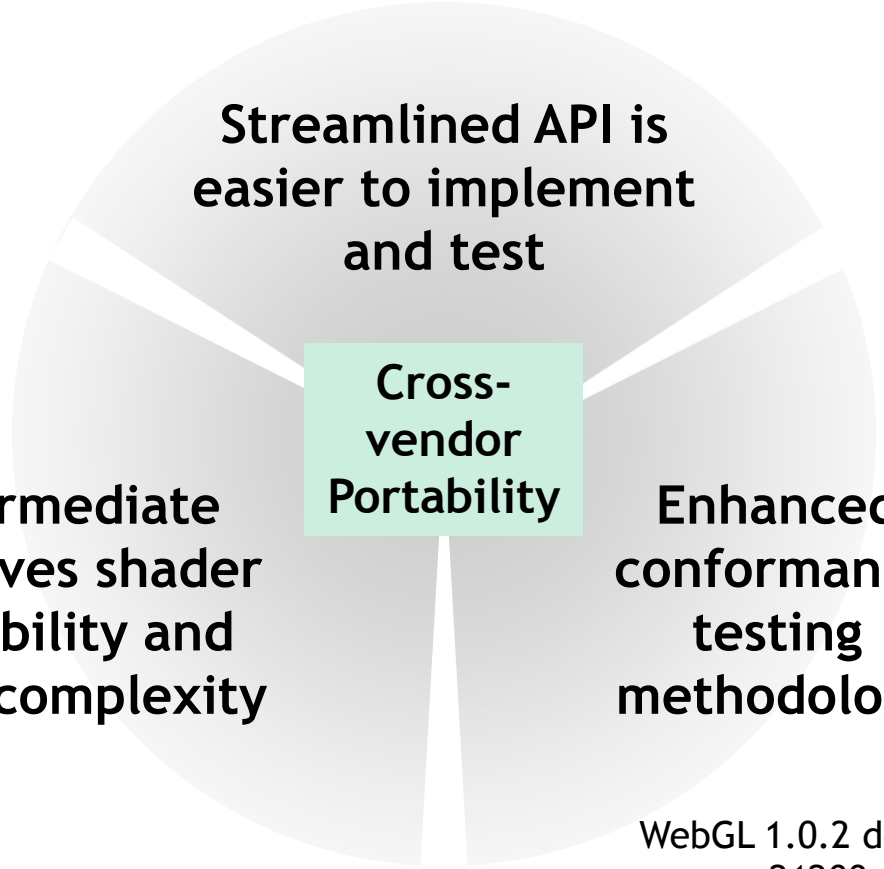


One GPU on one OS

All Modern Platforms and GPUs

Participation of key players
Proven IP Framework
Battle-tested cooperative model
The *drive* to not let the 3D industry fragment

Portability

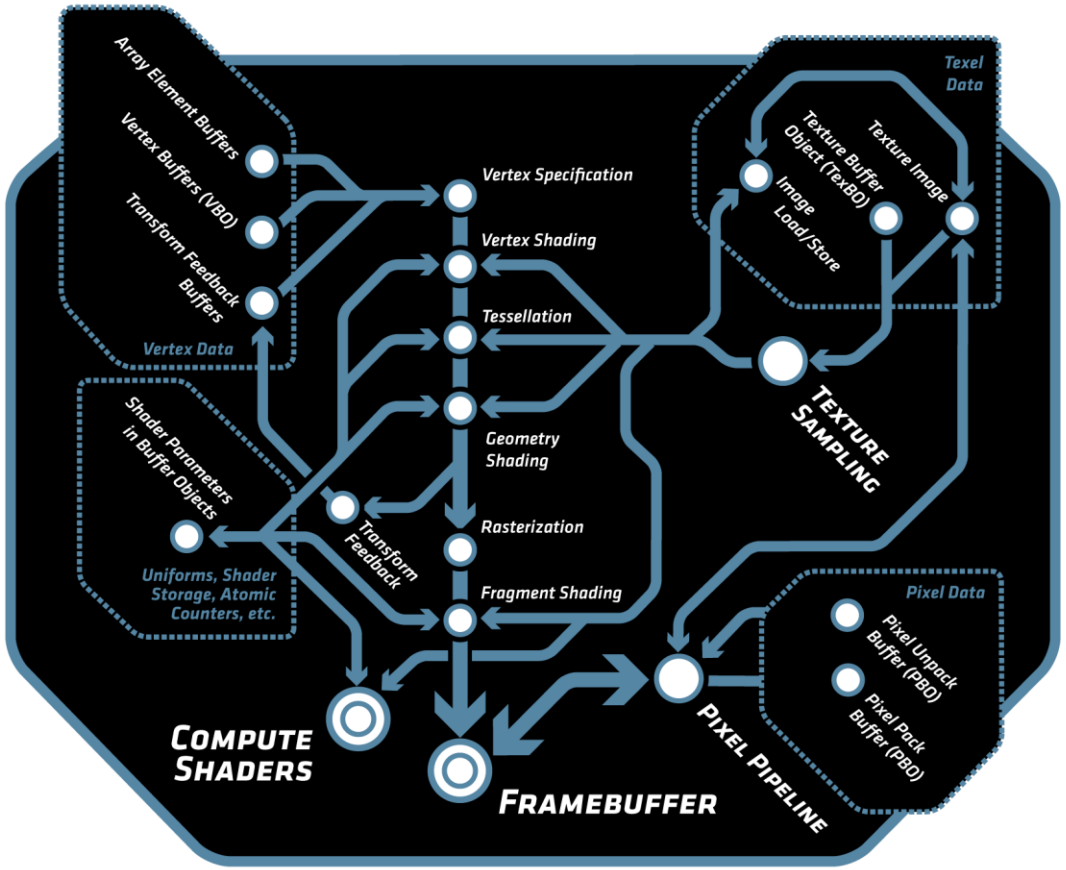


WebGL

WebGL 1.0.2 doubles conformance tests over 1.0.1
~21200 vs. ~8900
1.0.3 suite will contain ~20% more tests
Most contributed by community

Next Generation OpenGL Initiative

- Fast-paced work on detailed proposals and designs are already underway
- NOT your father's multi-year design-by-committee process
- Goal to unify OpenGL and OpenGL ES - hence one BOF!
- If this new API relevant to your business - PLEASE JOIN!



KHRONOS GROUP

OpenGL®

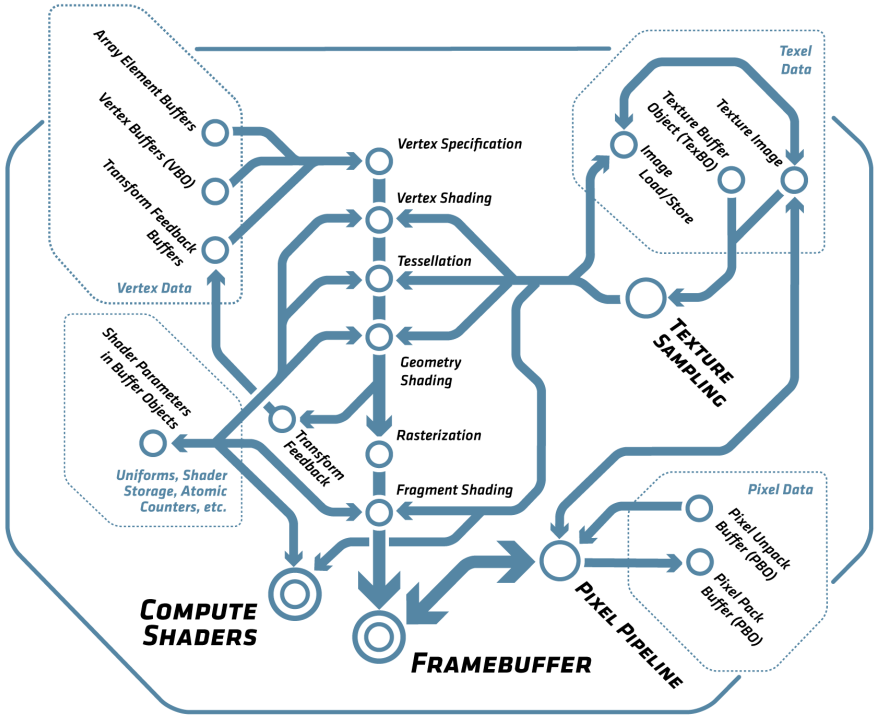
BOF

SIGGRAPH 2014

Barthold Lichtenbelt
OpenGL ARB chair

Announcing

Continuing OpenGL Innovation



Bringing state-of-the-art functionality to cross-platform graphics

OpenGL 4.5

OpenGL 4.4

OpenGL 4.3

OpenGL 4.2

OpenGL 4.1

OpenGL 3.3/4.0

OpenGL 3.2

OpenGL 3.1

OpenGL 2.0 **OpenGL 2.1** **OpenGL 3.0**

2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014
DirectX 9.0c		DirectX 10.0		DirectX 10.1		DirectX 11		DirectX 11.1		DirectX 11.2

Twitter Feedback

ExtremeTech

- OpenGL 4.5 released, next-gen OpenGL unveiled: Cross-platform Mantle killer, DX12 competitor bit.ly/1sO7JzX

pjcozzi

- Big things are happening with OpenGL! khronos.org/news/press/khr… @thekhronosgroup

Aras Pranckevičius @aras_p

- New GL_ARB_direct_state_access is good. Awesome work @g_truc and @grahamsellers!

More Feedback

esmittramirez

- Opengl 4.5, compat y glsl 4.5!! amazing, ahora a seguir leyendo las novedades (que de hecho no son novedades sino chismes ya oficializados)

phosphor112

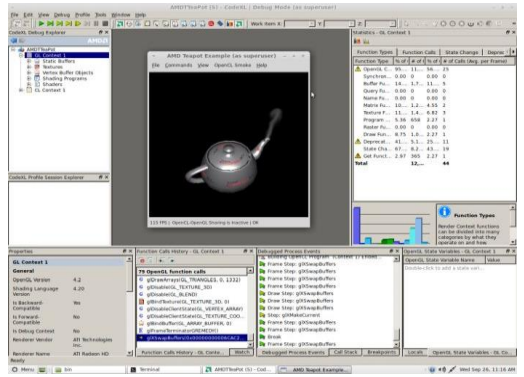
- #OpenGL 4.5? YEAAAAH BUDDY! @thekhronosgroup

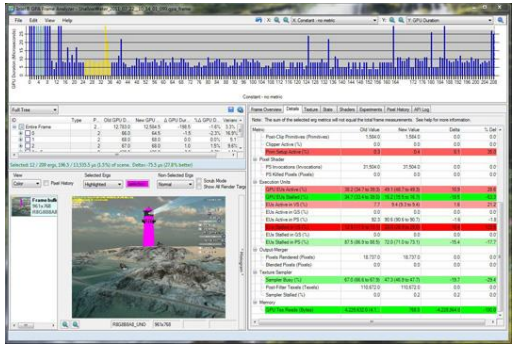
OpenGL Ecosystem News

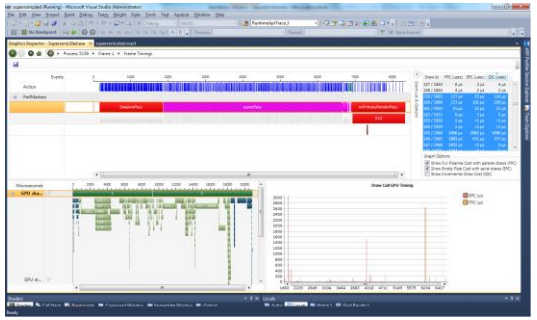
- **Epic's Unreal Engine 4 supports OpenGL 4**
  - Tessellation, Compute shaders, more
- **OpenGL Extension Wrangler (GLEW) supports OGL 4.5**
  - Release 1.11.0 has OpenGL 4.5 support
  - http://glew.sourceforge.net/
- **Regal updated to OpenGL 4.5 support**
  - https://github.com/p3/regal
- **VOGL updated**
  - now supports Windows
- **GLIntercept 1.3 with full OpenGL 4.4 all ARB extensions**
  - https://code.google.com/p/glintercept/
- **New version of the OpenGL Hardware database**
  - 274 devices listed to date.
  - http://delphigl.de/glcapsviewer/listreports2.php
-

Tools for Looking at Performance Bottlenecks

- **AMD**
  - AMD PerfStudio 2
  - http://developer.amd.com/tools-and-sdks/graphics-development/gpu-perfstudio-2/
  - AMD GPUperfAPI
  - http://developer.amd.com/tools-and-sdks/graphics-development/gpuperfapi/
  - AMD GPU Shader Analyzer
  - http://developer.amd.com/tools-and-sdks/graphics-development/gpu-shaderanalyzer/
- **Intel**
  - Intel GPA
  - https://software.intel.com/en-us/vcsource/tools/intel-gpa
- **NVIDIA**
  - NVIDIA PerfKit
  - https://developer.nvidia.com/nvidia-perfkit
  - NVIDIA NSIGHT
  - https://developer.nvidia.com/nvidia-nsight-visual-studio-edition







Glslang Reference Validator

- Virtually all of ESSL 3.1
- Compute shaders (minus arrays of arrays, coming soon…)
- Images (load/store/atomics, formats, memory qualifiers/barrier)
- Atomic Counters
- Shader Storage Buffer Objects
- Texture Gather
- Texture LOD
- Separate Shader Objects
- 420 Shading Pack completed
- All GLSL 4.5-specific features
- Others

test.comp

```
#version 310 es

layout(local_size_z = 1000) in;

void main()
{
   barrier();
}
```

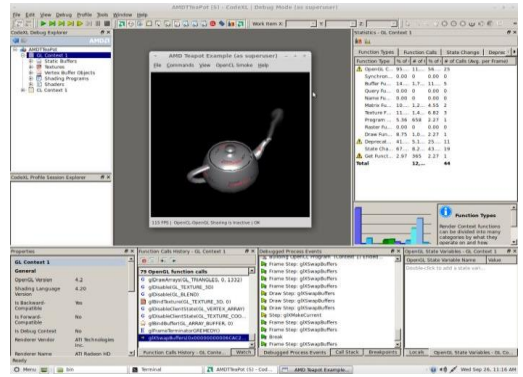
```
$ glslangValidator  test.comp
ERROR: 0:3: 'local_size' : too large; see gl_MaxComputeWorkGroupSize
ERROR: 1 compilation errors.  No code generated.
```

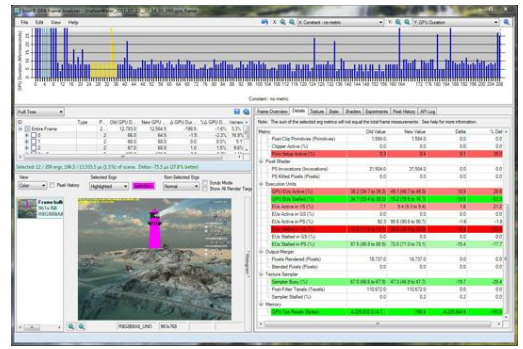
Tools for Performance Analysis

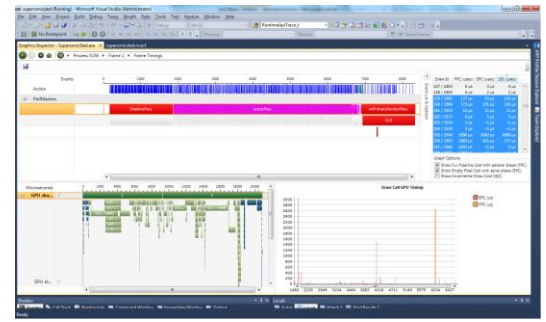
- **AMD**
  - AMD PerfStudio 2
    http://developer.amd.com/tools-and-sdks/graphics-development/gpu-perfstudio-2/
  - AMD GPUperfAPI
    http://developer.amd.com/tools-and-sdks/graphics-development/gpuperfapi/
  - AMD GPU Shader Analyzer
    http://developer.amd.com/tools-and-sdks/graphics-development/gpu-shaderanalyzer/

- **Intel**
  - Intel GPA
    https://software.intel.com/en-us/vcsource/tools/intel-gpa

- **NVIDIA**
  - NVIDIA PerfKit
    https://developer.nvidia.com/nvidia-perfkit
  - NVIDIA NSIGHT
    https://developer.nvidia.com/nvidia-nsight-visual-studio-edition



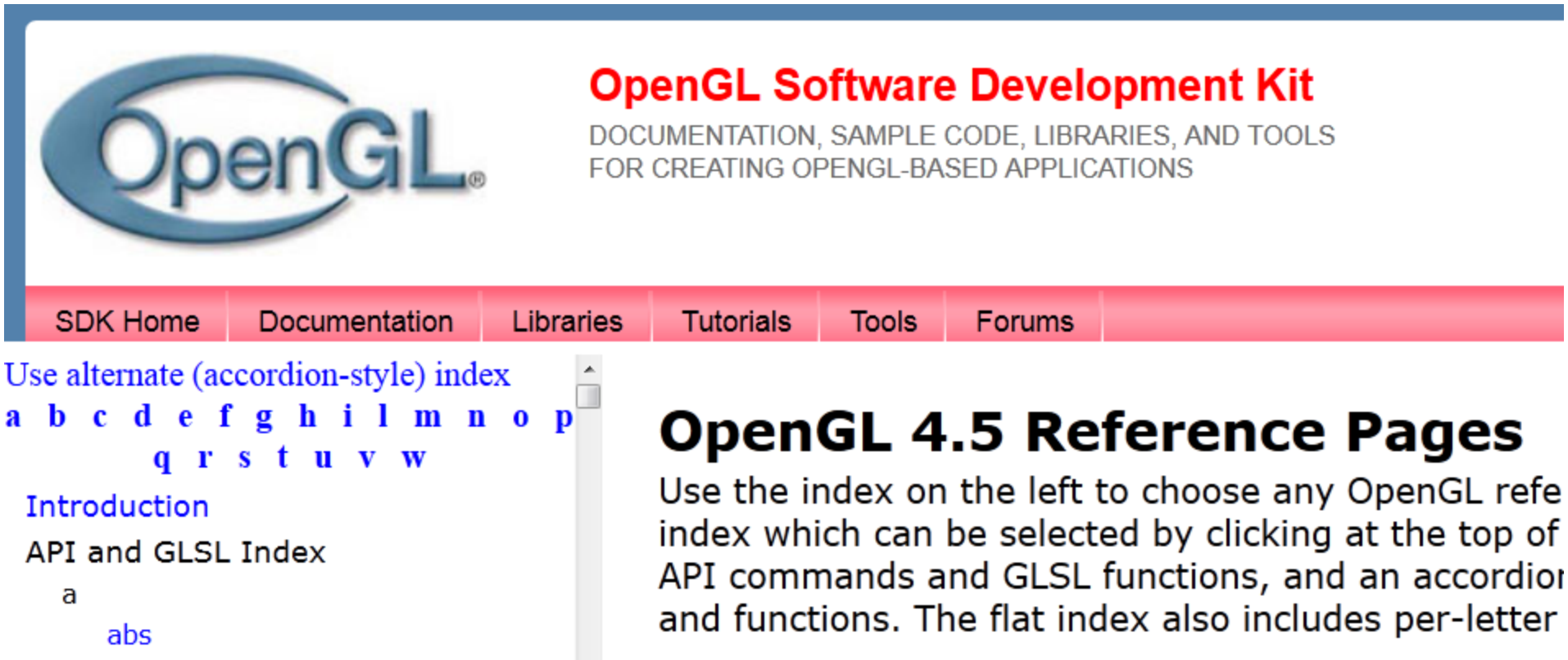




OpenGL Conformance Tests!

- **Conformance submissions required for GL 4.4 and 4.5 implementations**
  - Encouraged for earlier driver versions
- **Continued improvements to the code**
  - Significant OpenGL 4.4 coverage added
  - OpenGL 4.5 coverage in progress
- **Shared codebase with OpenGL ES 3.1 CTS**
  - Additional desktop-specific tests

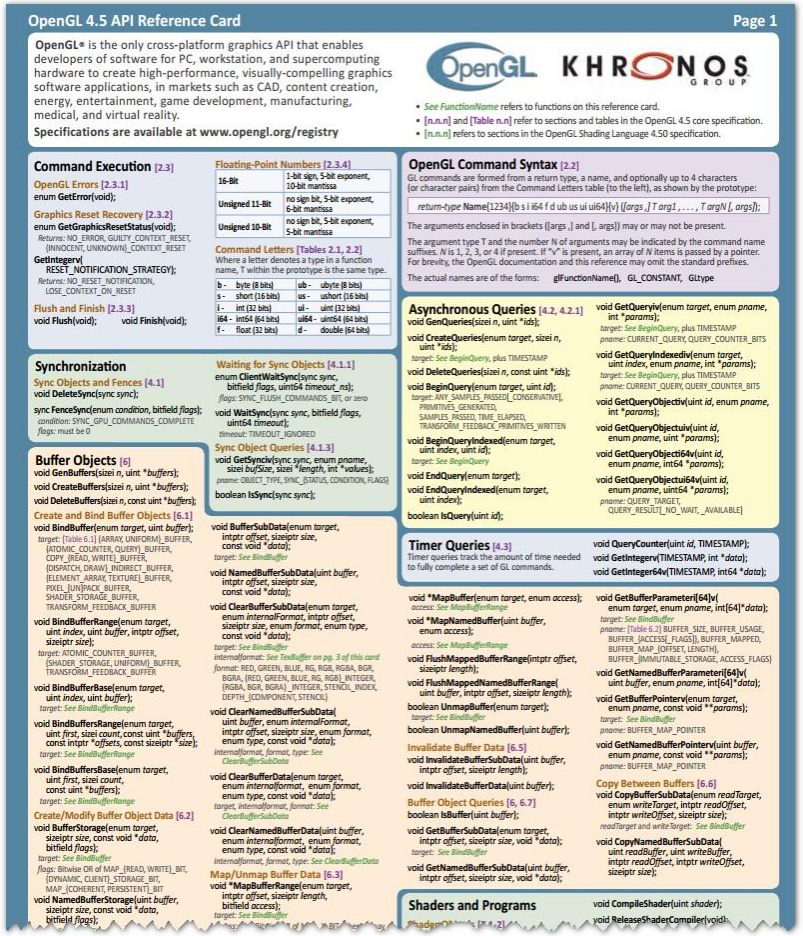
OpenGL 4.5 reference pages



Thanks to Dominic Witczak and Jon Leech!

OpenGL 4.5 Reference Cards

becomes



What is New in OpenGL 4.5?

- **Direct State Access (DSA)**
  - Assessors for objects, enabling state to be queried and modified directly
- **Robustness**
  - Providing a secure platform for applications such as WebGL browsers, including preventing a GPU reset affecting any other running applications
- **DirectX11 emulation features**
  - Easier porting of applications between OpenGL and Direct3D
- **OpenGL ES 3.1 API and shader compatibility**
  - Enables easy development and execution of OpenGL ES applications on desktop
- **Faster MakeCurrent**
  - Explicit control over context flush behavior
- **Texture Barrier**
  - Avoid read-after-write hazards

What is New in GLSL 4.5?

- **ARB_derivative_control**
  - Control spatial granularity of derivative computation
- **ARB_shader_texture_image_samples**
  - Allow shaders to query the number of samples of a texture
  - `textureSamples(), imageSamples()`
- **ARB_cull_distance**
  - adds a new GLSL `gl_CullDistance` shader output, similar to `gl_ClipDistance`, but used for whole primitive culling

New OpenGL ARB extensions

- **ARB_sparse_buffer**
  - Similar functionality to ARB_sparse_texture but for buffer objects
- **ARB_pipeline_statistics_query**
  - Query statistics about different parts of the OpenGL pipeline
  - DirectX 11 compatibility
- **ARB_transform_feedback_overflow_query**
  - Query if a transformfeedback buffer overflowed
  - For one or more streams
  - DirectX 11 compatibility

OpenGL 4.5 - DirectX11 emulation

- **ARB_conditional_render_inverted**
  - Add mode to BeginConditinalRender to invert the condition to draw, or not
- **ARB_get_texture_subimage**
  - Retrieve sub-regions of textures
- **ARB_clip_control**
  - Match D3D coordinate system
  - Improve numerical precision of Z coordinate mapping

OpenGL 4.5 - OpenGL ES 3.1 Compatibility

- **ARB_ES3_1_compatibility**
  - Add `MemoryBarrierRegion` API
  - Increase minimum SSBO size to `2^27`
  - Support GLSL ES version 310
  - Add `imageAtomicExchange` for r32f FP images
  - Extend `mix()` to select between int, uint and bool
  - Add GLSL built-in fragment shader input : `gl_HelperInvocation`
  - Add GLSL built-in constants
    - `gl_MaxSamples, gl_Max*ImageUniforms, gl_MaxCombinedShaderOutputResources`
  - Add coherent qualifier to all memory variables for `atomic*` and `imageAtomic*`

Direct State Access

Before DSA

```
void Texture2D::SetMagFilter(GLenum filter)
{
    GLuint oldTex;
    glGetIntegerv(GL_TEXTURE_BINDING_2D, &oldTex);
    glBindTexture(GL_TEXTURE_2D, m_tex);
    glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_MAG_FILTER, filter);
    glBindTexture(GL_TEXTURE_2D, oldTex);
}
```

After DSA

```
void Texture2D::SetMagFilter(Glenum filter)
{
    glTextureParameteri(m_tex, GL_TEXTURE_MAG_FILTER, filter);
}
```

Direct State Access

Before DSA

```
glBindFramebuffer(GL_DRAW_FRAMEBUFFER, msFBO);

DrawStuff();

glBindFramebuffer(GL_DRAW_FRAMEBUFFER, nonMsFBO);
glBindFramebuffer(GL_READ_FRAMEBUFFER, msFBO);
glBlitFramebuffer(...);
glBindFramebuffer(GL_DRAW_FRAMEBUFFER, msFBO);
```

After DSA

```
glBindFramebuffer(GL_DRAW_FRAMEBUFFER, msFBO);

DrawStuff();

glBlitNamedFramebuffer(msFBO, nonMsFBO, ...);
```

Differences with EXT_direct_state_access

- **Only OpenGL 4.5 Core functionality supported**
- **Some minor name changes to some functions**
  - Mostly the same, but drops EXT suffix
    - `TextureParameterfEXT -> TextureParameterf`
  - VAO function names shortened
    - `glVertexArrayVertexBindingDivisorEXT-> glVertexArrayBindingDivisor`
  - Texture functions no longer require a target parameter
    - Target comes from `glCreateTextures(<target>,)`
    - Use “3D” functions with CUBE_MAP where z specifies the face
- **DSA functions can no longer create objects**
  - Use `glCreate*` functions to create name and object at once

Texture Barrier programmable blending

- **programmable blending with non-self-overlapping draw calls**
  - Bind texture as a render target **and** for texturing

```
glBindTexture(GL_TEXTURE_2D, tex);
glFramebufferTexture(GL_FRAMEBUFFER, GL_COLOR_ATTACHMENT0, tex, 0);

dirtybbox.empty();

foreach (object in scene) {
    if (dirtybbox.overlaps(object.bbox())) {
        glTextureBarrier();
        dirtybbox.empty();
    }
    object.draw();
    dirtybbox = bound(dirtybbox, object.bbox());
}
```

Robustness

- **ARB_robustness functionality now part of OpenGL 4.5**
  - Called `KHR_robustness` for use with OpenGL ES too
  - Does not include compatibility functions
- **Adds “safe” APIs for queries that return data to user pointers**
- **Adds mechanism for app to learn about GPU resets**
  - Due to my app or some other misbehaving app
- **Stronger out-of-bounds behavior**
  - No more undefined behavior
- **Used by WebGL implementations**

Robustness safe API example

Before Robustness

```
GLubyte tooSmall[NOT_BIG_ENOUGH];
glReadPixels(0, 0, H, W, GL_RGBA, GL_UNSIGNED_BYTE, tooSmall);
// CRASH
```

After Robustness

```
GLubyte tooSmall[NOT_BIG_ENOUGH];
glReadnPixels(0, 0, H, W, GL_RGBA, GL_UNSIGNED_BYTE,
              sizeof tooSmall, tooSmall);

// No CRASH, glGetError() returns INVALID_OPERATION
```

Robustness Reset Notification example

- Typical render loop with reset check

```
while (!quit) {
    DrawStuff();
    SwapBuffers();
    if (glGetGraphicsResetStatus() != GL_NO_ERROR) {
        quit = true;
    }
}
DestroyContext(glrc);
```

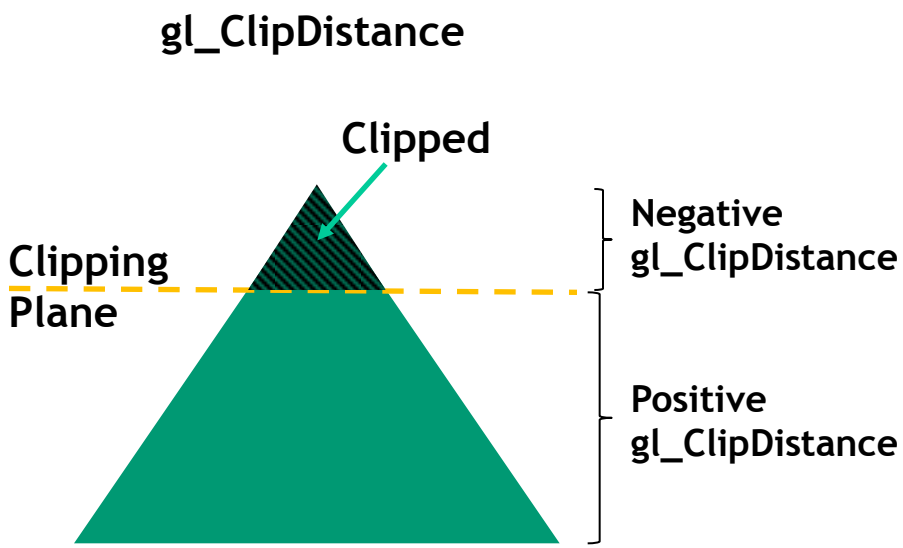
Faster MakeCurrent

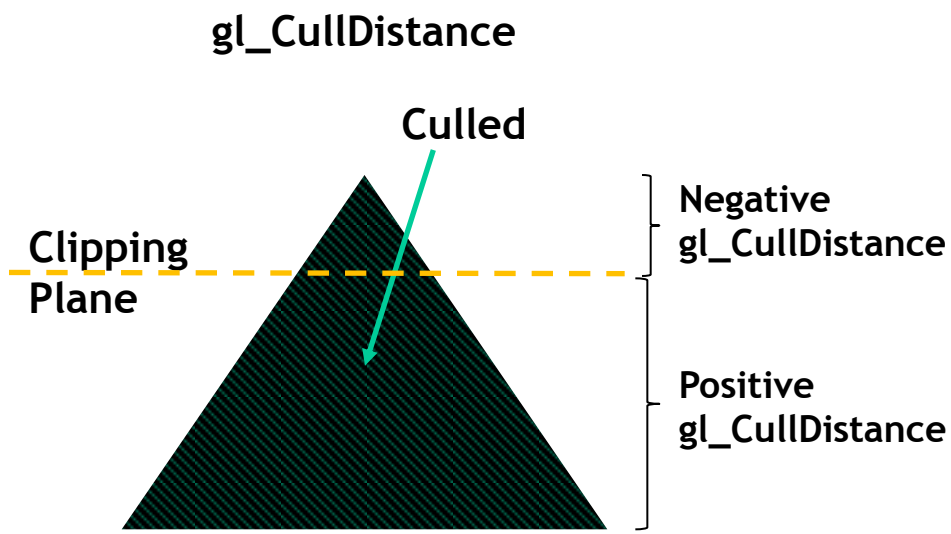
- **An implicit `glFlush` is called on `MakeCurrent`**
  - Makes switching contexts slow
- **New WGL and GLX extensions allow `glFlush` to be skipped**
  - Commands wait in context queue
  - App has more control over flush
- **Can provide significant `MakeCurrent` performance boost**

```
StartTimer();
for (int i = 0; i < iterations; ++i) {
    DrawSimpleTriangle();
    wglMakeCurrent(context[i % 2]);
}
StopTimer();
```

ARB_cull_distance

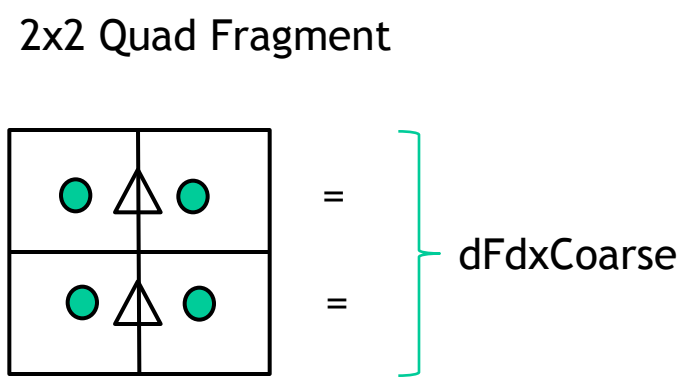
- Adds new `gl_CullDistance` to Vertex, Tessellation, and Geometry shaders
- Like `gl_ClipDistance` except when any vertex has negative distance whole primitive is culled

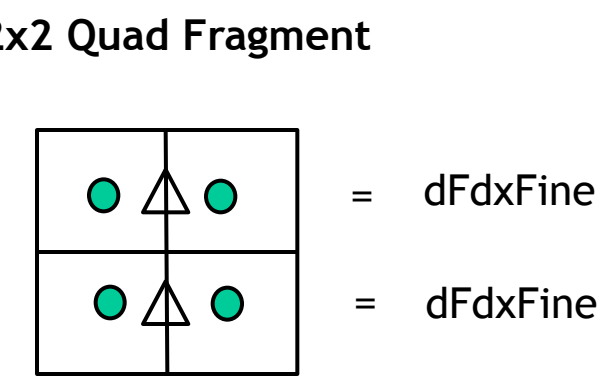




GLSL derivative Control

- **Adds “coarse” and “fine” variant of GLSL derivative functions**
- **`dFdxCoarse, dFdyCoarse`**
  - Potentially faster performance
- **`dFdxFine, dFdyFine`**
  - More correct
  - Default behavior of old `dFdx` and `dFdy` functions
- `fwidthCoarse` **and** `fwidthFine` **are also added**





Get Texture Sub Image

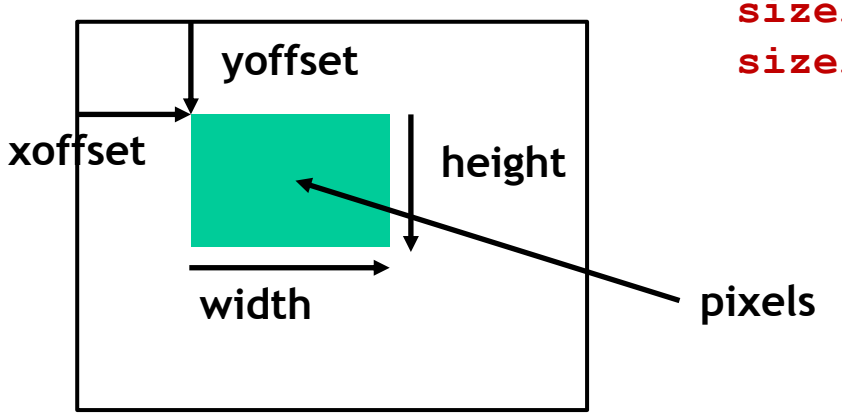
- Like glGetTexImage, but now you can read a sub-region

- `glGetTextureSubImage()`
  - DSA only variant

Direct State Access

```
void GetTextureSubImage(uint texture, int level,
    int xoffset, int yoffset, int zoffset, sizei width,
    sizei height, sizei depth, enum format, enum type,
    sizei bufSize, void * pixels);
```

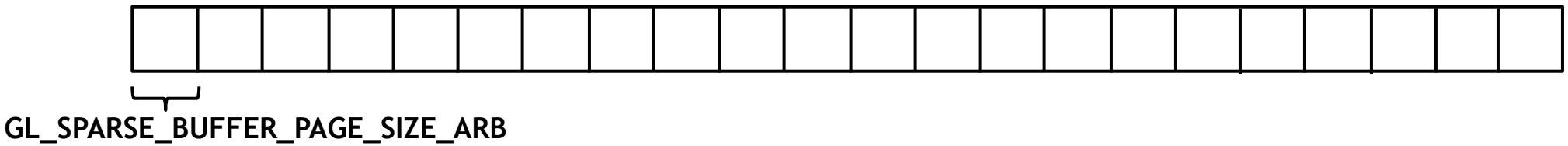
Robustness



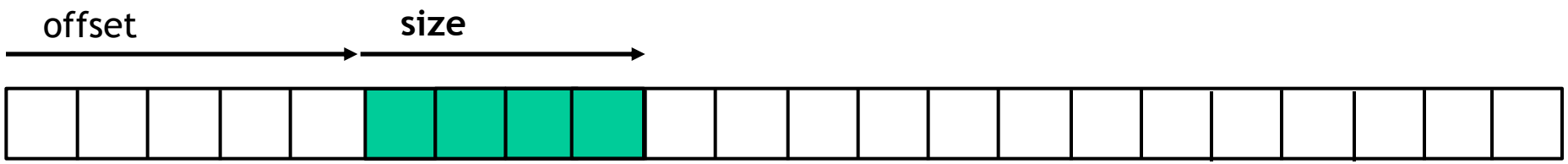
ARB_sparse_buffer

- **Have large buffer objects without the whole buffer being resident**
  - Analogous to ARB_sparse_texture for buffer objects
  - Application controls page residency

Create uncommitted buffer: glBufferStorage(,SPARSE_STORAGE_BIT_ARB)



Make pages resident: glBufferPageCommitmentARB(, offset, size, GL_TRUE);



New OpenGL ARB extensions

- **ARB_sparse_buffer**
  - Similar functionality to ARB_sparse_texture but for buffer objects
- **ARB_pipeline_statistics_query**
  - Query statistics about different parts of the OpenGL pipeline
  - DirectX 11 compatibility
- **ARB_transform_feedback_overflow_query**
  - Query if a transformfeedback buffer overflowed
  - For one or more streams
  - DirectX 11 compatibility

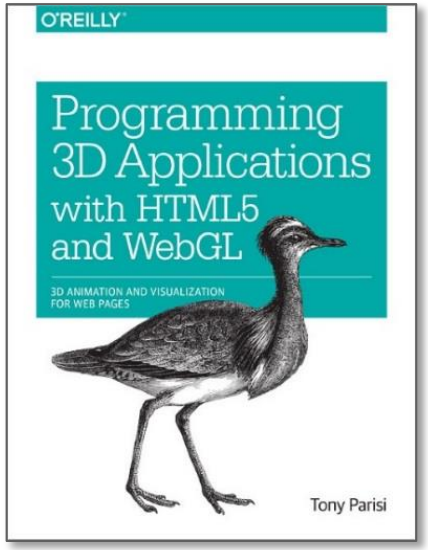
OpenGL 4.5 - DirectX11 portability

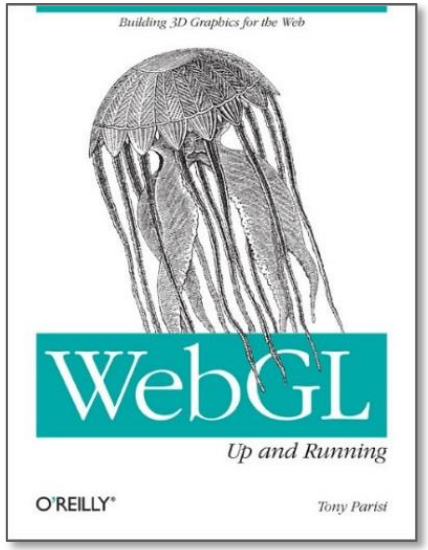
- **ARB_conditional_render_inverted**
  - Add mode to `BeginConditinalRender` to invert the condition to draw, or not
- **ARB_get_texture_subimage**
  - Retrieve sub-regions of textures
- **ARB_clip_control**
  - Match D3D coordinate system
  - Improve numerical precision of Z coordinate mapping

OpenGL 4.5 - OpenGL ES 3.1 compatibility

- **ARB_ES3_1_compatibility**
  - Add `MemoryBarrierRegion` API
  - Increase minimum SSBO size to `2^27`
  - Support GLSL ES version 310
  - Add `imageAtomicExchange` for r32f FP images
  - Extend `mix()` to select between int, uint and bool
  - Add GLSL built-in fragment shader input : `gl_HelperInvocation`
  - Add GLSL built-in constants
    - `gl_MaxSamples, gl_Max*ImageUniforms, gl_MaxCombinedShaderOutputResources`
  - Add coherent qualifier to all memory variables for `atomic*` and `imageAtomic*`

Win a book at the party...





What's New in OpenGL ES

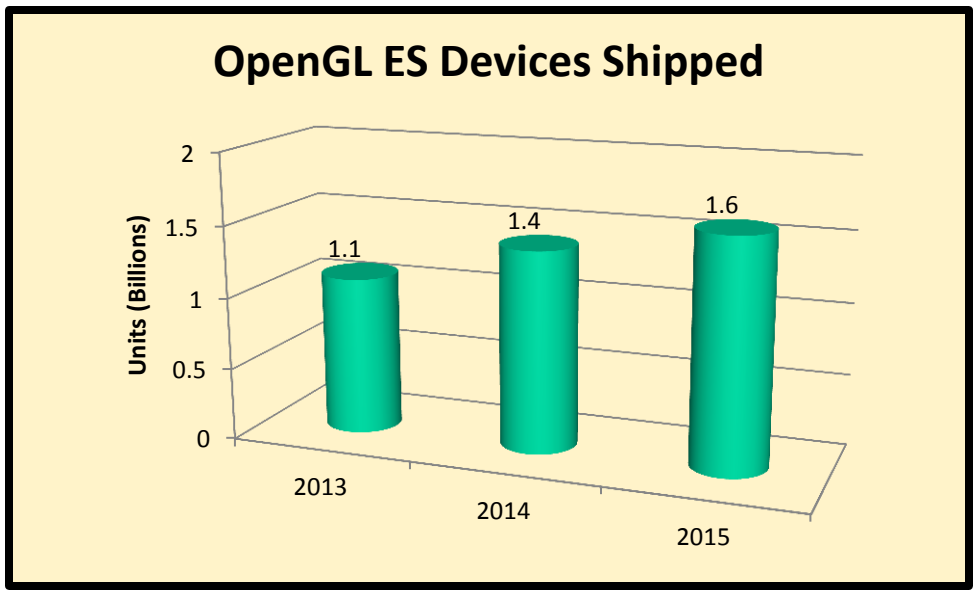
Tom Olson
Director of Graphics Research, ARM
Chair, OpenGL ES Working Group

Outline

- **Where we are today (Tom Olson, ARM)**
- **OpenGL ES 3.1 Overview (Bill Licea-Kane, Qualcomm)**
- **OpenGL ES 3.x rollout (Tom again)**
- **Demos (Members)**
  - Basemark OS II OpenGL ES 3.1 (Arto Ruotsalainen, Rightware)
  - NVIDIA: OpenGL ES 3.1 + Android Extension Pack (Neil Trevett, NVIDIA)
  - GFXBench Car Chase Preview (Laszlo Kishonti, Kishonti Informatics)

It’s been a good 10* years!

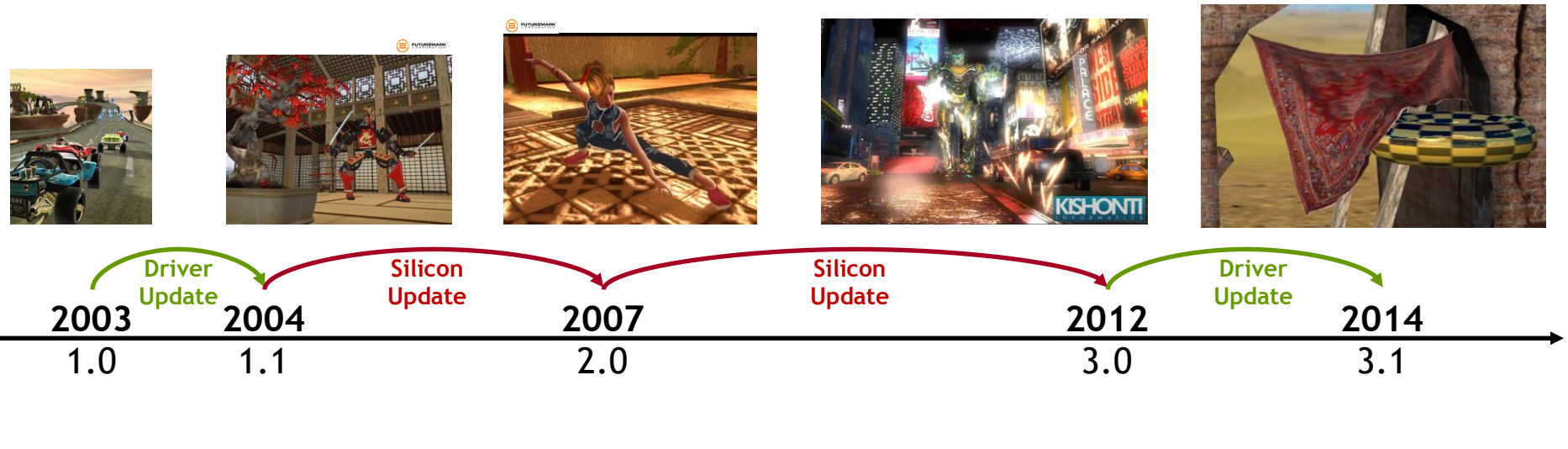
- SIGGRAPH 2007: 15 million devices sold!
- SIGGRAPH 2008: 54 million…
- Where are we today?



OpenGL ES is the most widely deployed 3D API in history

*For large values of 10

The Story So Far



- **What's new since SIGGRAPH 2013?**
  - Release of ES 3.1 at GDC
  - Conformance test upgrades
  - Ecosystem evolution

OpenGL ES 3.1 Goals

- **Bring appropriate features of desktop OpenGL 4.x to mobile devices**
  - Game-changing functionality: compute shaders, indirect drawing
  - Other advanced features, support for modern programming styles

- **Enable very rapid adoption**
  - Expose hidden capabilities of ***current*** devices
  - Backward compatible with ES 2.0/3.0
  - Run on any OpenGL ES 3.0 hardware

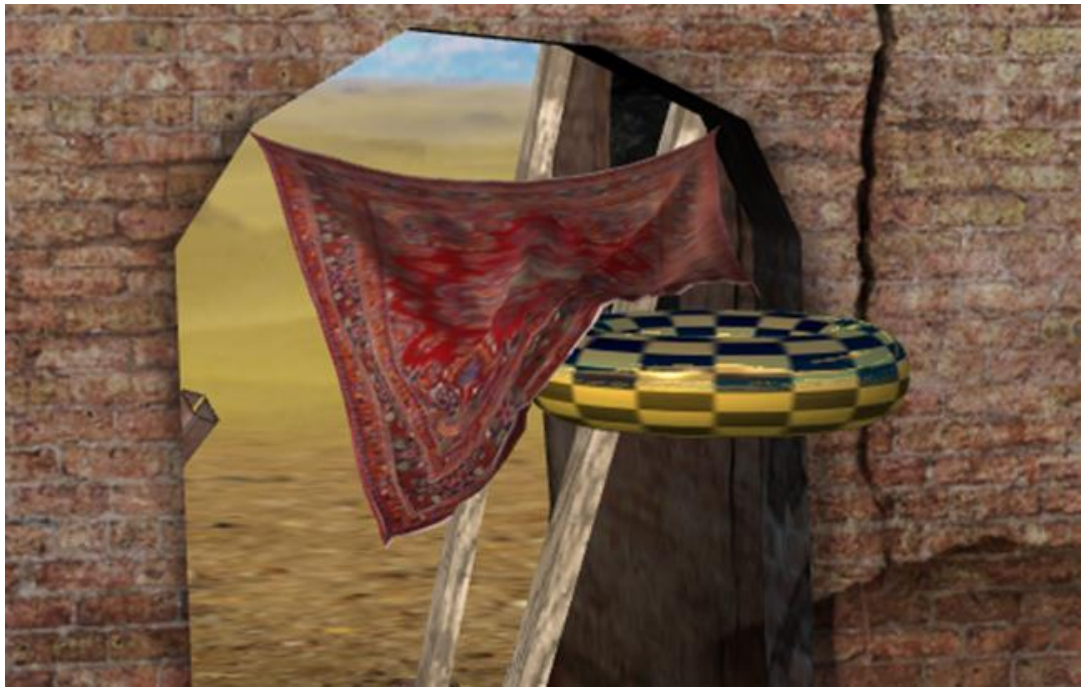
- **Improve application portability**
  - Tighter specs means less undefined behavior
  - Ever stricter conformance testing

OpenGL ES 3.1 Overview

Bill Licea-Kane, Qualcomm Technologies, Inc.
Shading Language Chair

OpenGL ES 3.1

- **Compute shaders**
  - Tightly coupled graphics and GPU computing
- **Draw-Indirect**
  - Draw command parameters from memory instead of from arguments
- **Separate Shader Object (SSO)**
  - Mix and match graphics shaders
- **New shading language features**
- **New Texture features**



Compute Shader Basics

- **Use the power of the GPU for general purpose computing**
  - New single-stage pipeline, separate from the graphics pipeline
- **OpenGL ES *graphics* pipeline**
  - Implicitly creates a thread per vertex and a thread per fragment
  - All thread input / output data routing and synchronization is done for you
- **OpenGL ES *compute* pipeline**
  - Explicitly creates a thread per point on an integer lattice
  - Data input, result output, and synchronization are all explicit

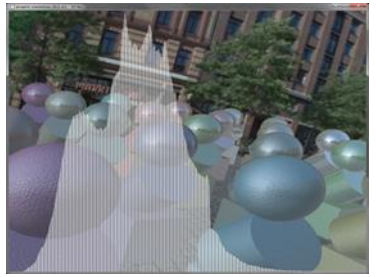
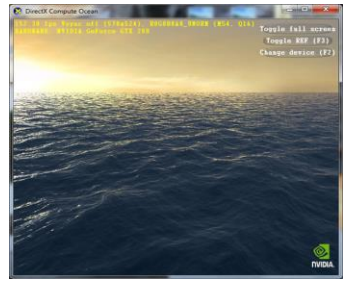


Image processing



Wave Simulation

AI Simulation

Global Illumination

Ray Tracing

Compute Shader Components

- **New shader type and dispatch call**
  - glDispatchCompute()
- **Shader Storage Buffer Objects (SSBOs)**
  - Essentially writeable UBOs; minimum max size is 128MB
  - Accessible via explicit load/store or atomics
- **Images**
  - Like textures, but writeable as well as readable
  - Every implementation provides at least four image binding points
  - Image atomics available as an extension
- **Serialization and synchronization primitives**
  - Atomic counters
  - Shader Memory barriers
  - API Memory barriers

Draw-Indirect

- **A new way to dispatch work on the GPU**
  - Draw call parameters stored in GPU memory
  - glDrawArraysIndirect()
  - glDrawElementsIndirect()
  - glDispatchComputeIndirect()

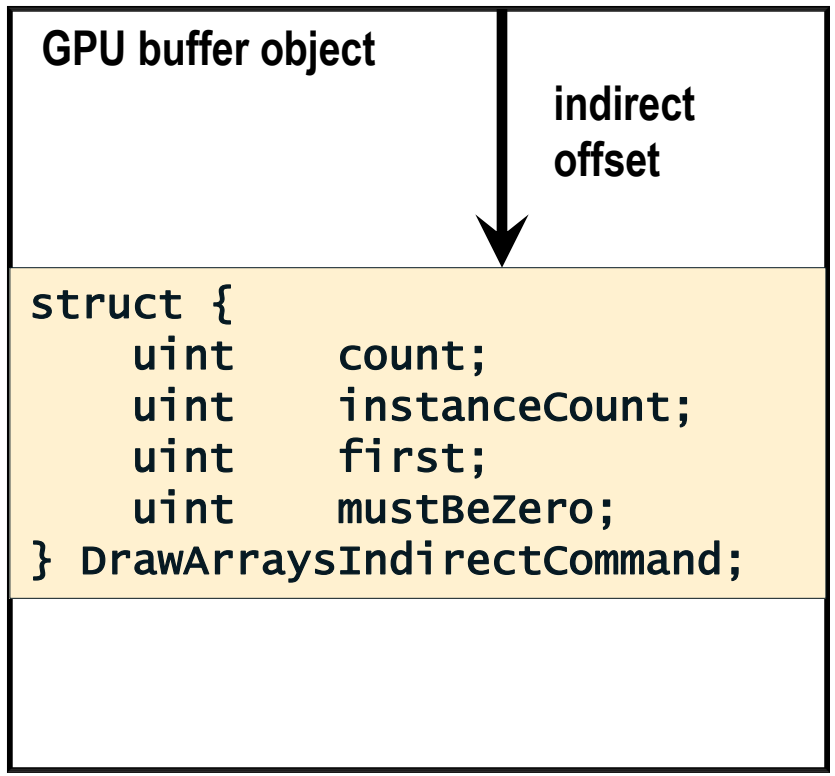
- **Useful for draw parameters set by GPU**
  - …for example, in a compute shader
  - Eliminate CPU/GPU synchronization point

Draw-Indirect

- **A new way to dispatch work on the GPU**
  - Draw call parameters stored in GPU memory
  - glDrawArraysIndirect()
  - glDrawElementsIndirect()
  - glDispatchComputeIndirect()

- **Useful for draw parameters set by GPU**
  - …for example, in a compute shader
  - Eliminate CPU/GPU synchronization point

glDrawArraysIndirect(mode, indirect);



Separate Shader Objects

- MIX and MATCH
  vertex shaders
  and
  fragment shaders

Texture Features

- Multisampled Textures
- Stencil Textures
- Texture Gather

Shading Language Features

- **Arrays of arrays**
  - `int a[2][3][4];`

- **Explicit uniform location**
  - `layout(location=0) in vec4 position; // vertex shader example`
  - `layout(location=2) out vec4 diffuse; // fragment shader example`

- **Shader bitfield ops**
  - Extract, Insert, Reverse, pack/unpack, Count, LSB, MSB,
  - Extended precision support
  - frexp / ldexp

- **Shader helper invocation**
  - `bool gl_HelperInvocation`

OpenGL ES 3.x Ecosystem Rollout

Tom Olson, ARM
GL Common TSG chair

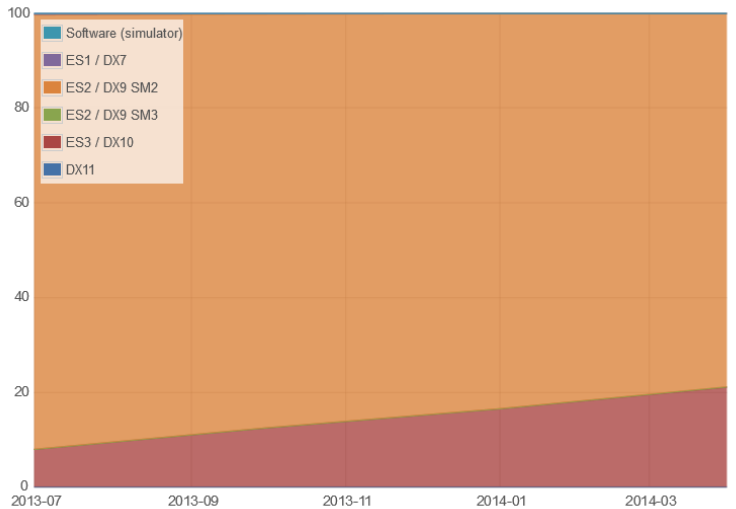
Ecosystem Rollout: OpenGL ES 3.0

- **Implementations from every major mobile GPU vendor**
  - Tools, documentation, sdks, etc.

- **Platform support**
  - Exposed in iOS 7, supported on iPhone 5s + recent iPads
  - Exposed in Android 4.3 and subsequent versions

- **Adoption is progressing**
  - E.g. Unity mobile survey: from 8% to 21%

http://stats.unity3d.com/mobile/index.html



Ecosystem Rollout: OpenGL ES 3.1

- **Documentation**
  - Specifications and man pages at http://www.khronos.org
- **SDKs, emulators, tools from multiple vendors**
  - Imagination Technologies
    - http://community.imgtec.com/developers/powervr/
  - NVIDIA
    - https://developer.nvidia.com/content/tegra-android-development-pack-3_0r2-released-start-developing-nvidia-shield-tablet-now
  - Qualcomm
    - https://developer.qualcomm.com/mobile-development/maximize-hardware/mobile-gaming-graphics-adreno/tools-and-resources
- **Platform support**
  - Will be exposed in Android L (selected devices)
  - Added functionality: 'Android extension pack', ASTC textures, etc.

Conformance Test Improvements

- **Updated ES 3.0 conformance (version 3.0.2.0) released in January 2014**
  - Substantial increase in test coverage
  - Adds subset of dEQP language tests (thanks drawElements Oy!)

- **ES 3.1 conformance is live as of 6 June 2014**
  - Directed tests for all features added in ES 3.1
  - Thirteen submissions received to date
    - Mobile parts from ARM, Imagination, Intel, NVIDIA, Qualcomm, Vivante
    - Desktop drivers from Intel, NVIDIA

Reference compiler

- Now supports virtually all of GLSL ES 3.1
- Compute shaders
  - (minus arrays of arrays, coming soon)
- Images
- Atomic Counters
- SSBOs
- Texture Gather
- Texture LOD
- SSO
- …and more

test.comp

```
#version 310 es

layout(local_size_z = 1000) in;

void main()
{
    barrier();
}
```

```
$ glslangValidator  test.comp
ERROR: 0:3: 'local_size' : too large; see gl_MaxComputeWorkGroupSize
ERROR: 1 compilation errors.  No code generated.
```

See: http://www.khronos.org/opengles/sdk/tools/Reference-Compiler/

Basemark OS II OpenGL ES 3.1

Arto Ruotsalainen, Rightware

OpenGL ES 3.1 + AEP

Neil Trevett, NVIDIA
Khronos President

Google Android Extension Pack (AEP)

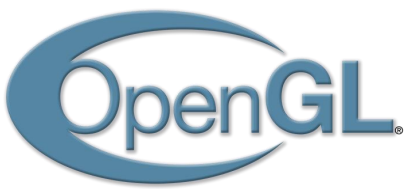
- **Android L mandates OpenGL ES 3.1**
  - On every device
- **Android Extension Pack (AEP)**
  - Set of OpenGL ES 3.1 extensions accessible through a single query
- **AEP includes OpenGL AAA gaming functionality**
  - Tessellation
  - Geometry shaders
  - ASTC Texture Compression + more…
  - https://www.khronos.org/registry/gles/extensions/ANDROID/ANDROID_extension_pack_es31a.txt
- **Enables AAA graphics effects**
  - Global Illumination
  - Physically-based shading
  - Deferred rendering
  - HDR tone mapping
  - Smoke and particle effects

Epic's Rivalry demo using full Unreal Engine 4
Running in real-time on NVIDIA Tegra K1 with OpenGL ES 3.1 + AEP
https://www.youtube.com/watch?v=jRr-G95GdaM

Benchmark Previews

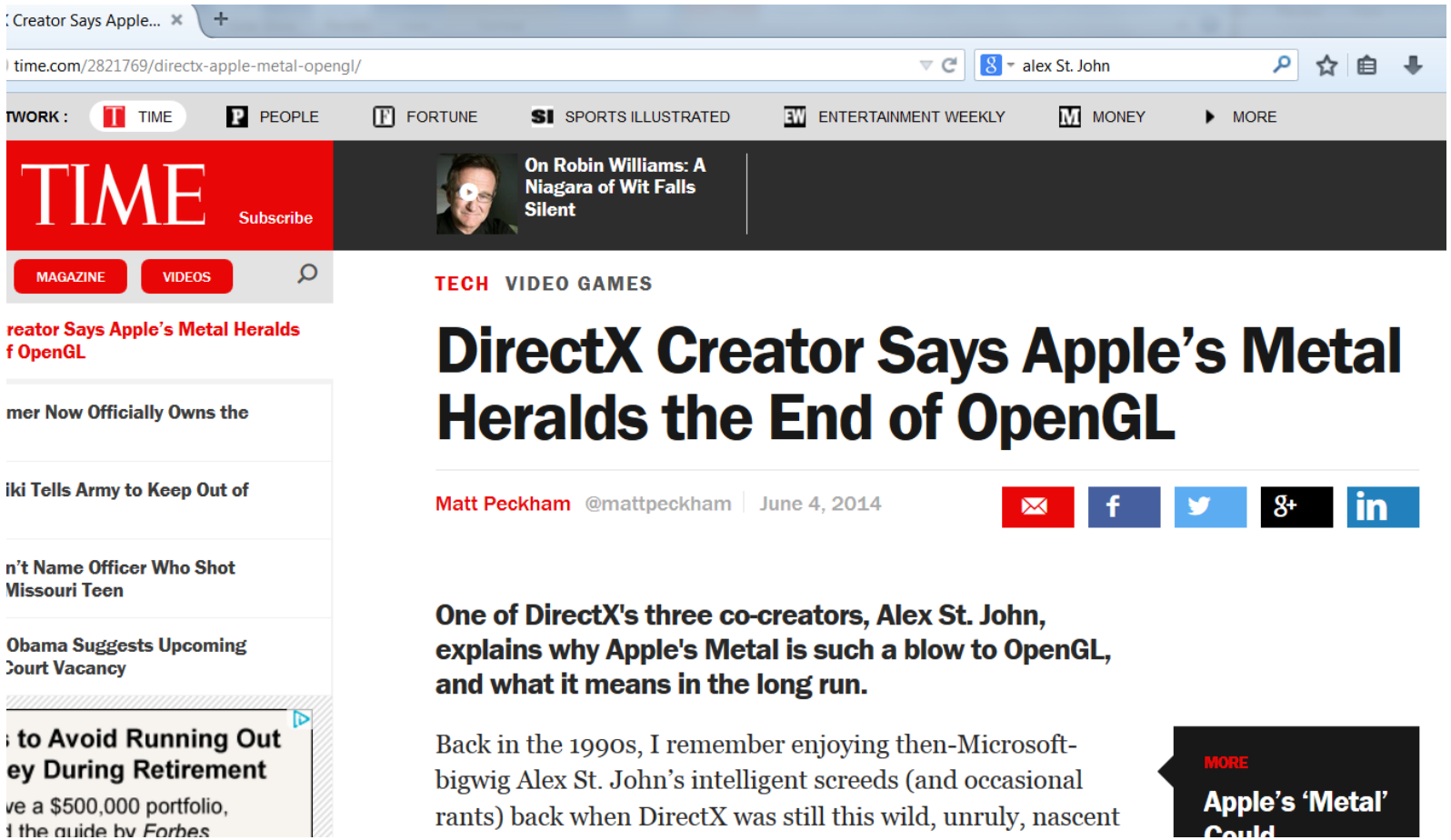
Laszlo Kishonti, Kishonti Informatics

Next Generation OpenGL Initiative



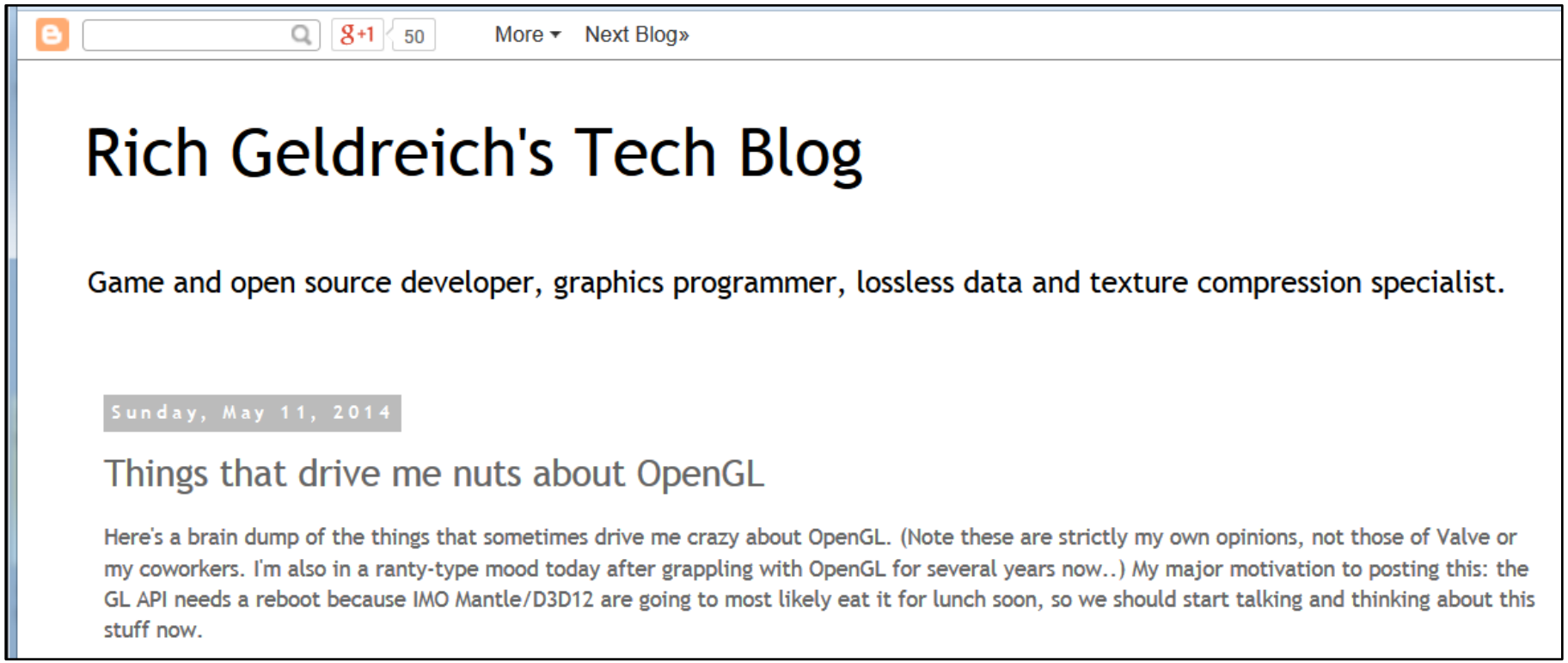
Tom Olson, ARM
GL Common TSG chair

OpenGL is a *huge* success



http://time.com/2821769/directx-apple-metal-opengl/

OpenGL needs to change



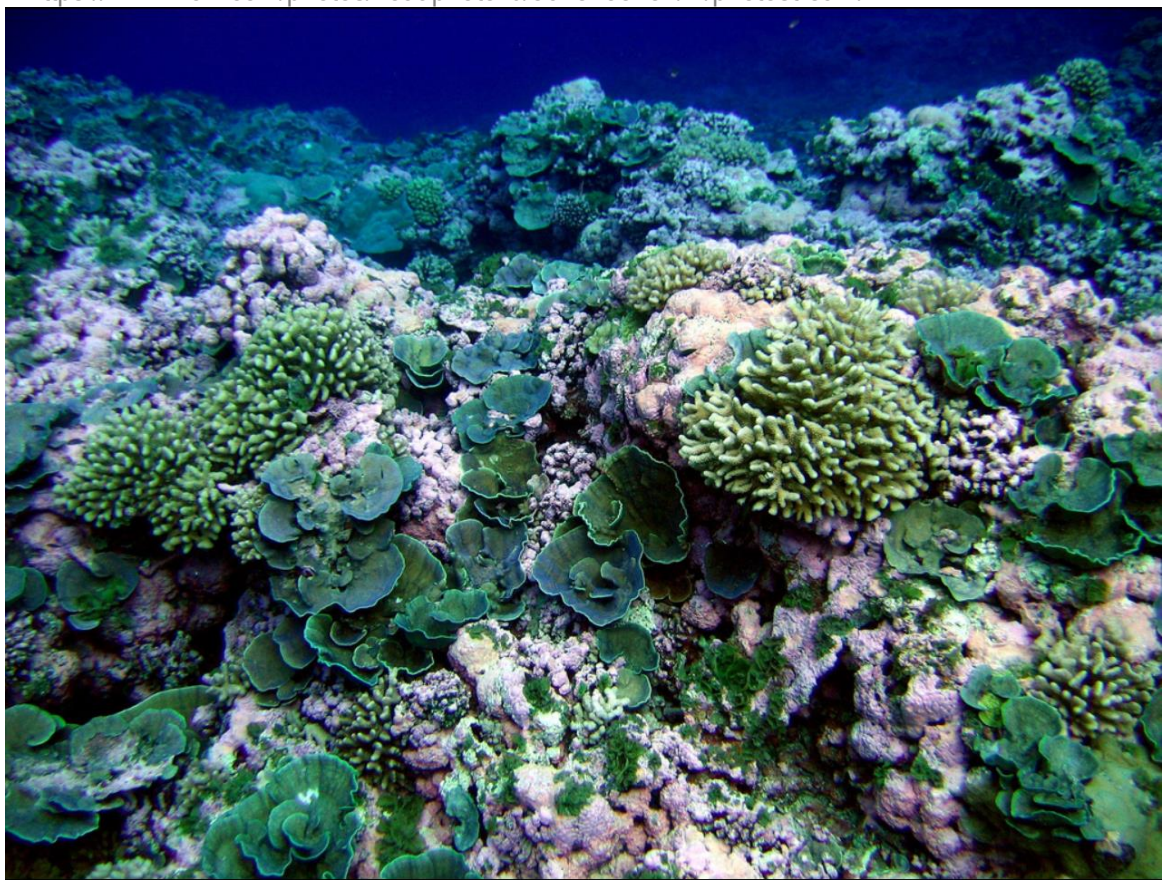
http://richg42.blogspot.com/2014/05/things-that-drive-me-nuts-about-opengl.html

Problems with OpenGL today

- **Programming model doesn't match architecture of modern GPUs**
  - Especially in mobile
- **Arcane and archaic syntax**
  - Twenty years of legacy cruft
  - Needless complexity
- **CPU intensive**
  - State validation, dependency tracking, error checking
  - Hard to predict where CPU load will occur – varies with implementation
- **Not multicore-CPU-friendly**
  - Primitive threading model, inconsistently implemented
- **Implementation variability**
  - Spec looseness, performance variation, driver bugs

Incremental change is not enough

https://www.flickr.com/photos/noaaphotolib/5015135752/in/photostream/



- GL has a good track record of adaptive evolution
- …but structural problems can't be fixed by accretion

Next Generation OpenGL Initiative

- **An open-standard, cross-platform 3D+compute API for the modern era**
  - Compatibility break with OpenGL
  - Start from first principles

- **Goals**
  - Clean, modern architecture
  - Multi-thread / multicore-friendly
  - Greatly reduced CPU overhead
  - Architecture-neutral – full support for tile-based as well as direct renderers
  - Predictable performance
  - Improved reliability and consistency between implementations

- **Key design principle: *explicit control***
  - Application is expected to tell the driver what it wants

KHRONOS GROUP

Status

- **Organized as a joint project of ARB and OpenGL ES working groups**
  - Chair: Tom Olson (ARM)
  - Language / IL group chair: Bill Licea-Kane (Qualcomm)
  - API spec editors: Graham Sellers (AMD) and Jeff Bolz (NVIDIA)

- **Working at very high intensity since June**
  - Very significant proposals and IP contributions received from members

- **Have committed to adopting a portable intermediate language (IL) for shaders**
  - Long-standing request from ISVs
  - Allows improved shader reliability and portability
  - Good target for machine-generated shaders and specialized HLLs
  - Provides a measure of IP protection for shaders

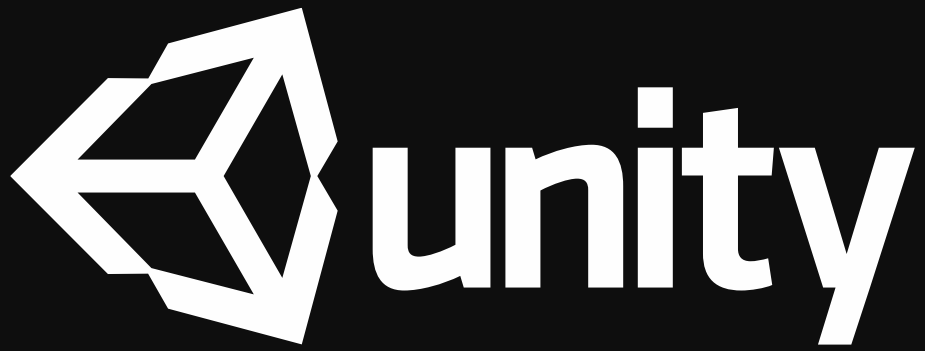
Who's on board?

- **Participants come from all segments of the graphics industry**
  - Including an unprecedented level of participation from game engine ISVs
  - Making rapid progress
  - New members are always welcome

OpenGL ecosystem support in Unity

Christophe Riccio, OpenGL Addict

unity

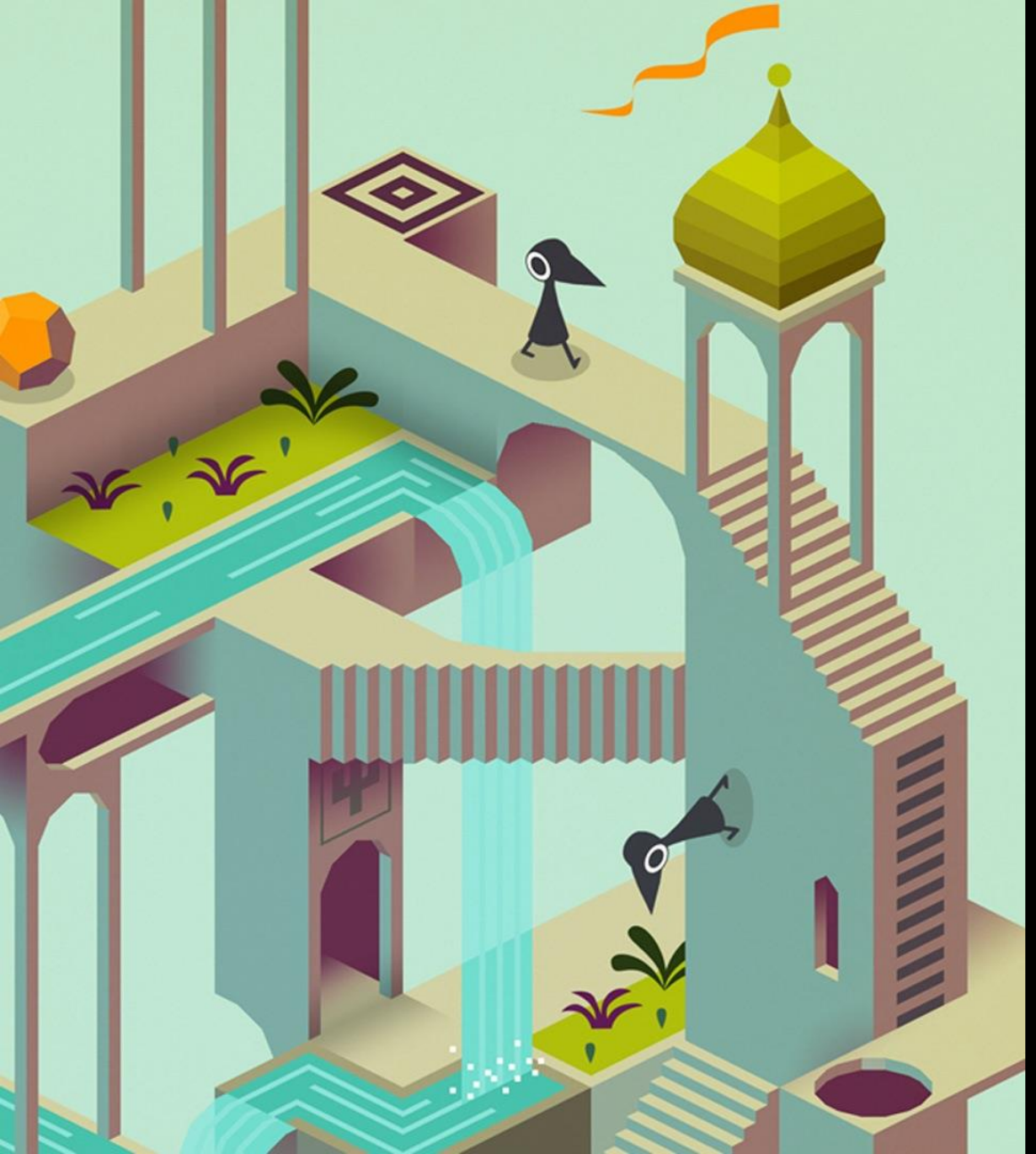


Democratizing games
development and success

A word from another of the team members…

We are super excited *to contribute and work with the Next Generation OpenGL Initiative, and bring our experience of low-overhead and explicit graphics APIs to build an efficient standard for multiple platforms and vendors in Khronos.* ***This work is of critical importance*** *to get the most out of modern GPUs on both mobile and desktop, and to make it easier to develop advanced and efficient 3D applications –* ***enabling us to build amazing future games*** *with Frostbite on all platforms.*

- **Johan Andersson**, Technical Director, Frostbite – Electronic Arts



Monument Valley
by ustwo

- iOS
- Puzzle game
- Multidisciplinary design agency

Unity benefits:

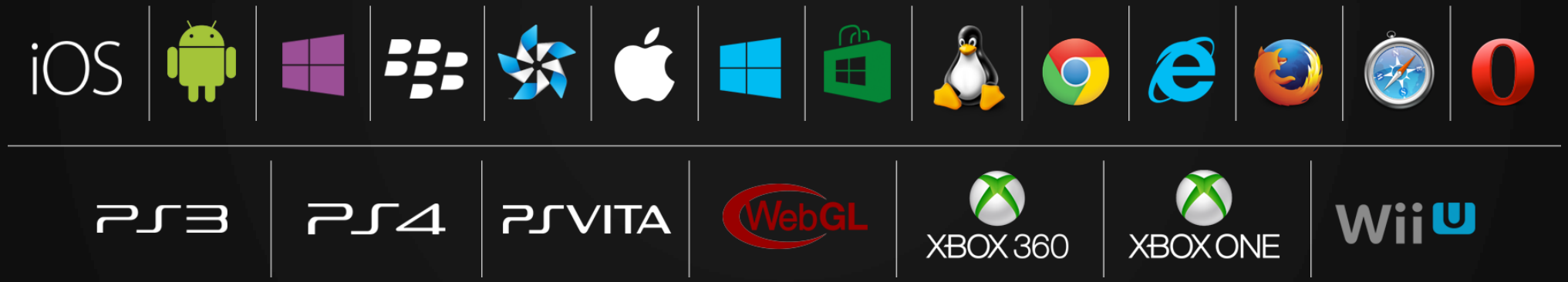
- Ease-of-use for the whole team
- Easy cross-platform production
- Custom editor scripts
- Unity Profiler

unity

Why OpenGL?

Why OpenGL?

Unity Multiplatform deployment



unity

Platform using on the OpenGL back-end

unity

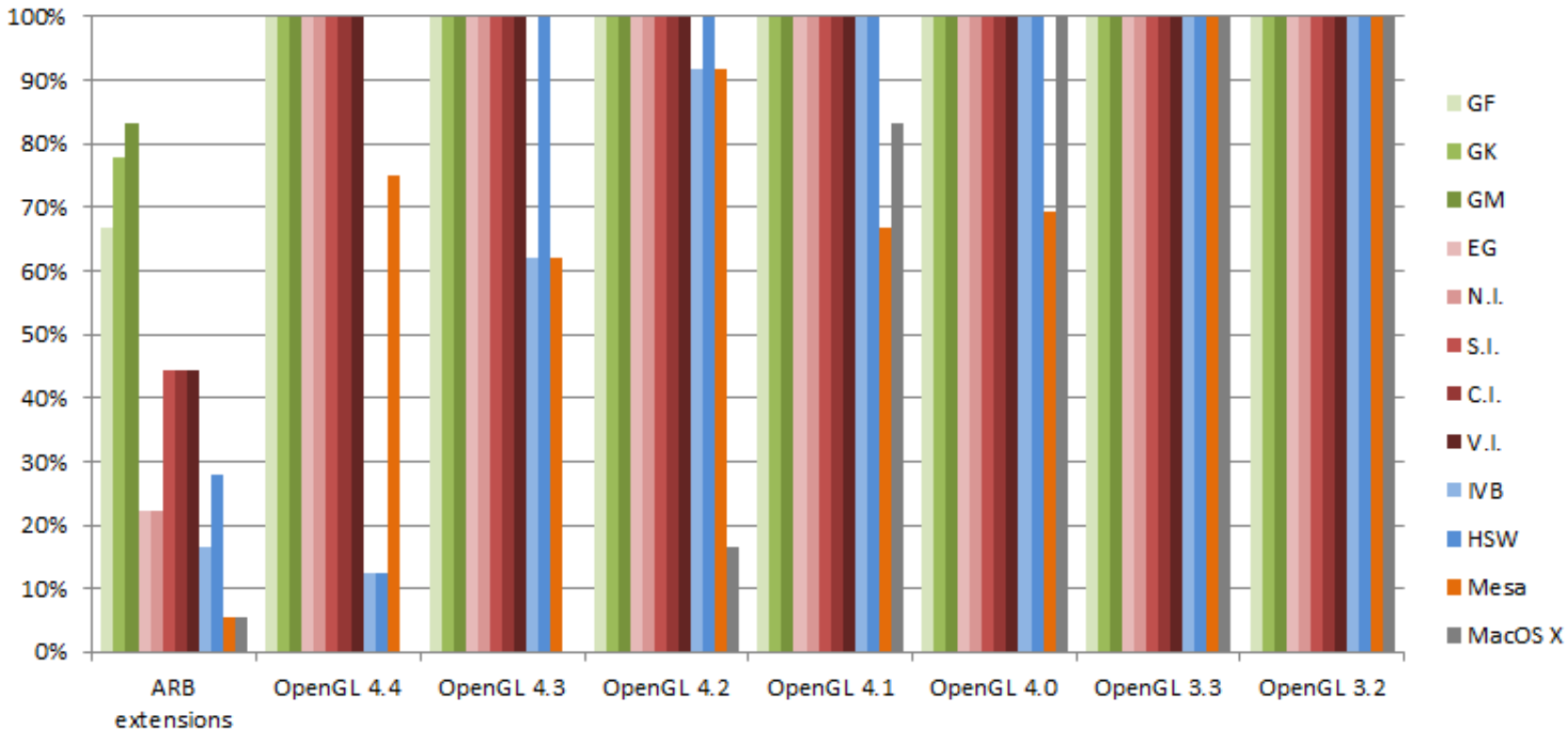
OpenGL is an ecosystem

- WebGL
- OpenGL ES
- OpenGL desktop

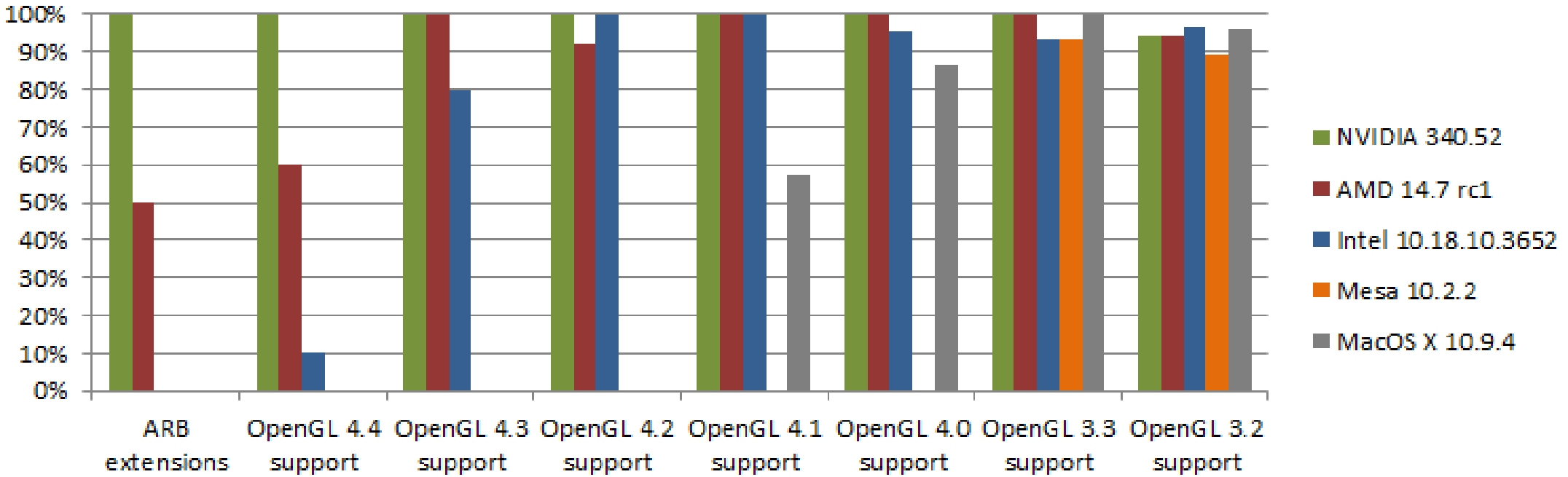
OpenGL drivers support

www.g-truc.net

Exposed extensions to all OpenGL 4 hardware on the last drivers available to date



What about effective support?



Re-evaluation the OpenGL implementations

The current ecosystem in Unity

OpenGL convergence

- Extensions:
  - GL_ARB_ES2_compatibility
  - GL_ARB_ES3_compatibility
  - GL_ARB_ES3_1_compatibility
- ES profile on desktop
  - WGL/GLX_EXT_create_context_es_profile
- EGL on desktop
  - This is the ultimate goal but the ecosystem is still W.I.P.
  - Software Emulators

There is at least one solution is available on each platform

unity

OpenGL convergence

A mix of feature levels and capabilities flags

Feature levels, essentially driven by our shader pipeline:

- OpenGL ES 2.0 / WebGL
- OpenGL ES 3.0 / OpenGL 3.2
- OpenGL ES 3.1 / OpenGL 4.3 or OpenGL 4.2 + compute shader stuff

“Player” base level: ES 2.0 / “*Editor”* base level: GL 3.2

The “subtleties” between the level: Handled by *capabilities*.

unity

Resolve difference between OpenGL versions

Skipping due to time constraints

It can be essentially resolve at initialization time or compilation time

- Capabilities: Initialized at initialization time
- Translation tables: from engine enums to appropriate OpenGL enums
- #ifdef to remove code on platform that can’t possibly run something

Working around drivers bugs

- Part of the capability flags: Disable the capability if it is buggy
  - Micro managing bugs doesn't scale
- Fallback to lower feature level
  - If GL4 feature level doesn't work, use GL3. If GL3 doesn't work use GL2
- Have shader fallbacks
  - We are going as far as showing a pink shaders
- Document the bug workarounds: Platform, vendor, what's wrong
  - They will be a point where no body remember why this ugly workaround is here
  - They will be a point where that workaround should be removed because not useful anymore

Skipping due to time constraints

unity

“Pro tips”

Skipping due to time constraints

- Working on desktop OpenGL?
  - Have AMD, Intel and NVIDIA GPUs in your dev machine all the time
  - Plug a screen per GPU
  - Change main display to select a different GPU
- Starts from OpenGL ES to OpenGL desktop
  - Convergence is ensured by desktop the OpenGL specifications
  - Use a single OpenGL back-end to avoid divergence
- Document drivers bugs and workaround in your code
  - Otherwise we won’t know when we could discard workaround!
- Have an automated graphics tests farm
- Use (or implement!) KHR_debug
- Thread the OpenGL back-end

The next ecosystem

The next ecosystem

“♫ Everything is awesome ♪” The Lego Movie, 2014

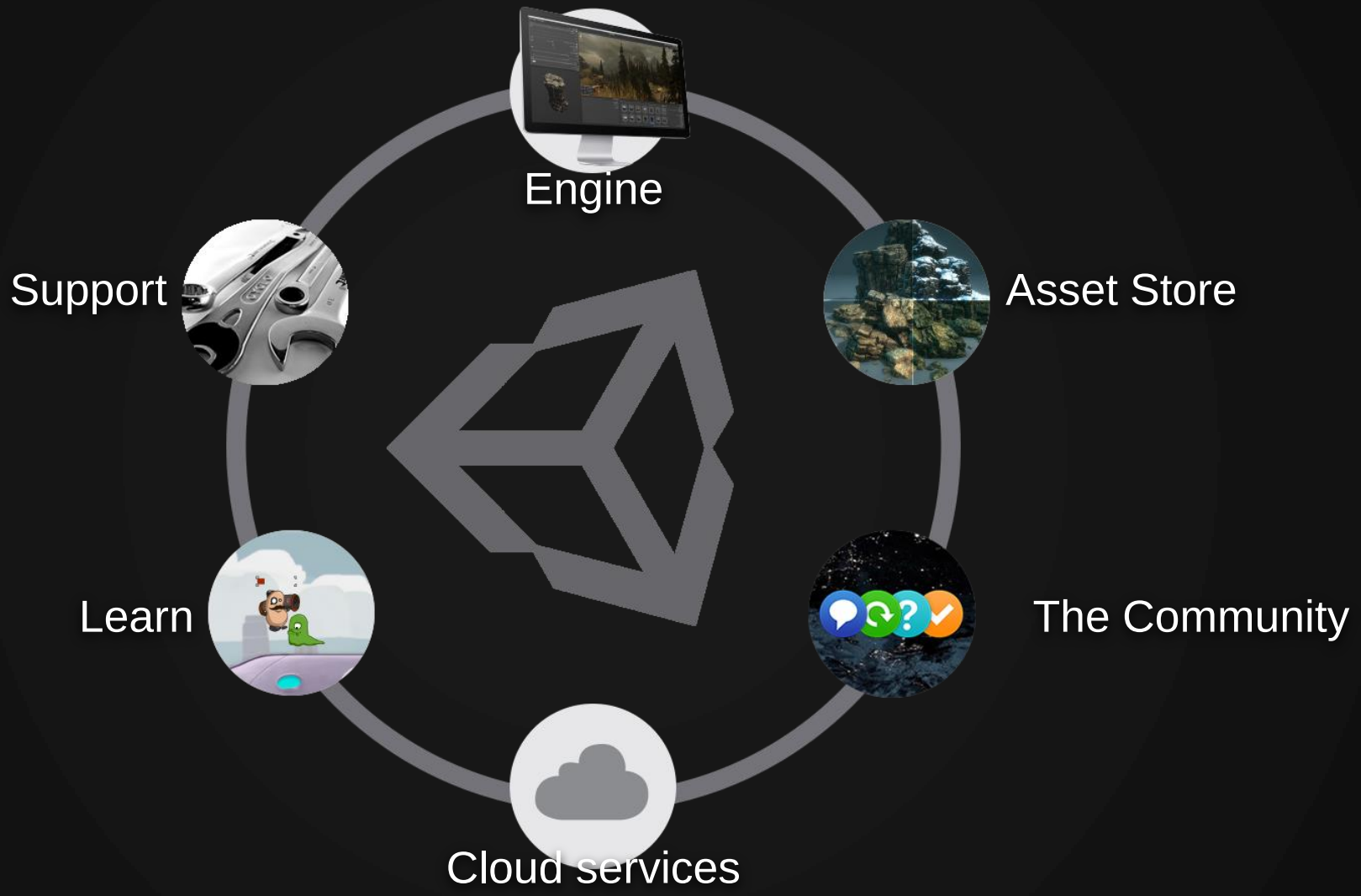
unity

Unity in numbers:

- 600,000 + monthly active users
- 750,000 + registered Asset Store users
- Close to 3 million registered developers

- Real time graphics is not a matter of a few millions people anymore
- A potential target is 6 billions people

unity


Engine
Asset Store
The Community
Cloud services
Learn
Support

Solving issues globally

(Instead of individually)

- Dealing with issues individually (current OpenGL ecosystem):
  - Per-vendor
  - Other programmers will hit the same issues
  - N => 1 relation
- Dealing with issues globally:
  - A programmer reports one issue, it propagates to all programmers and vendors
  - 1 => N relation
- Example of globally solving issues: WebGL conformance test
  - Open sourced
  - Users can submit tests
  - WebGL platforms are reliable! (We don’t have per platform wordaround flags!)

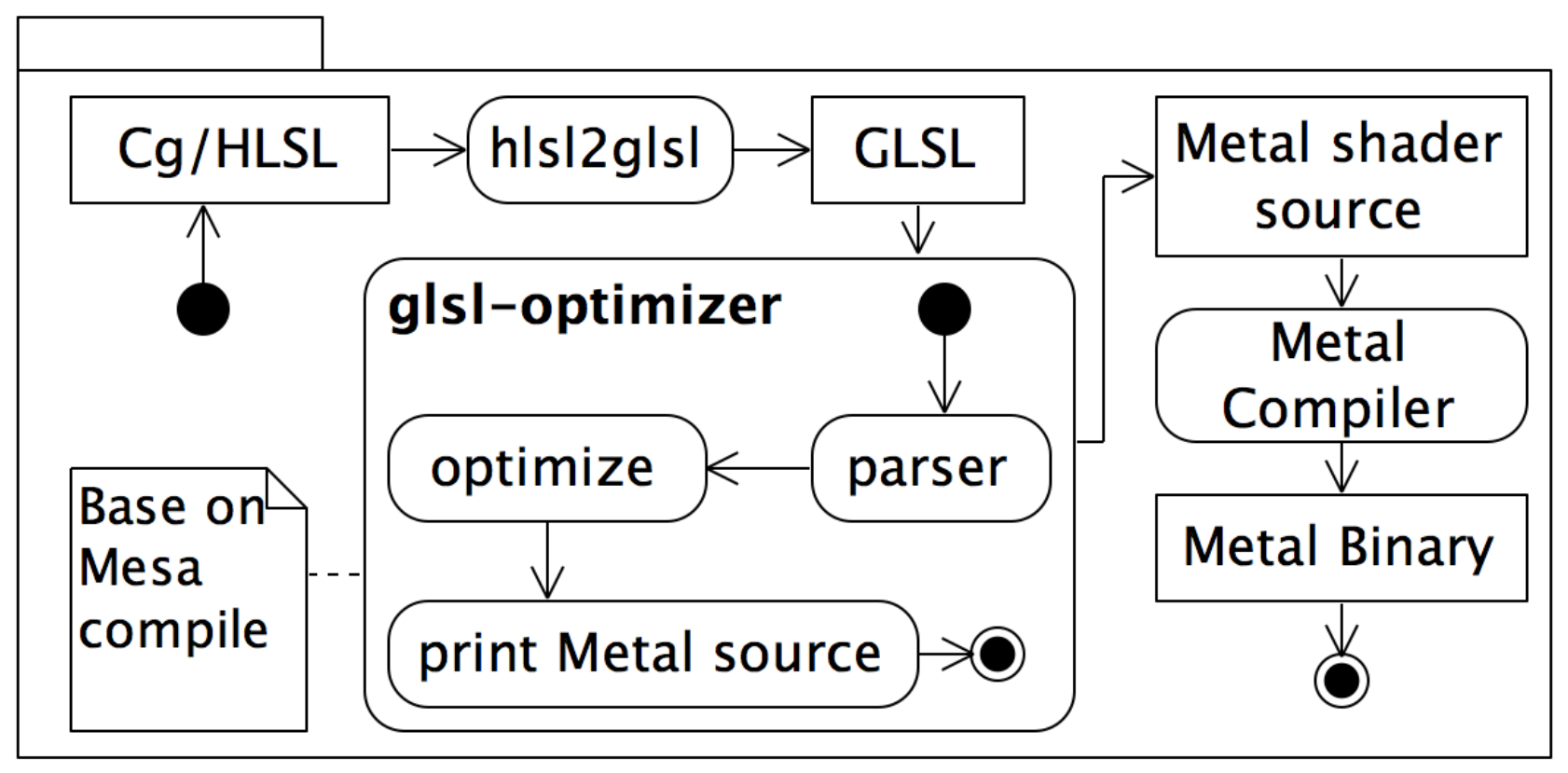
Intermediate Shading Language

- With current GLSL:
  - 4 ES versions !
  - 8 core profile versions !!
  - 12 compatibility profile versions !!!
- Issue 1: GLSL couples syntax sugar and hardware features
- Issue 2: Vendors don’t fix some issues to avoid breaking current software

How do we deal with this in our shader compilation pipeline?

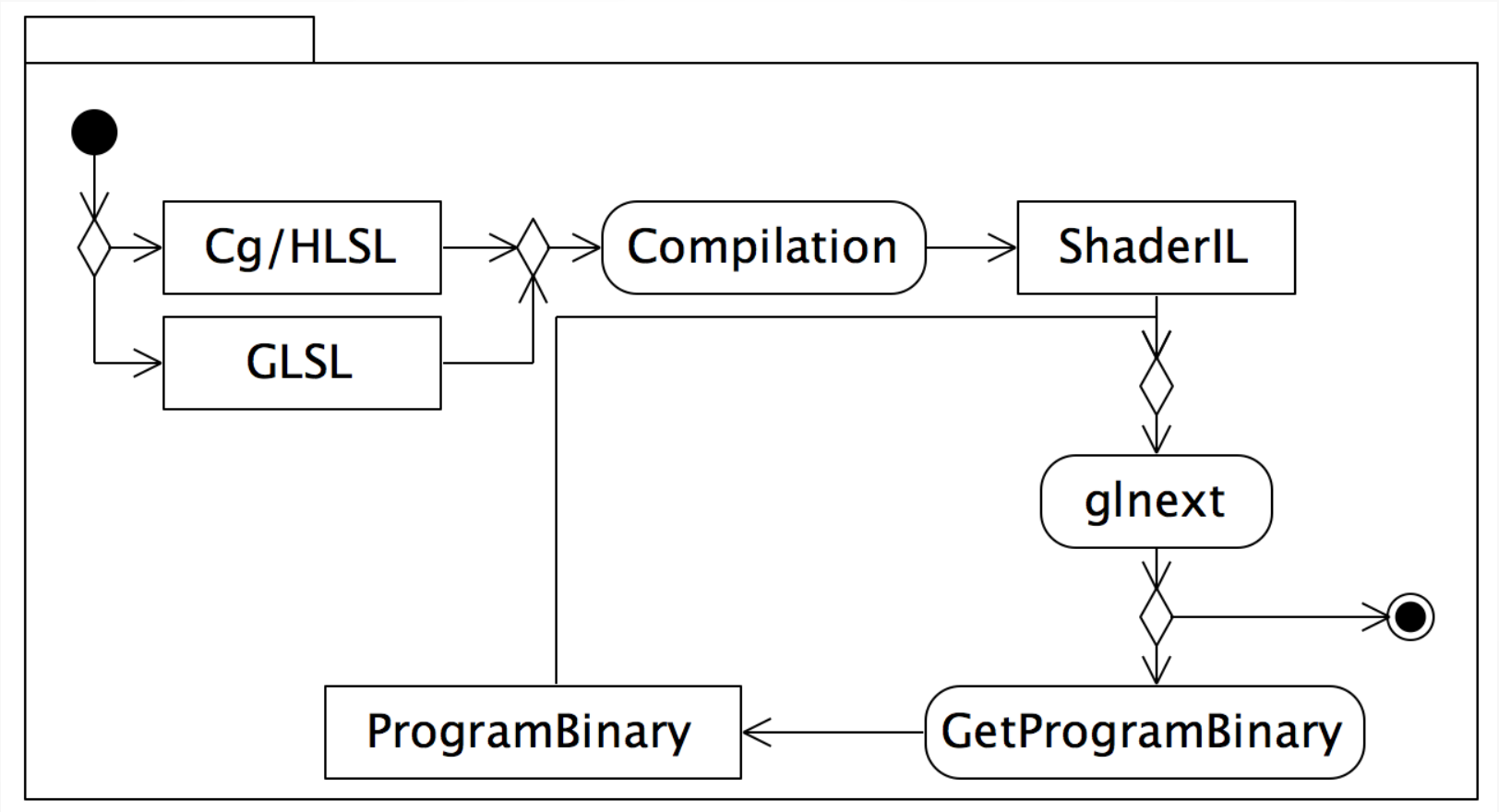
unity

Example of a current shader pipeline

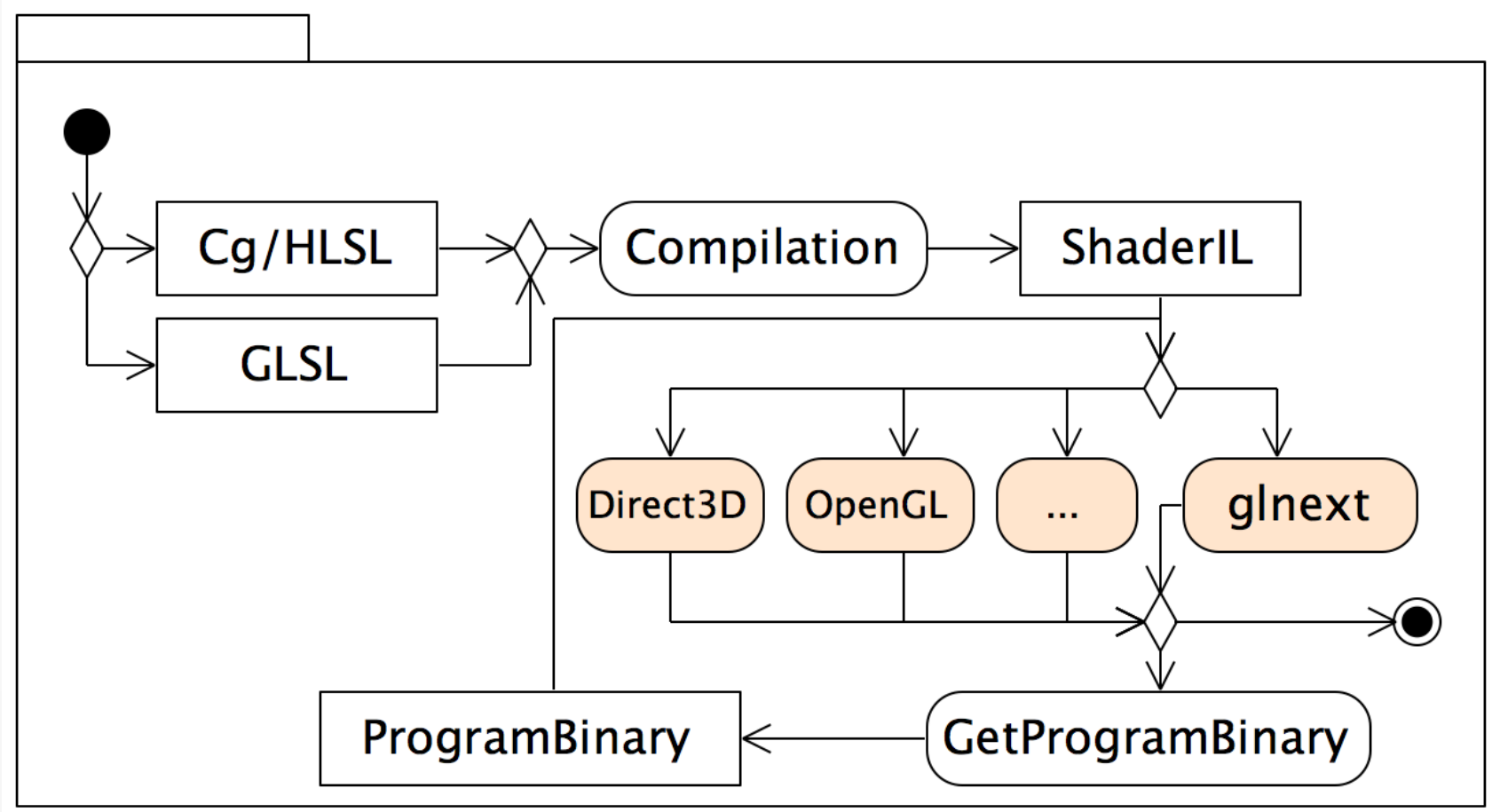


unity

Wish shader pipeline

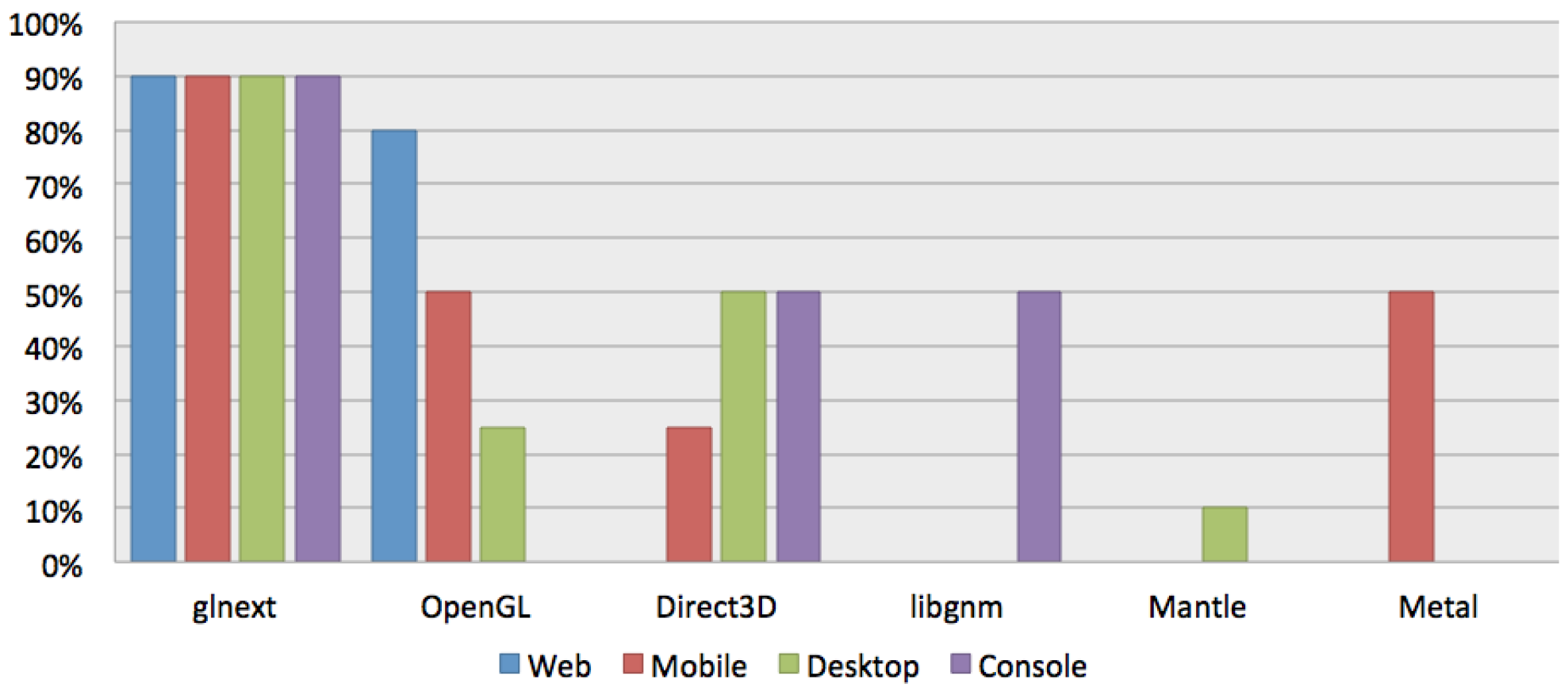


Bigger wish shader pipeline



unity

"Relevance" of ecosystems per platform



unity

Special thanks to:

- Engine representatives at Khronos:
  - Timothy Lottes (Epic)
  - John McDonald (Valve)
  - Patrick Doane (Blizzard)
  - Johan Andersson (Frostbite)
- Aras Pranckevičius (Graphics Plumber at Unity)
  - “Ship it!”
- The OpenGL community
  - Ask your hardware vendors and platform vendors for roadmaps, show your interest!

unity

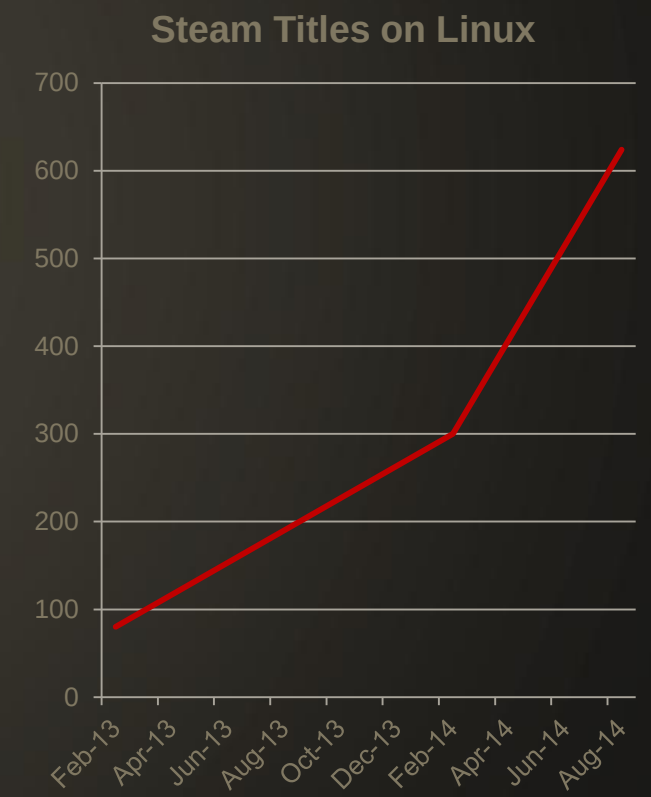
Join and contribute!

Valve <3s Open Standards

John McDonald

Open Technologies are Critical

- Over the past few years, Valve has opened or released:
  - Steam for Linux
  - Steam Runtime
  - SteamOS
  - VOGL
- And has contributed funding to the development of:
  - Glslang
  - LunarGLASS / LunarGOO

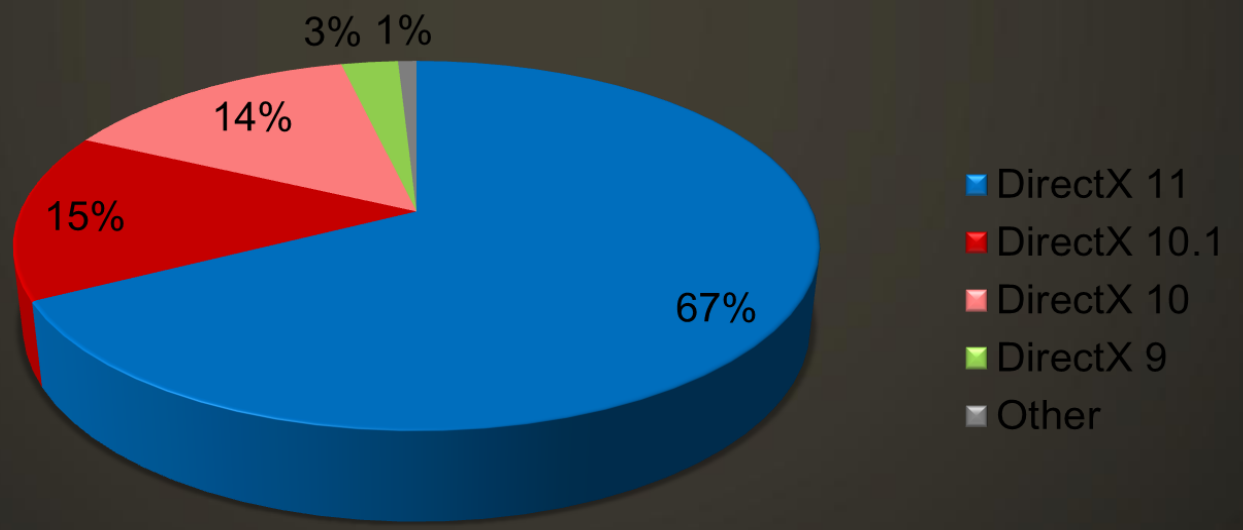


Portal on Android

- Partnered with NVIDIA to release Portal on NVIDIA Shield
- Uses OGL ES 2.x for rendering
- Full desktop assets
- Currently has 4.8 stars on Play Store

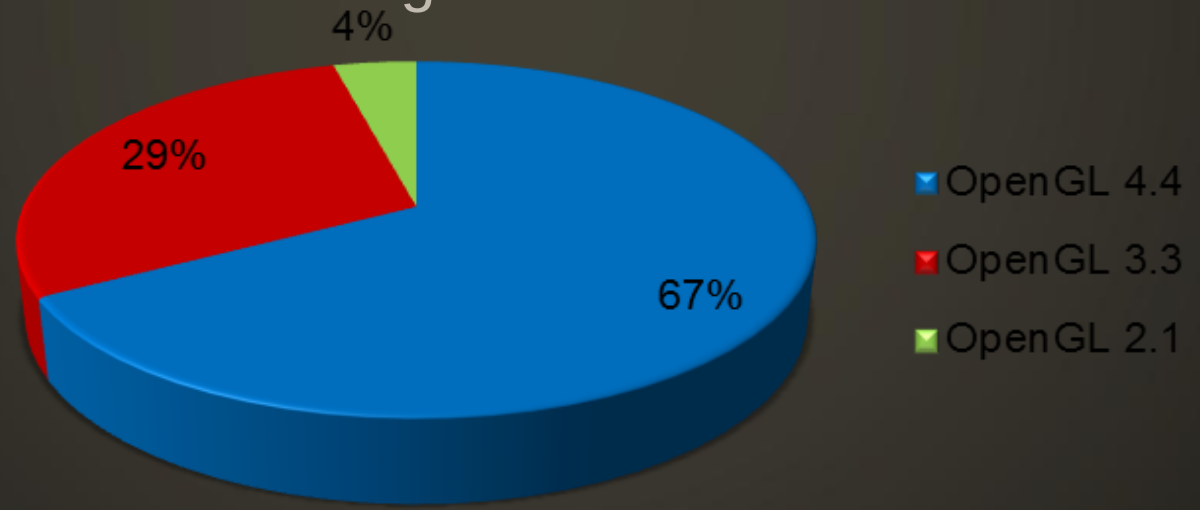
Graphics Today

- Direct3D is the primary used graphics API for desktop gaming
- Available only on Windows
- Tied to OS version—D3D10/11 are Vista+



OpenGL Today

- Available on Windows, OSX & Linux
  - ES is available on ~100% of mobile
- Developers can target HW features
- Significant performance advantages over D3D



glnext is shaping up to be *amazing*

- glnext will have the expected features and control of a modern API
- And the portability story of OpenGL
- OpenGL is already a critically important component of SteamOS
  - We fully anticipate that glnext will continue this tradition.

Questions?