JogAmp Fast Media & Processing

Across devices – Desktop & Mobile

SIGGRAPH 2014 – Vancouver

August 11, 2014

Julien Gouesse
Mark Raynsford
Xerxes Ranby

Harvey Harrison
Rami Santina
Sven Gothel
Wade Walker

Info

Slides and BOF Video

will be made available

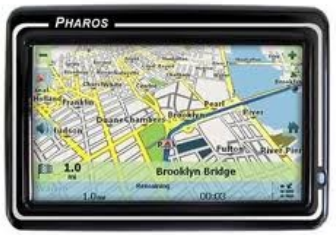
on

Jogamp.org.

What is *it*?

JogAmp

GlueGen - JOGL – JOCL – JOAL ...



Technology Enabler ..

Technology enabler across Platforms for high performance:

Feature	*Module*	*Native API*
Audio	**JOAL**	**OpenAL**
Graphics	**JOGL**	**OpenGL**
Multimedia	**JOGL**	**FFMpeg / libAV, ...**
Compute / Processing	**JOCL**	**OpenCL**

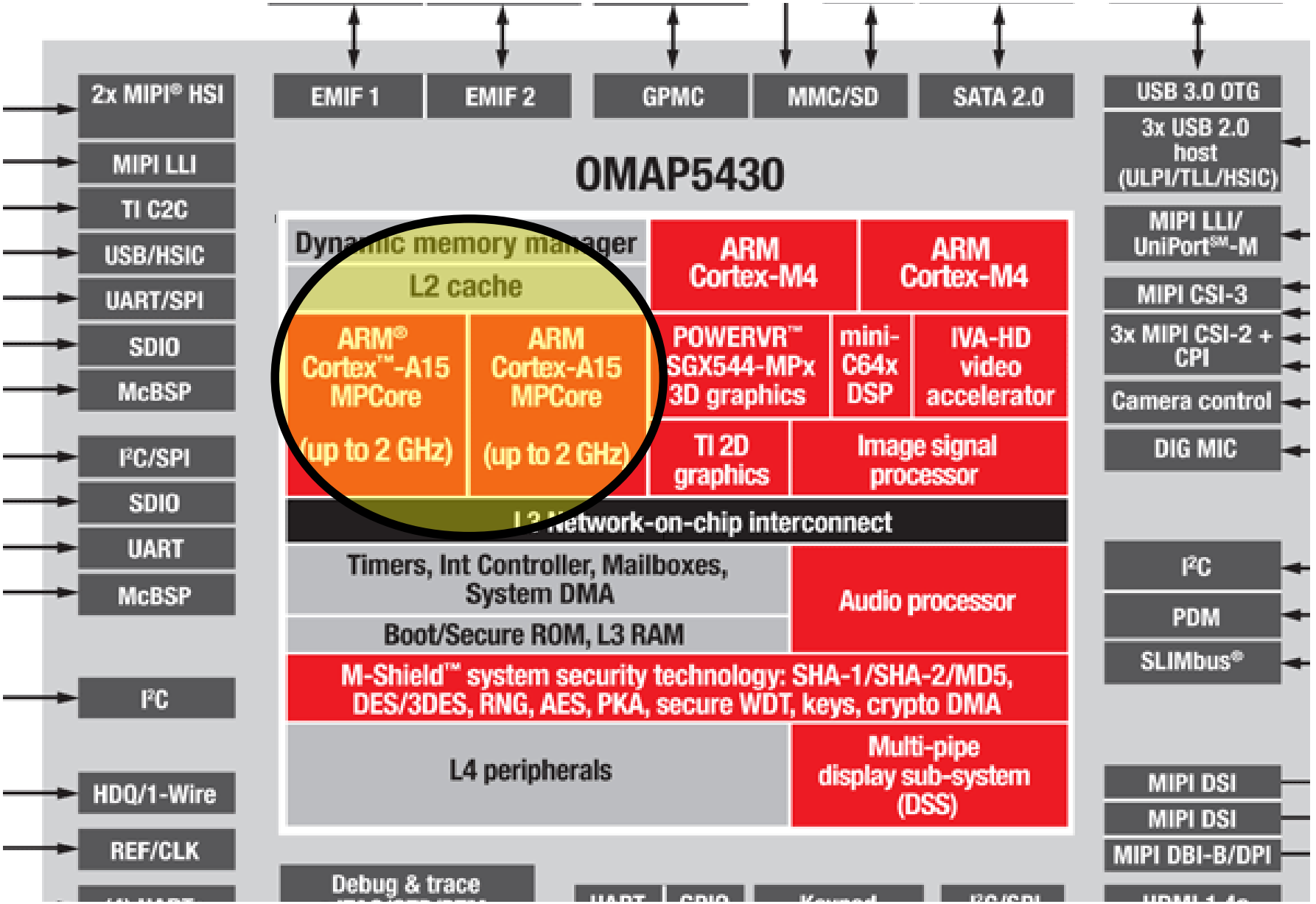
.. on top of a VM

Running on top of a virtual machine for

- CPU Abstraction
- Basic access to OS features
  - Multithreading
  - I/O incl. Network
  - …

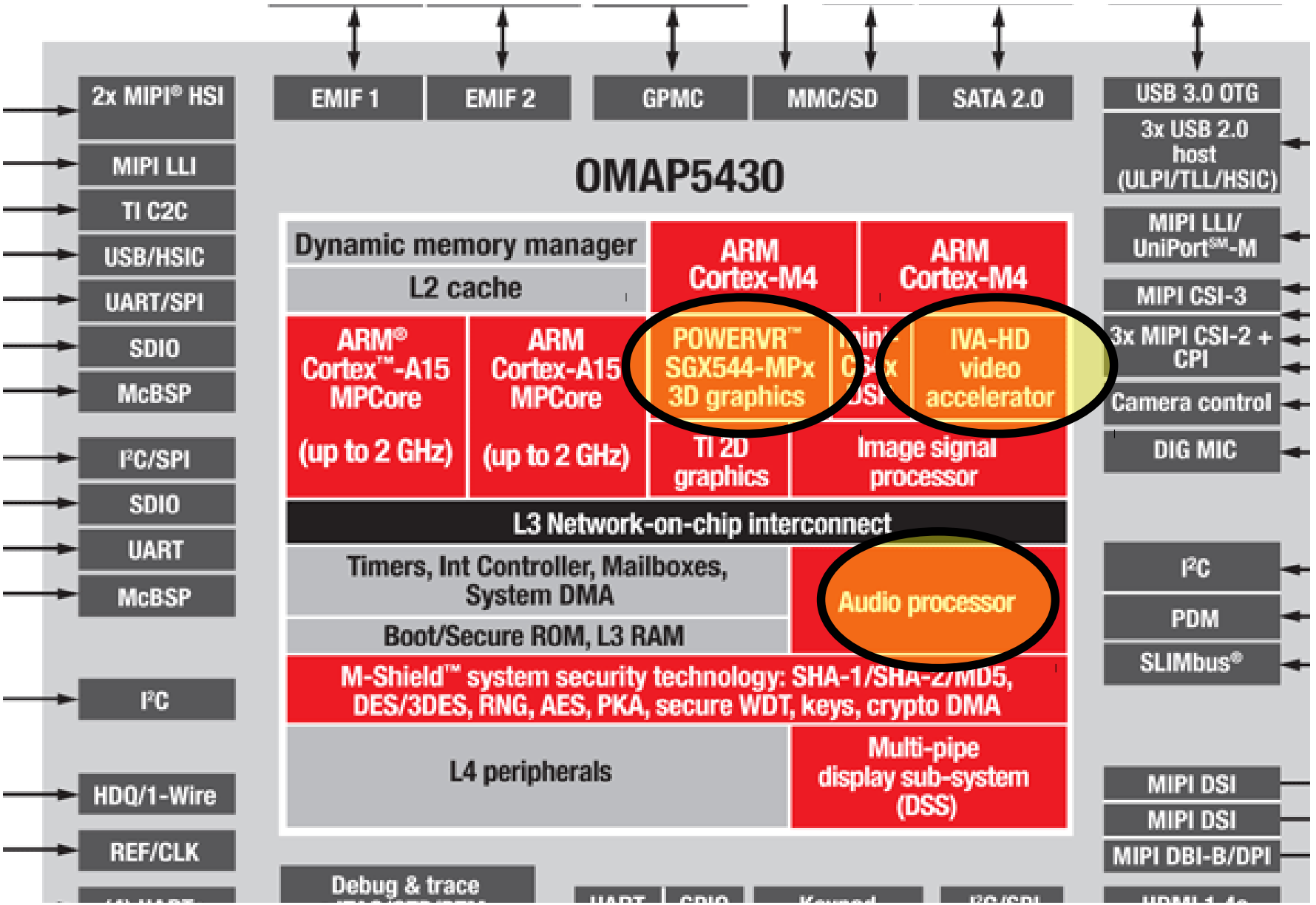
JogAmp complements the VM with access to named hardware functionality via standardized open APIs.

CPU Abstraction



What is *it*?

Feature Abstraction



What is *it*?

Doing *it* for 11 years ..

- 2003-06-06 GlueGen, JOGL, JOAL
- 2008-04-30 JOGL Release 1.1.1
- 2009-07-24 JOCL
- 2009-11-09 Independent Project
- 2010-05-07 JogAmp Project Name, Server, ..
- 2010-11-23 JogAmp RC 2.0-rc1
- 2013-07-17 JogAmp Release 2.0.2
- 2014-08-07 JogAmp Release 2.2.0

SIGGRAPH2014

Who does *it*?

As we have a core team developing *JogAmp*,
only a fine line distinguishes them from the ***users***:

- Software Developer
  - independent
  - employees
  - students
- Interest Groups
  - Research / Science
  - Product-Dev. / Companies
  - Education

Legal / Risks of *it*

- New BSD License, and similar ..
- No vendor risk (End of Business → EOL)
- Free to maintain yourself, costly but possible
- Available source code
  - Documentation
  - Debugging / Security
  - Maintenance

Motivation doing *it*?

- Ideology
  - Freedom to chose target platform
  - Run anywhere ...
- Interesting Problems
  - Efficiency / High Performance
  - Platform abstractions
  - Graphics, Audio, Multimedia & Computing
  - Use cases across domains ..
- Interesting People
  - .. across domains
  - .. different level of expertise

How do we do *it*?

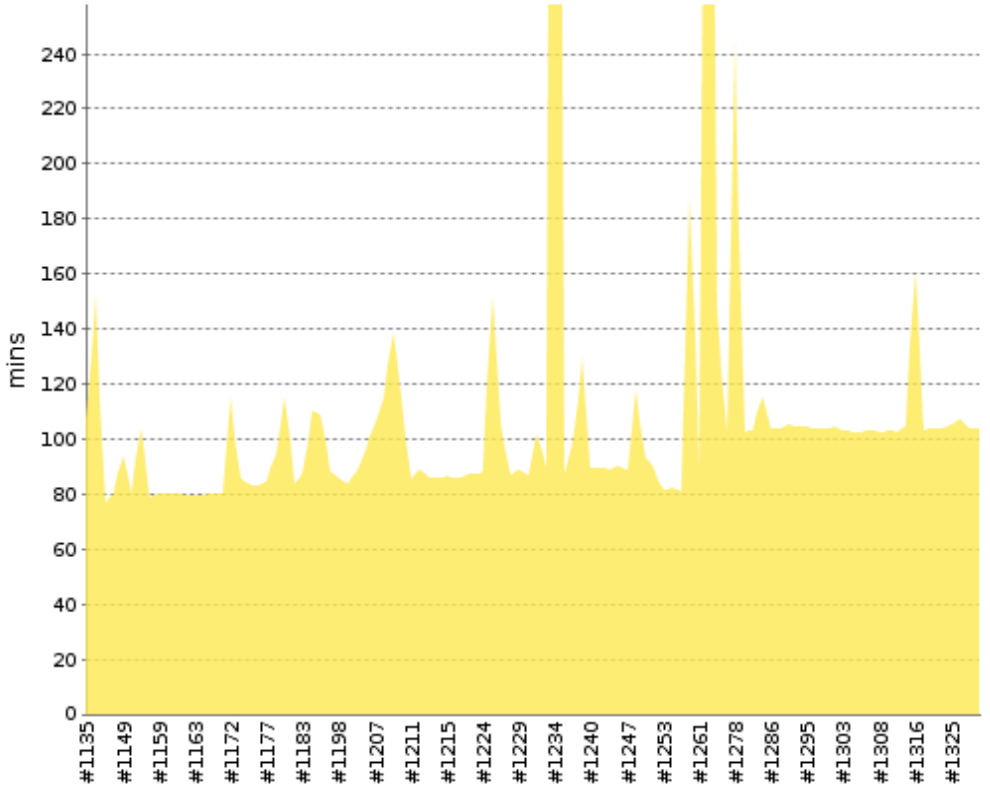
- Free and Open Tools
  - GIT SCM
  - Bugzilla Bugtracking
  - Jenkins Build Server
  - Mailinglist/Forum,..
- Quality
  - Unit Test Coverage
  - Public Bugreports
  - Publicly suggested new Use Cases

Continuous Progress

Overall Build Monitoring

Build Time Trend

Build ↑	Duration	Slave
#1329	1 hr 43 min	master
#1328	1 hr 43 min	master
#1327	1 hr 47 min	master
#1325	1 hr 44 min	master
#1322	1 hr 43 min	master
#1320	1 hr 43 min	master
#1317	1 hr 42 min	master
#1316	2 hr 40 min	master
#1315	1 hr 44 min	master
#1314	1 hr 42 min	master
#1309	1 hr 42 min	master
#1308	1 hr 42 min	master
#1307	1 hr 43 min	master
#1305	1 hr 42 min	master
#1304	1 hr 42 min	master



Continuous Progress

Overall Build Monitoring

Test Result

15 failures

10,802 tests

Configuration Name	Duration	All	Failed	Skipped
label=android-armv7-img	11 ms	1	0	0
label=macosx-10_6-x86_64-nvidia	32 min	975	6	0
label=win7-x86_64-nvida	26 min	986	1	0
label=win7-x86_64-amd	26 min	986	0	0
label=solaris-x86_64-nv	28 min	957	0	0
label=linux-armv7hf-img	11 ms	1	0	0
label=linux-x86_64-nvidia	25 min	986	0	0
label=solaris-x86_32-nv	29 min	986	1	0
label=linux-armv7-img	11 ms	1	0	0
label=linux-x86_32-amd	11 ms	1	0	0
label=macosx-10_7-x86_64-nvidia_jre7	46 min	979	5	0
label=linux-x86_32-nvidia	26 min	986	0	0
label=win7-x86_32-amd	29 min	986	0	0
label=win7-x86_32-nvida	28 min	986	1	0
label=linux-x86_64-amd	26 min	985	1	0

Continuous Progress

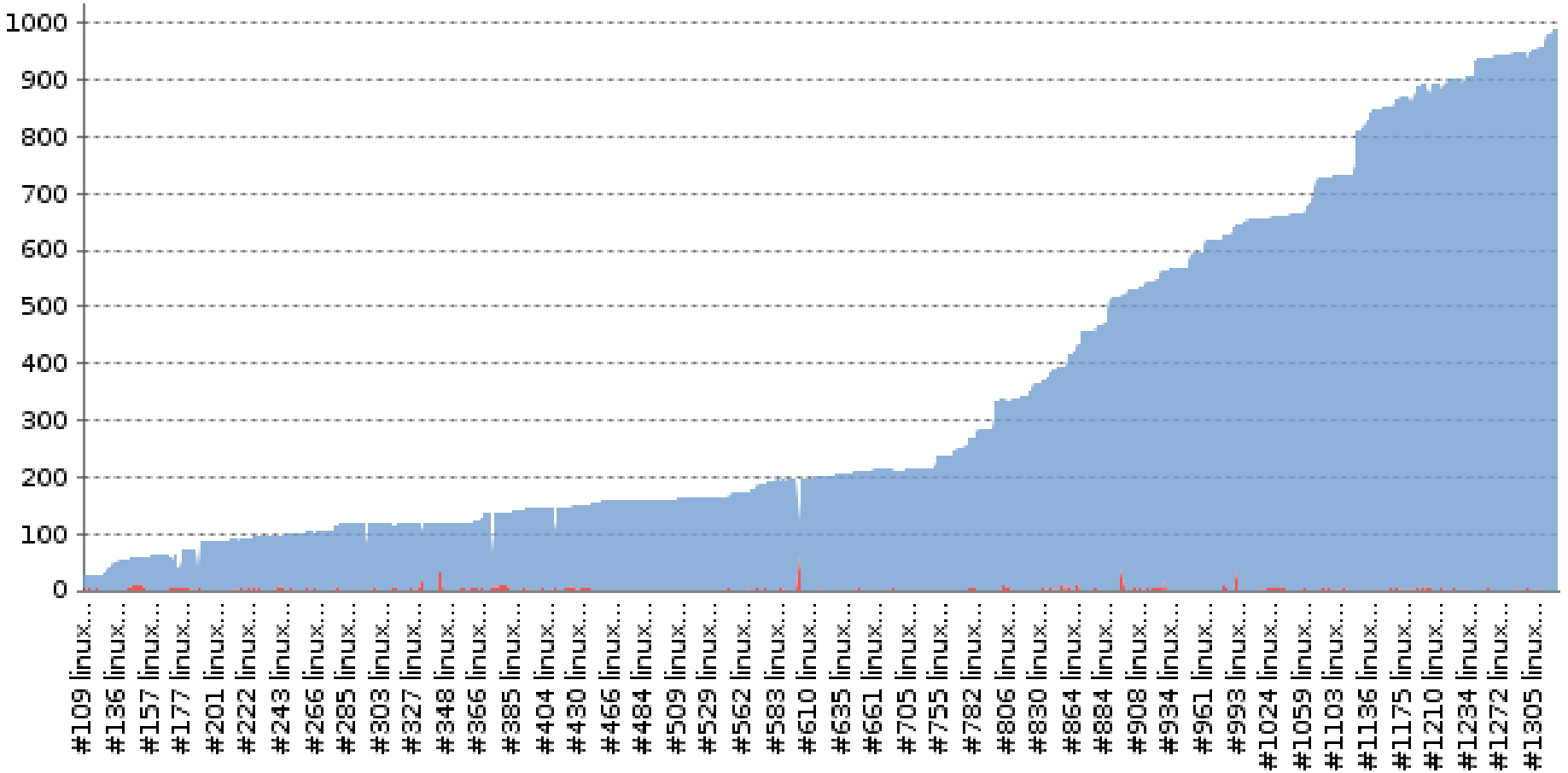
Unit Test Monitoring

All Tests

Package	Duration	Fail	(diff)	Skip	(diff)	Pass	(diff)	Total	(diff)
com.jogamp.opengl.test.junit.graph	25 sec	0		0		10		10	
com.jogamp.opengl.test.junit.jogl.acore	5 min 16 sec	0		0		274		274	
com.jogamp.opengl.test.junit.jogl.acore.anim	18 sec	0		0		10		10	
com.jogamp.opengl.test.junit.jogl.acore.ect	59 sec	0		0		40		40	
com.jogamp.opengl.test.junit.jogl.acore.gleis	1 min 47 sec	0		0		47		47	
com.jogamp.opengl.test.junit.jogl.awt	2 min 22 sec	0		0		65		65	
com.jogamp.opengl.test.junit.jogl.awt.text	1.4 sec	0		0		2		2	
com.jogamp.opengl.test.junit.jogl.caps	24 sec	0		0		29		29	
com.jogamp.opengl.test.junit.jogl.demos.es1.newt	6.7 sec	0		0		3		3	
com.jogamp.opengl.test.junit.jogl.demos.es2.awt	16 sec	0		0		19		19	
com.jogamp.opengl.test.junit.jogl.demos.es2.newt	26 sec	0		0		14		14	
com.jogamp.opengl.test.junit.jogl.demos.gl2.awt	6.2 sec	0		0		9		9	
com.jogamp.opengl.test.junit.jogl.demos.gl2.newt	8.2 sec	0		0		5		5	
com.jogamp.opengl.test.junit.jogl.demos.gl3.newt	6.5 sec	0		0		4		4	
com.jogamp.opengl.test.junit.jogl.glsl	13 sec	0		0		15		15	
com.jogamp.opengl.test.junit.jogl.glu	4.3 sec	0		0		2		2	
com.jogamp.opengl.test.junit.jogl.math	3.2 sec	0		0		104		104	
com.jogamp.opengl.test.junit.jogl.newt	11 sec	0		0		2		2	
com.jogamp.opengl.test.junit.jogl.offscreen	16 sec	0		0		9		9	
com.jogamp.opengl.test.junit.jogl.perf	1 min 42 sec	0		0		21		21	
com.jogamp.opengl.test.junit.jogl.tile	2 min 33 sec	0		0		46		46	
com.jogamp.opengl.test.junit.jogl.util	27 sec	0		0		15		15	
com.jogamp.opengl.test.junit.jogl.util.texture	2 min 11 sec	0		0		89		89	
com.jogamp.opengl.test.junit.newt	47 sec	0		0		34		34	
com.jogamp.opengl.test.junit.newt.event	3 min 15 sec	0		0		38		38	
com.jogamp.opengl.test.junit.newt.mm	2 min 32 sec	0		0		21		21	
com.jogamp.opengl.test.junit.newt.parenting	1 min 28 sec	0		0		30		30	

Continuous Progress

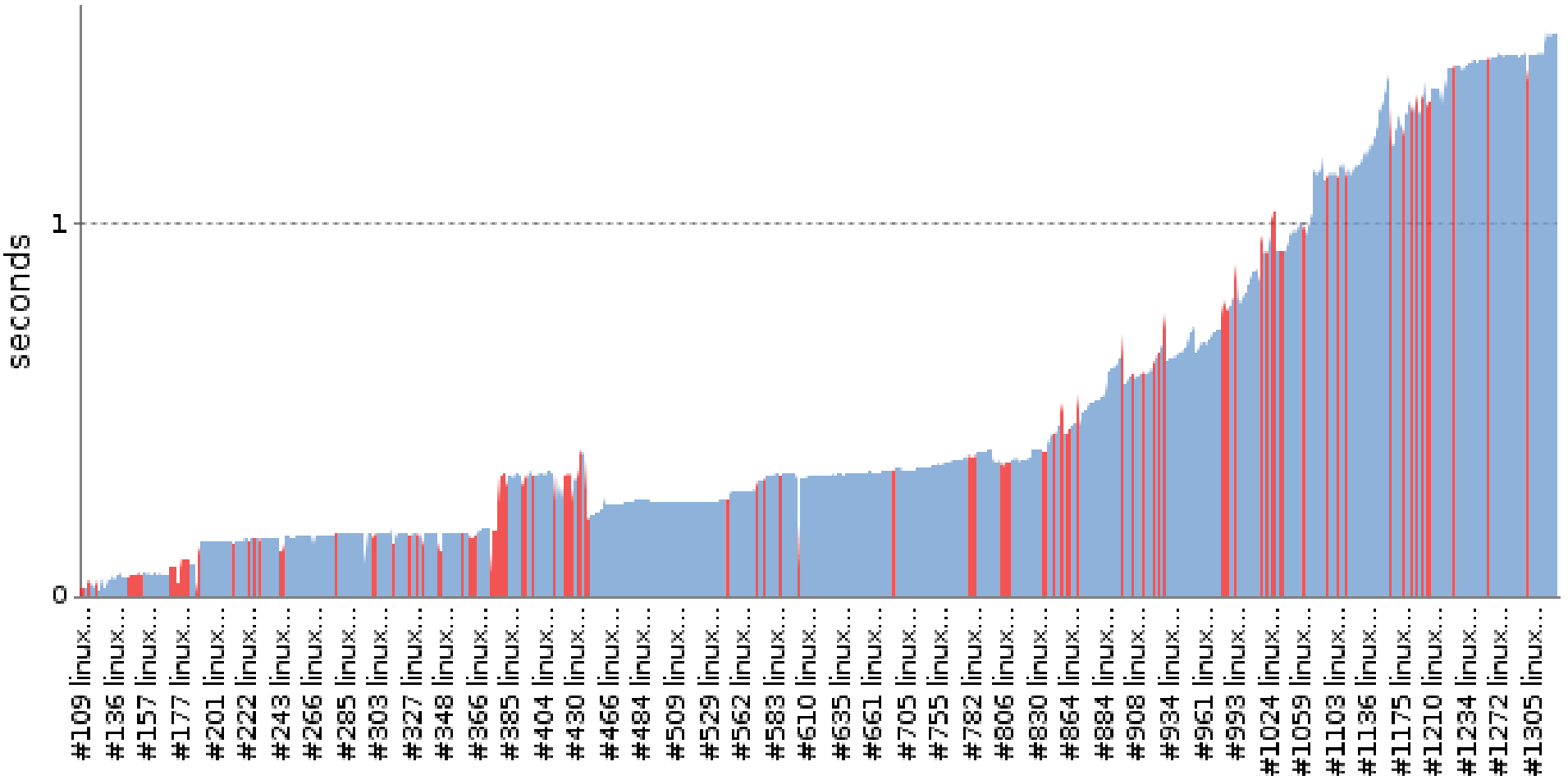
No performance regressions ...



Doing *it..*

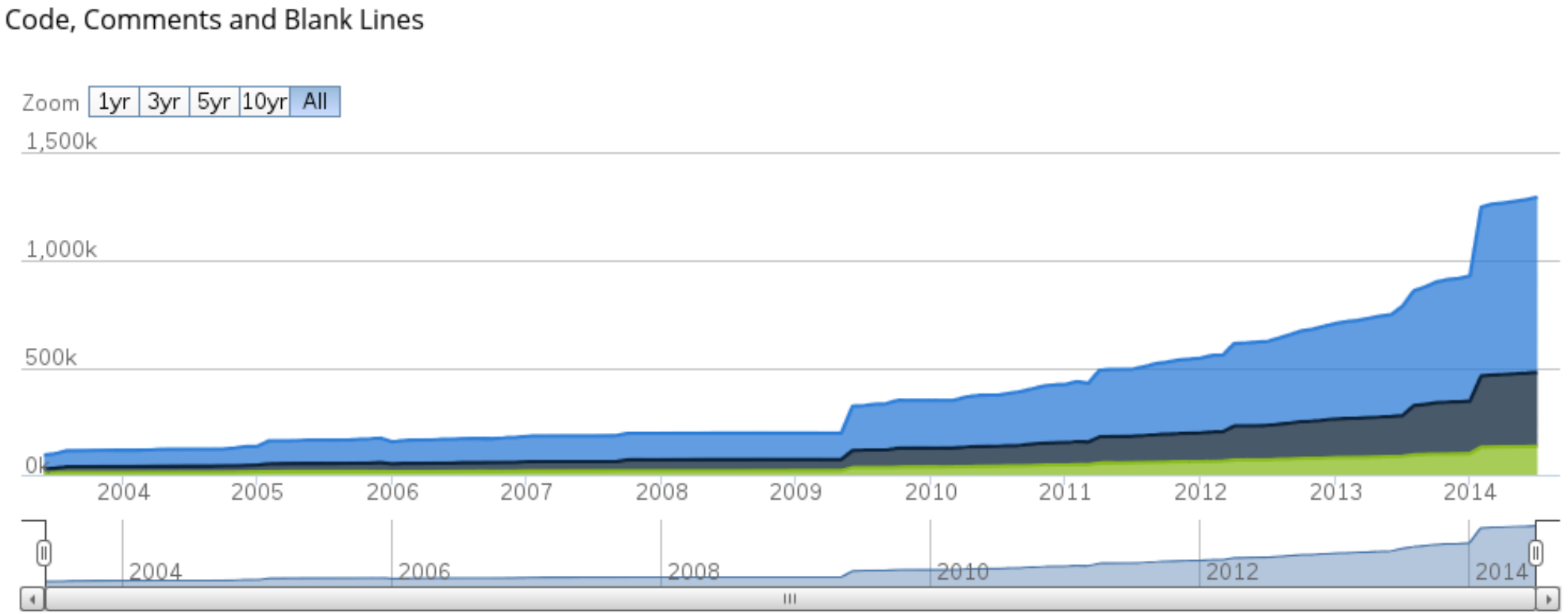
Continuous Progress

No performance regressions ...



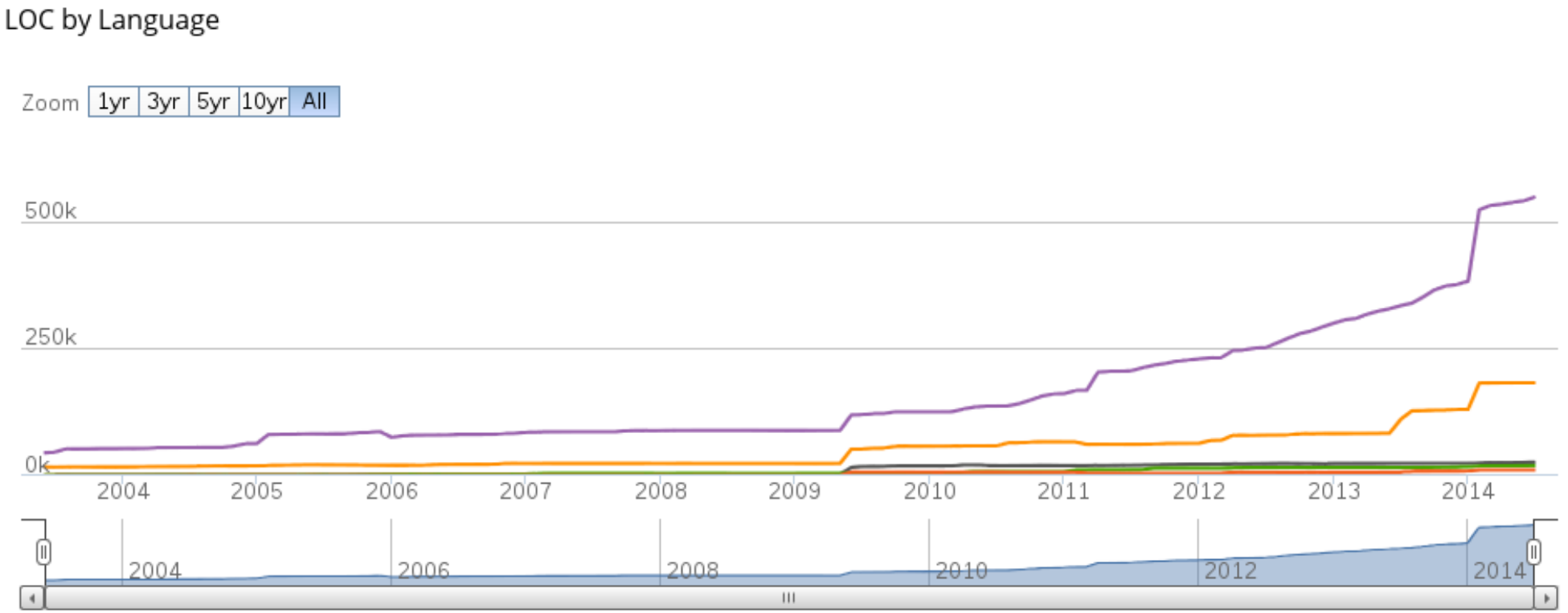
Continuous Progress

Total Code Growth (incl. comments)



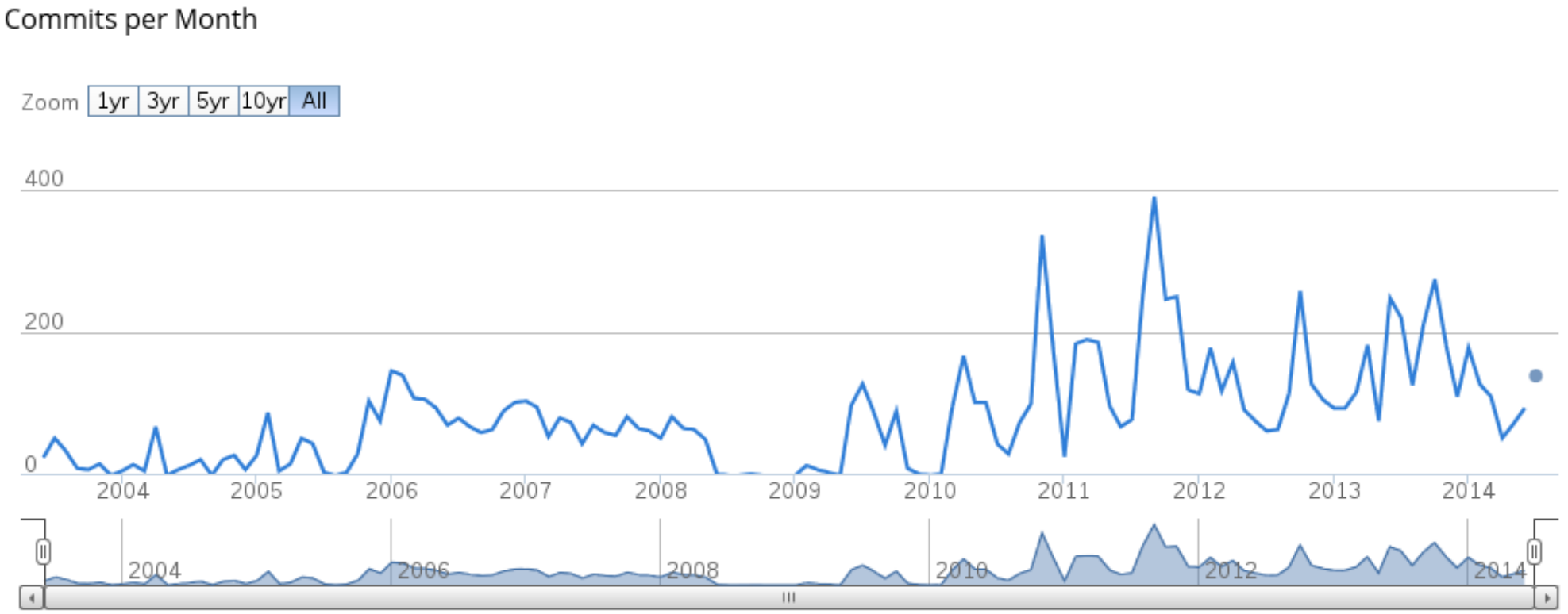
Continuous Progress

Total Code Growth (incl. comments)



Continuous Progress

Continuous Code Growth ...



Continuous Progress

- LOC net code growth (w/o comments)
- Appropriate growth per module …
- Test code gains weight toward desired ratio of 1:1

	v2.0-rc2	v2.0.2		v2.2.0	
nativewindow	4326	*+80%*	7797	*+29%*	10077
jogl	60982	*+63%*	99205	*+13%*	112107
newt	11916	*+91%*	22799	*+19%*	27317
Source total	**77224**	***+68%***	**129801**	***+15%***	**149501**
Test	10839	*+329%*	46544	*+44%*	67098
Test / Source	**14%**		**36%**		**45%**

Using *it!*

See Last Year's BOF Video / Slides

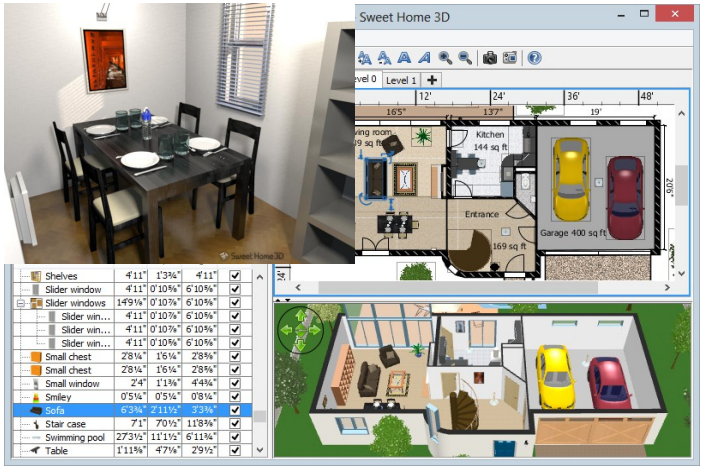
Java3D – I'm not Dead!

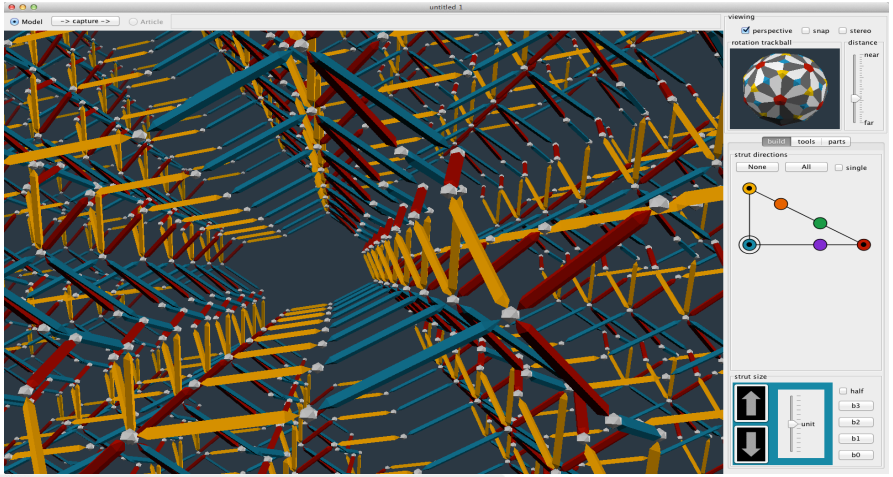
- A scenegraph library developed at Sun microsystems, eventually turned over to the community with a relicensing to GPLv2 w/ classpath exception.
- Picked up maintaining Java3D in 2012 from the unreleased 1.6.0-devel branch.
- Currently at version 1.6.0-pre11

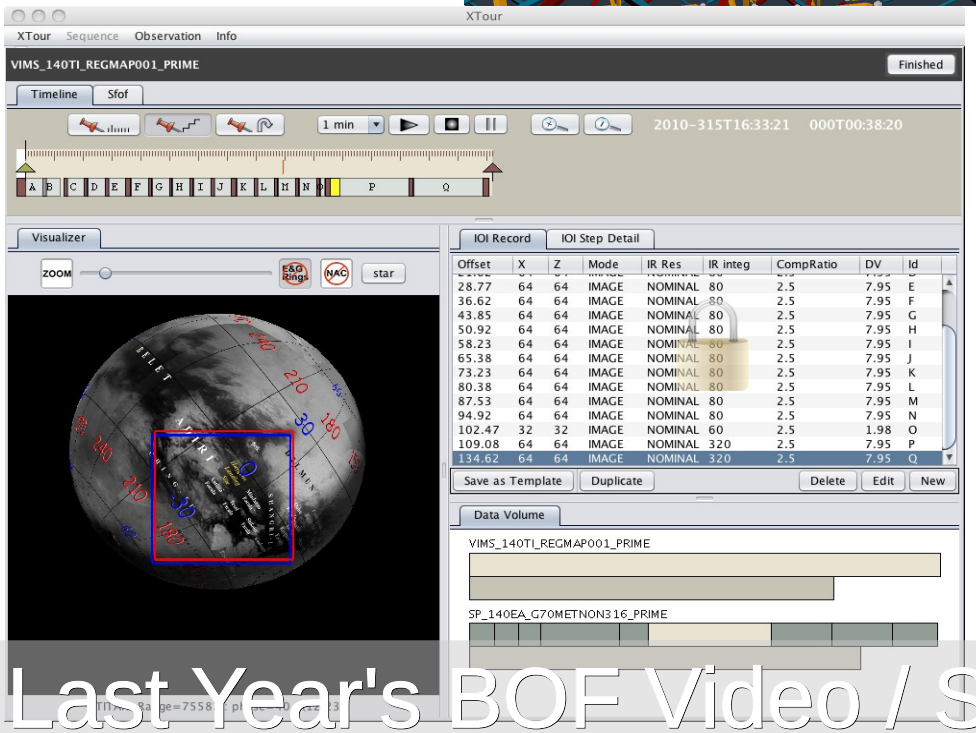
Java3D

- Removed all native backends, rely solely on the JOGL2 backend. This allows Java3D to continue to work on contemporary versions of OSX and newer Java (>= 7)
- Zero API changes, existing Java3D programs should work without any changes.
- Some multithreading bugs have been fixed.
- Two remaining regressions before declaring 1.6.0-final

Java3D Users







See Last Year's BOF Video / Slides

C3D

GlueGen

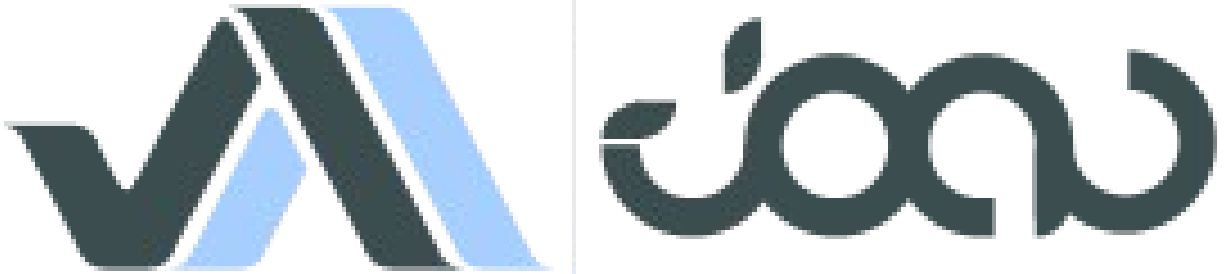
Often neglected, no logo, not mentioned.
However, GlueGen is core part of all JogAmp modules :)

- Compiler
  - Producing Java and Native Glue-Code from C Header files
    - C Functions
    - C Structs
- Runtime Tools
  - Native JAR locating and loading, supports deployment
  - Essential Glue-Code Utilities
  - Concurrency, I/O Helper, etc

Webstart Examples

JOCL Status

- Picked up maintenance
- Aligned *Build System* w/ GlueGen, JOGL, ..
- Added *Runtime Version* information like JOGL
- Added Android support
- In process of zero unit test failures
- Maps OpenCL 1.0 and 1.1
- No new features



Java™ binding for the openAL® api

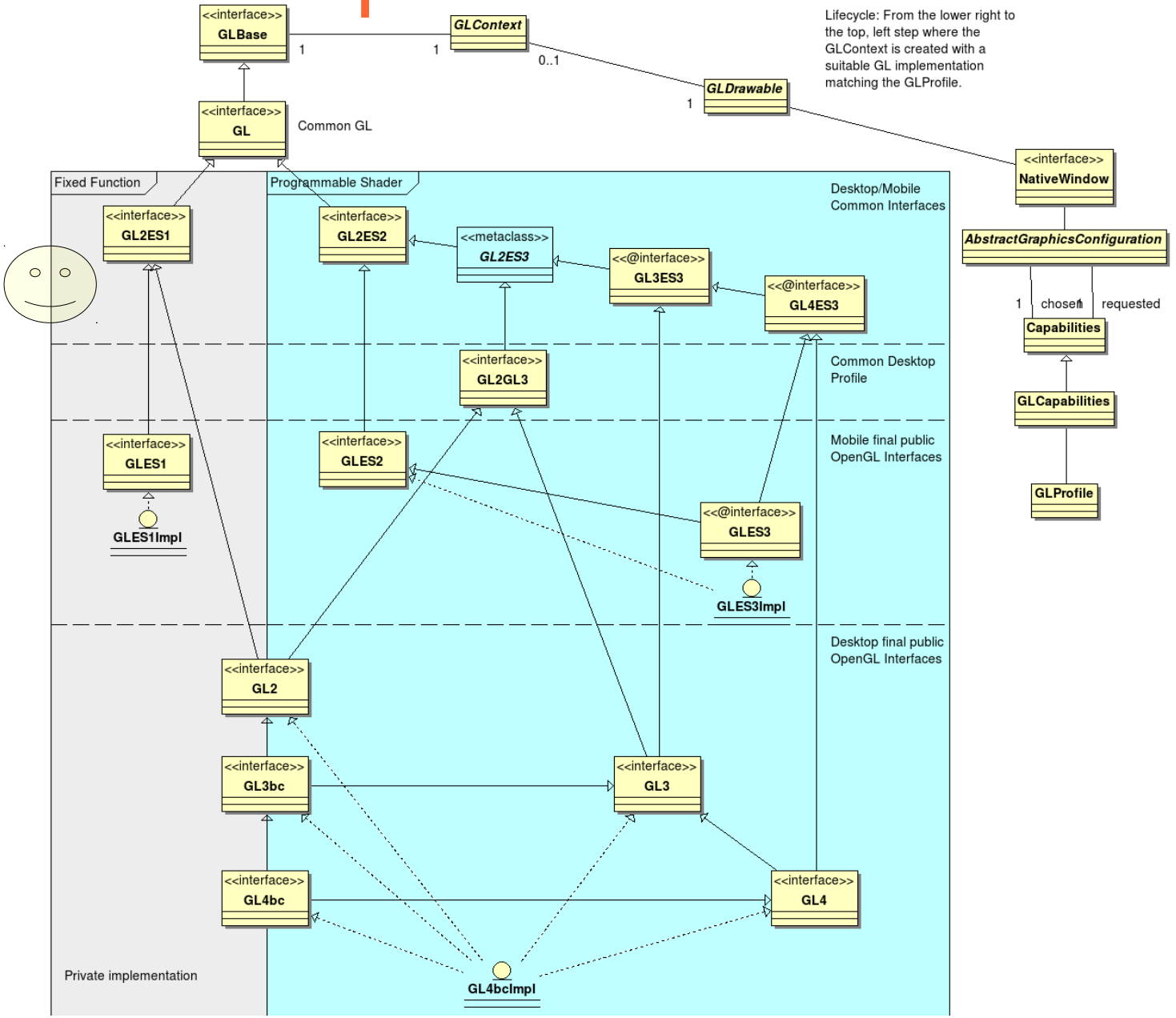
- JOAL is a Java binding of the OpenAL API
- OpenAL provides:
  - Spatial Sound
  - Low level audio buffer control / Streaming
  - Mixing of streams, incl. Doppler Effect
- Providing OpenAL-Soft on all platforms:

JOAL – More Information

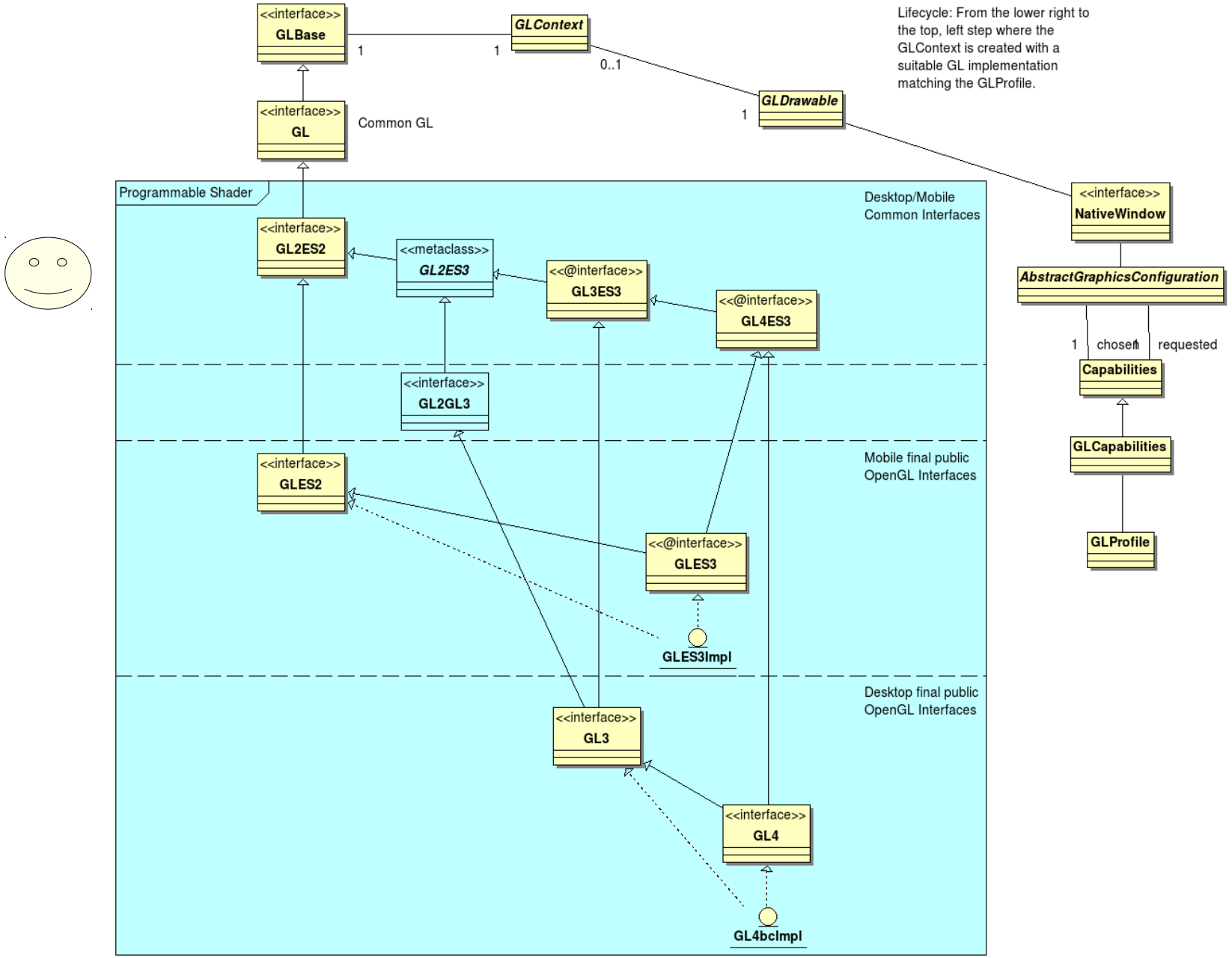
- Project Pages:
  - Demo code and tutorials https://jogamp.org/joal-demos/www/
  - Homepage: https://jogamp.org/joal


Java™ binding for the OpenGL® API

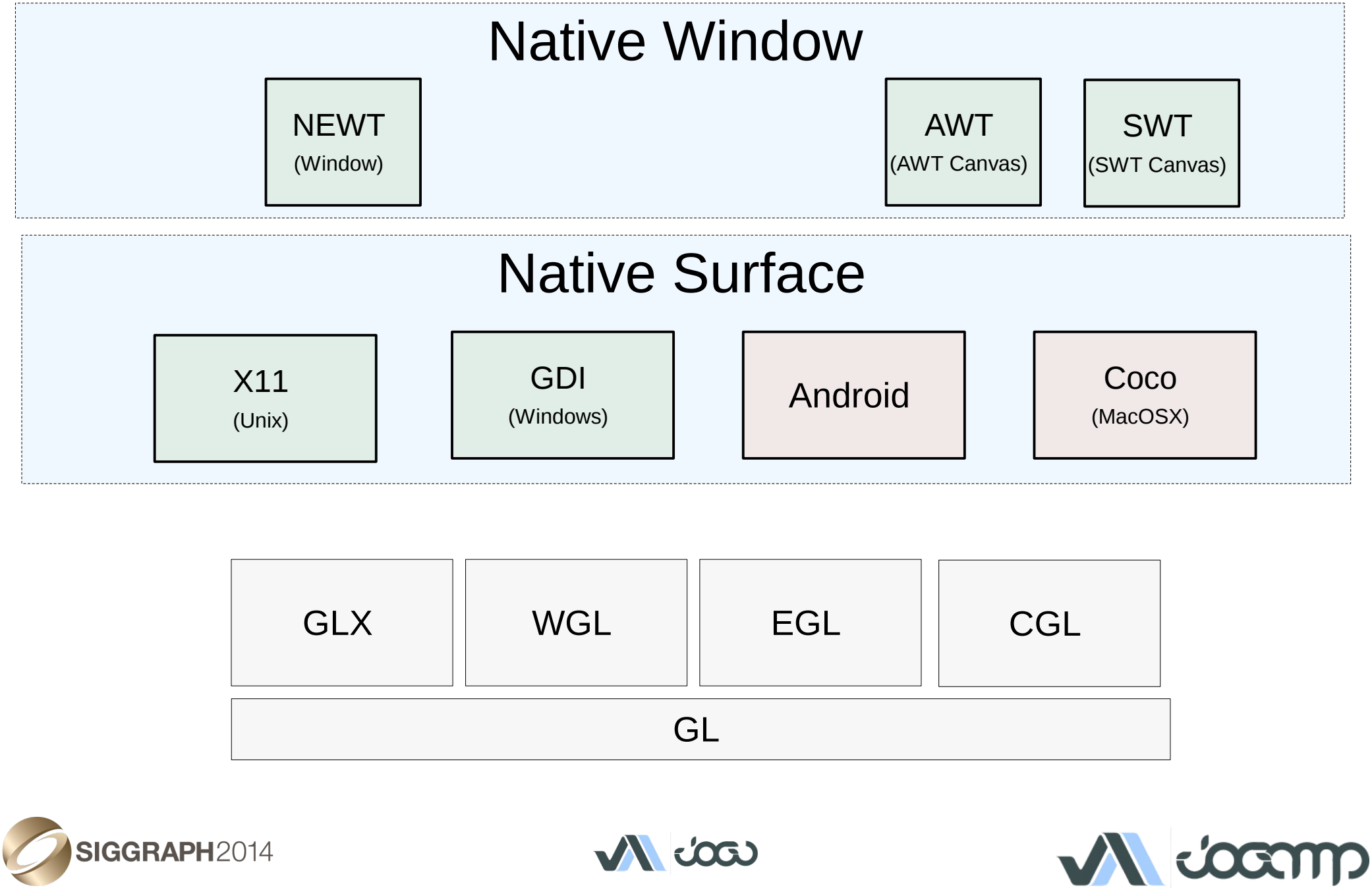
OpenGL Profiles



OpenGL Profiles



Windowing Toolkits



NEWT

- Cross Platform & Devices
- Multithreaded Surface Access
- Lock free event handling

Input Events

Keybd

Pointer

Output Events

Monitor

Windowing Features

- Create / Destroy
- Native Parenting
  - NEWT
  - AWT, SWT, …
- Fullscreen & Transparency
- Monitor
  - Multiple Devices
  - Mode Change

Backends

X11
(Unix)

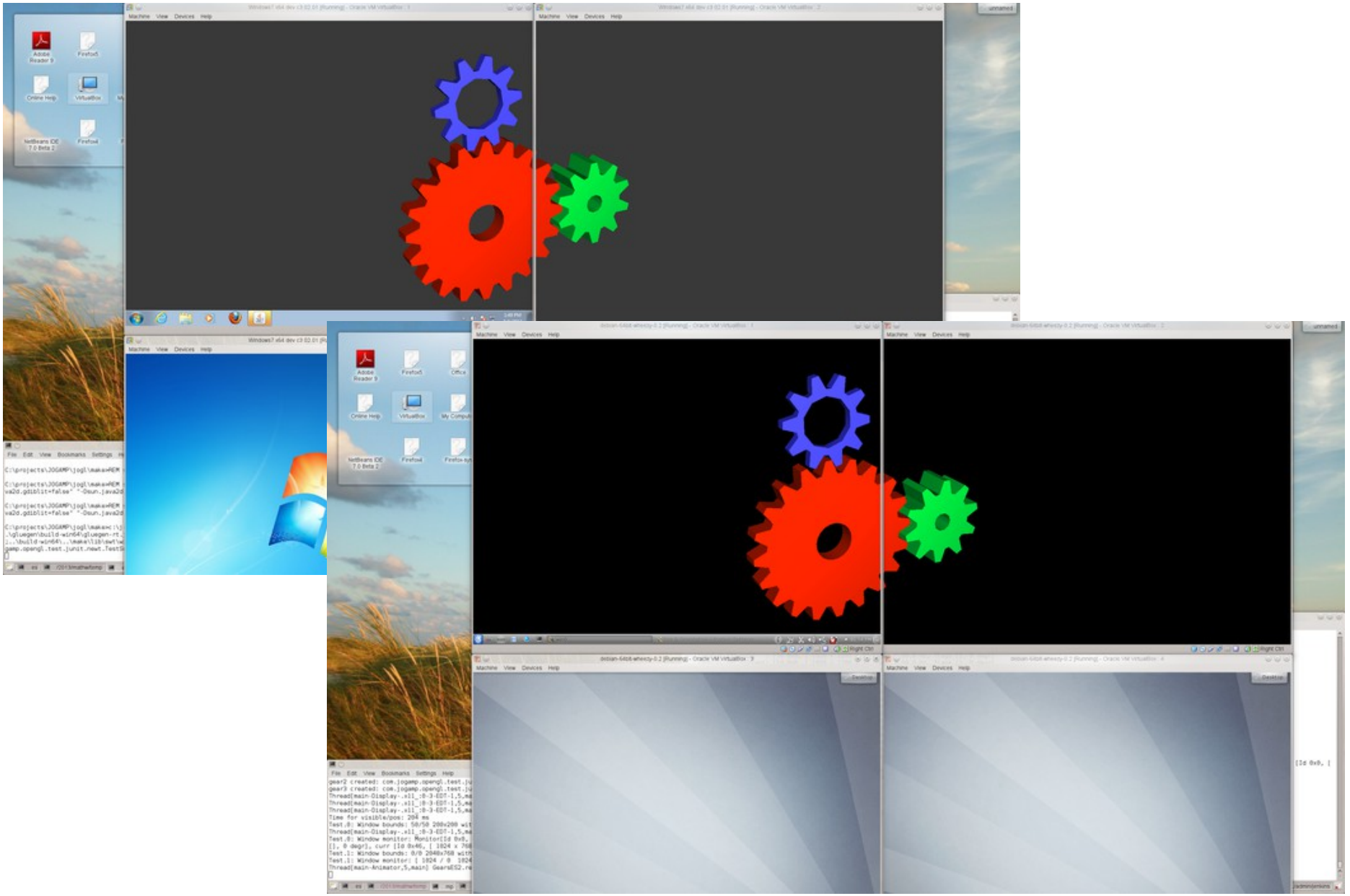
GDI
(Windows)

Android

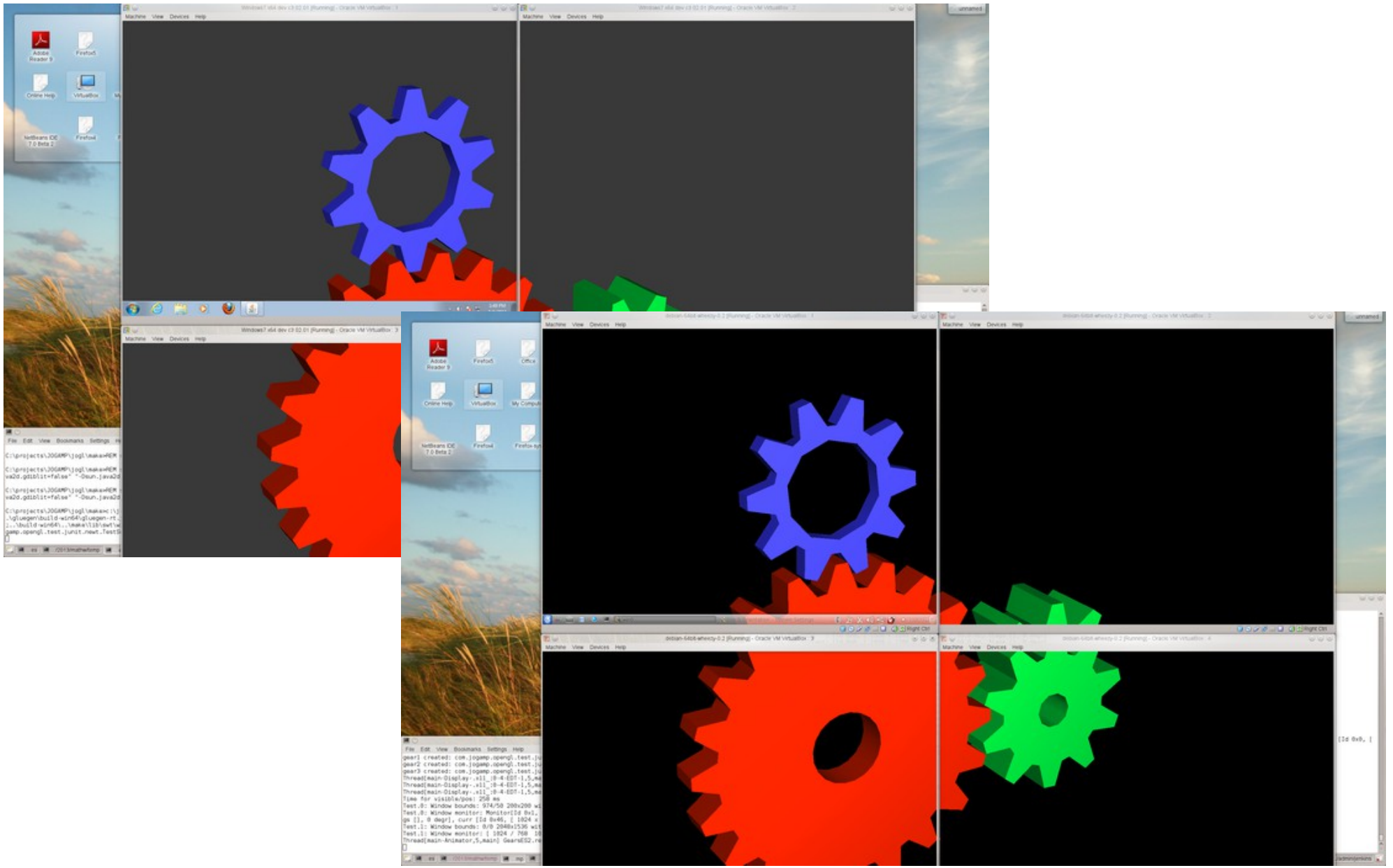
Coco
(MacOSX)

NEWT

NEWT



NEWT



GLAutoDrawable Context Sharing

```
public interface GLSharedContextSetter
          extends GLAutoDrawable
{
    void setSharedContext(GLContext sharedContext)
         throws IllegalStateException;

    void setSharedAutoDrawable(GLAutoDrawable
sharedAutoDrawable)
         throws IllegalStateException;
}
```

- Lifecycle Safe
- Guaranteed order of creation
- Used for
  - Off-thread texture loading, e.g. GLMediaPlayer
  - Isolating GLContext states

GLContext Reassociation

```
// switch context _and_ its
// GLEventListener synchronously

GLAutoDrawable glad1, glad2;

GLDrawableUtil.swapGLContextAndAllGL
EventListener(glad1, glad2);
```

GLContext Reassociation

Used by:

- GLContext Preservation
  - Implementing *GLStateKeeper*
  - *Used for temp. surface loss (e.g. Android)*
- Swapping On/Offscreen
  - Hi-DPI Print
  - Swapping OSX-CALayer / NEWT Window
- ...

GL Buffer Tracking

```
Class GLBufferStorage {
   public int getName();
   public long getSize();
   public ByteBuffer getMappedBuffer();
   ...
}
GLBufferStorage gl.getBufferStorage(int
bufferName);
int gl.getBoundBuffer(int target);
```

HiDPI Support

```
interface NativeSurface {
  // Returns the width of the
  // client area in pixel units.
  public int getSurfaceWidth();

  // Returns the height of the
  // client area in pixel units.
  public int getSurfaceHeight();
  ...
}
```

```
interface NativeWindow {
  // Returns the width of the
  // client area in window units.
  public int getWidth();

  // Returns the height of the
  // client area in window units.
  public int getHeight();
}
```

- Separated Window- and Surface space
- Immutable downstream API
- Mutable upstream via *ScalableSurface* implementation:

```
interface ScalableSurface {
    void setSurfaceScale(final int[] pixelScale);
}
```

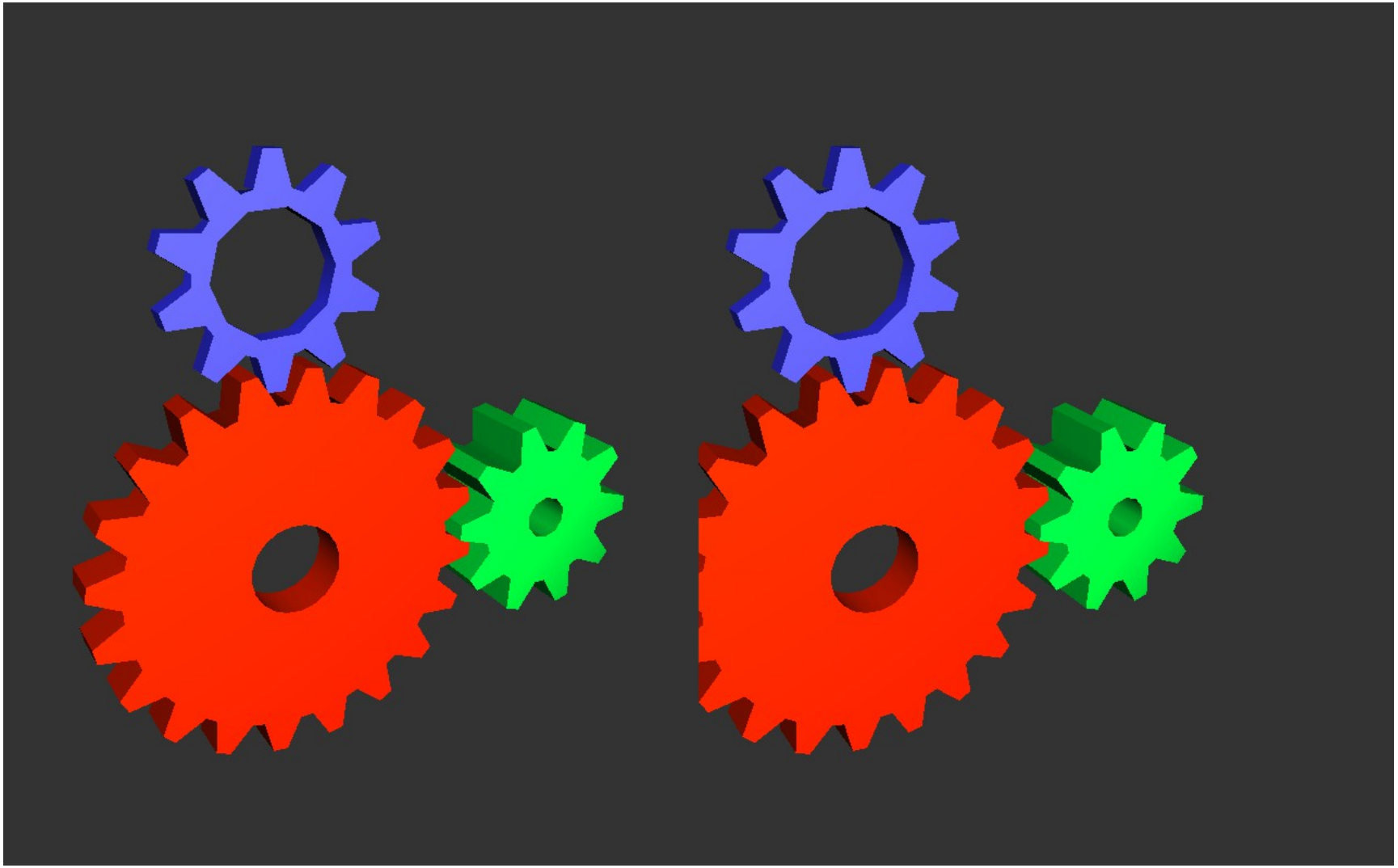
Stereoscopy

- *StereoDevice* provides
  - *StereoDeviceRenderer*
- *StereoClientRenderer*
  - Uses *StereoDeviceRenderer*
  - Correct asymmetric FOV Rendering, *off-axis*

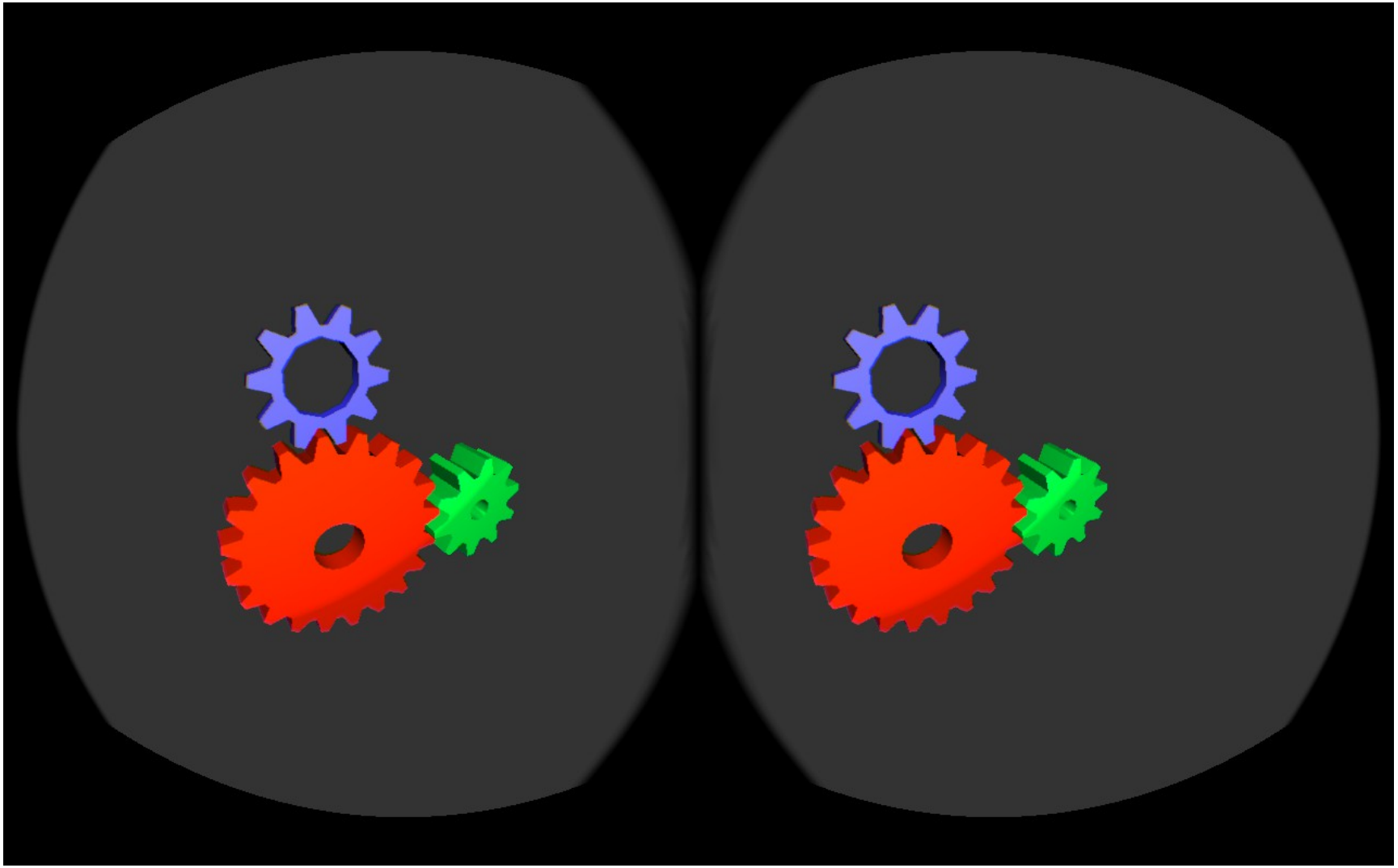
StereoDevice Implementations*:*

- *Soft Mono*
- *Soft Side-By-Side (SBS)*
- *Soft SBS w/ Lense Distortion*
- *OculusVR*

Stereoscopy



Stereoscopy

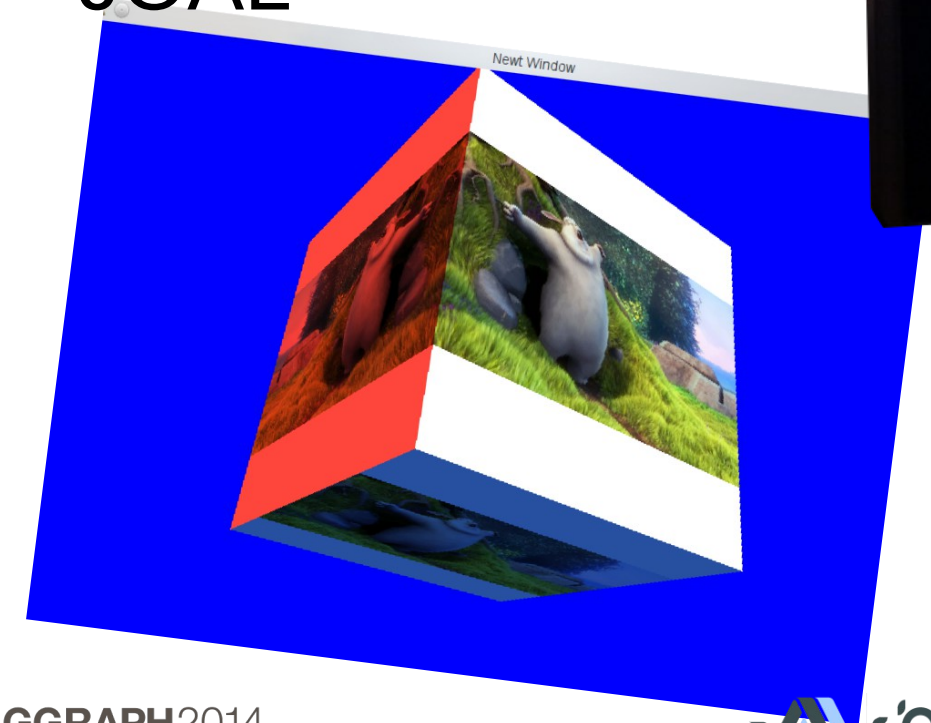


Stereoscopy

```
StereoGLEventListener demo = new GearsES2(0); GLWindow window = ..;

StereoDeviceFactory stereoDeviceFactory =

StereoDeviceFactory.createFactory(StereoDeviceFactory.DeviceType.Default);
StereoDevice stereoDevice =
          stereoDeviceFactory.createDevice(0, null, true /* verbose */);

FovHVHalves[] defaultEyeFov = stereoDevice.getDefaultFOV();
float[] eyePositionOffset = stereoDevice.getDefaultEyePositionOffset();
int textureUnit = 0;
int reqDistortionBits = stereoDevice.getRecommendedDistortionBits();
float pixelsPerDisplayPixel = 1f;
StereoDeviceRenderer stereoDeviceRenderer =
 stereoDevice.createRenderer(reqDistortionBits, 1, eyePositionOffset,
                             defaultEyeFov, pixelsPerDisplayPixel,
textureUnit);
int texFilter = GL.GL_LINEAR; // GL.GL_NEAREST;
StereoClientRenderer renderer = new StereoClientRenderer(
        stereoDeviceRenderer, true /* ownsDist */, texFilter, texFilter,
numSamples);

renderer.addGLEventListener(demo);
window.addGLEventListener(renderer);
window.setVisible(true);
```

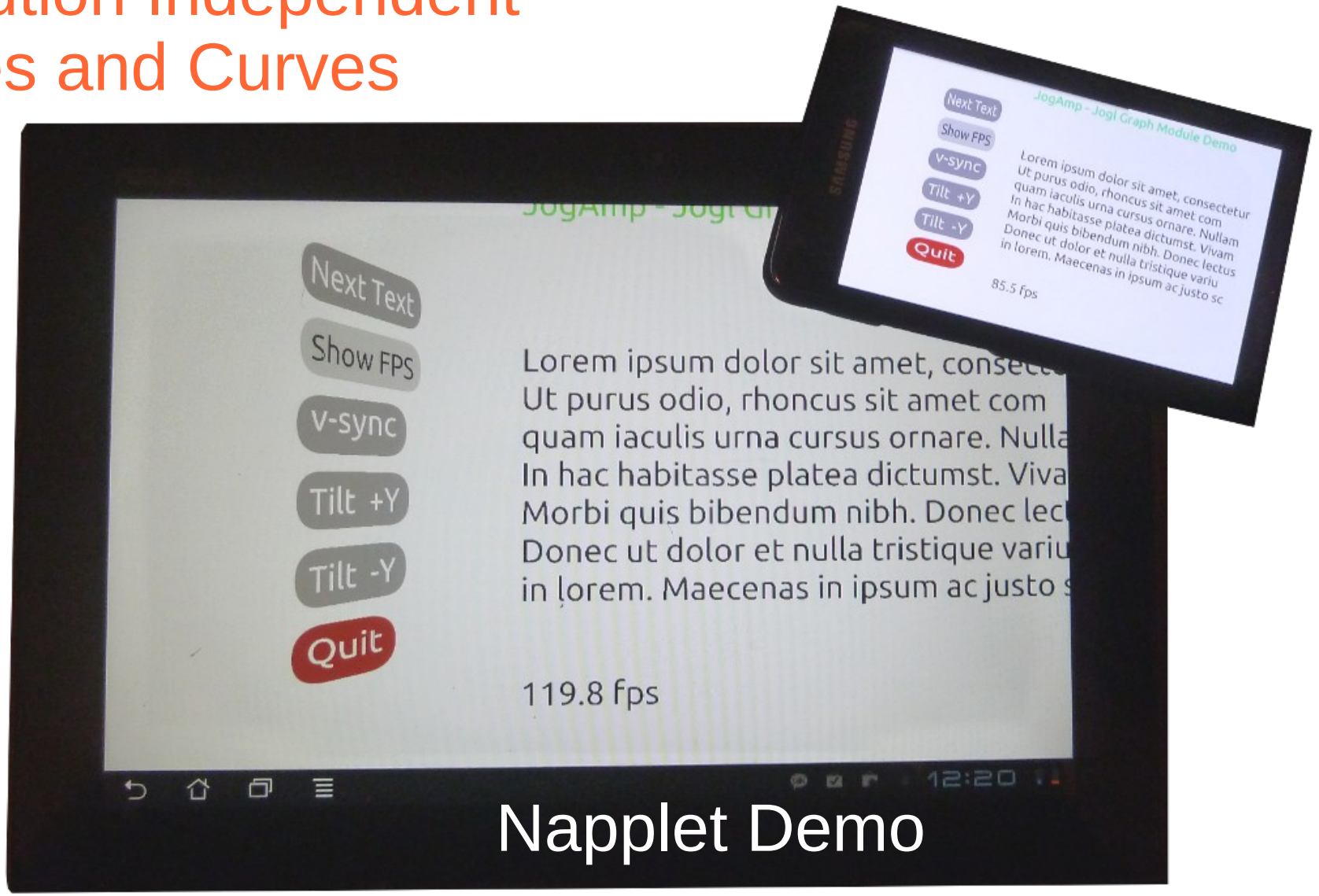
GLMediaPlayer

- Platform agnostic API
- Backends:
  - Android
  - FFMPEG / libav*
  - OpenMAX (wip)
  - JOAL



* Binds to system library, providing libav is WIP.

Graph API
Resolution Independent
Shapes and Curves



Graph

Resolution Independent Curve Rendering API

- Based on Paper:
  - R Santina, “Resolution Independent NURBS Curve Rendering using Programmable Graphics Pipeline”, presented in GraphiCon2011.
- **NOT** Loop/Blinn
- Patent Free
- Can Render Bezier, Bsplines, NURBS

Resolution Independent Curve Rendering API

- Why?
  - Resolution Independent Text Rendering
  - GPU based - Fast
  - Seamless integration into Renderer (Scenegraph,...)
  - New User Interface – across devices

- http://jogamp.org/deployment/jogamp-current/jogl-test-applets.html
- http://www.youtube.com/watch?v=Rqsu46ifMaw

Lorem ipsum dolor sit amet, consec
Ut purus odio, rhoncus sit amet con
quam iaculis urna cursus ornare. Nu
In hac habitasse platea dictumst. Vi
Morbi quis bibendum nibh. Donec l
Donec ut dolor et nulla tristique va
in lorem. Maecenas in ipsum ac just

Graph – Backend

Graph GPU based curve rendering backend

- *Graph Enhancements*
  - Caches processed OutlineShapes
    to be placed arbitrary using 2D transforms
  - Allowing diverse OutlineShape renderer
    - Single- or multicolor
    - Texture & TextureSequence
      - GLMediaPlayer
  - Anti-Aliasing modes:
    - VBAA: Brute force (1, 2, .. 8), Flip-Quad
    - MSAA
  - Automatic shader selection for above renderer modes, incl. anti-aliasing 2nd pass.

Graph

- *Graph* TTF Text Rendering
  - Caching Glyph OutlineShape, high performance
- *Graph UI*
  - WIP
  - Selection by intersection
  - Propagate mouse coordinates in graph coordinates
  - Utilizing different renderer (colors, textures, ..)
  - Implementing:
    - GLEventListener Button using shared GLContext/FBO
    - GLMediaPlayer Button using shared GLContext/FBO
    - ..

JogAmp Deployment

- Preinstalled Bundles
  - Modularized JARs
  - Android APKs (modular, or all-in-one)
  - Maven / Gradle
- Online / Cached
  - Automatic Native-JAR loading support
  - Applet
    - Classical
    - JNLP
  - Webstart (JNLP)

Maven

- The JogAmp project currently distributes Jar files and .7z archives containing compiled code, source code, and documentation
- June 2012 - Stable versions and release candidates are released to the Central Repository, and bleeding edge packages are published to a testing repository at http://jogamp.org/deployment/maven

Maven

- Traditionally JogAmp locates native JAR files derived from it's java JAR URL without utilizing the classpath for performance reasons.
- As of 2.2.0, the JogAmp attempts to loading native JAR files referenced from the classpath as a fallback. This allows users to place them in arbitrary URL locations.
- As a result, JogAmp Maven packages can now be used from Gradle!

Maven

- As of 2.2.0, all (previously incomplete) packages have been updated and all atomics are now deployed to Maven Central.
- Extra atomic packages have been added to make it easier to pull in atomics for all platforms. For example, depending on **nativewindow-main** will create a dependency on the **nativewindow** atomics for all platforms.

Maven

- As of 2.2.0, all packages are subjected to validation tests before being published to Maven Central.
- The 2.1.5 release had a mistake that broke the packages (corrected in 2.1.5-01). The new unit tests prevent this kind of error from occurring.

Thank You & Love You

Rami Santina

Mr. Max

Jens Hohmuth

Sven Gothel

Xerxes Ranby

Julien Gouesse

Emmanuel Puybaret

Andres Colubri

Harvey Harrison

Mark Raynsford

Wade Walker

… all the many contributors & users