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:

<table>
<thead>
<tr>
<th>Feature</th>
<th>Module</th>
<th>Native API</th>
</tr>
</thead>
<tbody>
<tr>
<td>Audio</td>
<td>JOAL</td>
<td>OpenAL</td>
</tr>
<tr>
<td>Graphics</td>
<td>JOGL</td>
<td>OpenGL</td>
</tr>
<tr>
<td>Multimedia</td>
<td>JOGL</td>
<td>FFMpeg / libAV, ...</td>
</tr>
<tr>
<td>Compute / Processing</td>
<td>JOCL</td>
<td>OpenCL</td>
</tr>
</tbody>
</table>

What is *it*?
.. 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.

What is it?
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
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

Doing it..
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

Doing *it*..
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

<table>
<thead>
<tr>
<th>Build</th>
<th>Duration</th>
<th>Slave</th>
</tr>
</thead>
<tbody>
<tr>
<td>#1329</td>
<td>1 hr 43 min</td>
<td>master</td>
</tr>
<tr>
<td>#1328</td>
<td>1 hr 43 min</td>
<td>master</td>
</tr>
<tr>
<td>#1327</td>
<td>1 hr 47 min</td>
<td>master</td>
</tr>
<tr>
<td>#1325</td>
<td>1 hr 44 min</td>
<td>master</td>
</tr>
<tr>
<td>#1322</td>
<td>1 hr 43 min</td>
<td>master</td>
</tr>
<tr>
<td>#1320</td>
<td>1 hr 43 min</td>
<td>master</td>
</tr>
<tr>
<td>#1317</td>
<td>1 hr 42 min</td>
<td>master</td>
</tr>
<tr>
<td>#1316</td>
<td>2 hr 40 min</td>
<td>master</td>
</tr>
<tr>
<td>#1315</td>
<td>1 hr 44 min</td>
<td>master</td>
</tr>
<tr>
<td>#1314</td>
<td>1 hr 42 min</td>
<td>master</td>
</tr>
<tr>
<td>#1309</td>
<td>1 hr 42 min</td>
<td>master</td>
</tr>
<tr>
<td>#1308</td>
<td>1 hr 42 min</td>
<td>master</td>
</tr>
<tr>
<td>#1307</td>
<td>1 hr 43 min</td>
<td>master</td>
</tr>
<tr>
<td>#1305</td>
<td>1 hr 42 min</td>
<td>master</td>
</tr>
<tr>
<td>#1304</td>
<td>1 hr 42 min</td>
<td>master</td>
</tr>
</tbody>
</table>

Doing it..
## Continuous Progress
### Overall Build Monitoring

### Test Result

<table>
<thead>
<tr>
<th>Configuration Name</th>
<th>Duration</th>
<th>All</th>
<th>Failed</th>
<th>Skipped</th>
</tr>
</thead>
<tbody>
<tr>
<td>label=android-armv7-limg</td>
<td>11 ms</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>label=macosx-10_6-x86_64-nvidia</td>
<td>32 min</td>
<td>975</td>
<td>6</td>
<td>0</td>
</tr>
<tr>
<td>label=win7-x86_64-nvidia</td>
<td>26 min</td>
<td>986</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>label=win7-x86_64-amd</td>
<td>25 min</td>
<td>986</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>label=solaris-x86_64-nv</td>
<td>28 min</td>
<td>957</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>label=linux-armv7hf-limg</td>
<td>11 ms</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>label=linux-x86_64-nvidia</td>
<td>25 min</td>
<td>986</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>label=solaris-x86_32-nv</td>
<td>29 min</td>
<td>986</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>label=linux-armv7-limg</td>
<td>11 ms</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>label=linux-x86_32-amd</td>
<td>11 ms</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>label=macosx-10_7-x86_64-nvidla_jre7</td>
<td>46 min</td>
<td>979</td>
<td>5</td>
<td>0</td>
</tr>
<tr>
<td>label=linux-x86_32-nvidia</td>
<td>26 min</td>
<td>986</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>label=win7-x86_32-amd</td>
<td>29 min</td>
<td>986</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>label=win7-x86_32-nvidia</td>
<td>23 min</td>
<td>986</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>label=linux-x86_64-amd</td>
<td>25 min</td>
<td>985</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Continuous Progress
Unit Test Monitoring

<table>
<thead>
<tr>
<th>Package</th>
<th>Duration</th>
<th>Fail</th>
<th>(diff)</th>
<th>Skip</th>
<th>(diff)</th>
<th>Pass</th>
<th>(diff)</th>
<th>Total</th>
<th>(diff)</th>
</tr>
</thead>
<tbody>
<tr>
<td>com.jogamp.opengl.test.junit.graph</td>
<td>25 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>10</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.jogamp.opengl.test.junit.jgcl.acore</td>
<td>5 min 16 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>274</td>
<td>274</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.jogamp.opengl.test.junit.jgcl.acore.anim</td>
<td>18 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>10</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.jogamp.opengl.test.junit.jgcl.acore.ect</td>
<td>59 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>40</td>
<td>40</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.jogamp.opengl.test.junit.jgcl.acore.gles</td>
<td>1 min 47 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>47</td>
<td>47</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.jogamp.opengl.test.junit.jgcl.awt</td>
<td>2 min 22 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>65</td>
<td>65</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.jogamp.opengl.test.junit.jgcl.awt.text</td>
<td>1.4 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.jogamp.opengl.test.junit.jgcl.caps</td>
<td>24 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>29</td>
<td>29</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.jogamp.opengl.test.junit.jgcl.demos.es1.newt</td>
<td>6.7 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.jogamp.opengl.test.junit.jgcl.demos.es2.awt</td>
<td>16 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>19</td>
<td>19</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.jogamp.opengl.test.junit.jgcl.demos.es2.newt</td>
<td>26 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>14</td>
<td>14</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.jogamp.opengl.test.junit.jgcl.demos.g2.awt</td>
<td>6.2 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>9</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.jogamp.opengl.test.junit.jgcl.demos.g2.newt</td>
<td>8.2 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.jogamp.opengl.test.junit.jgcl.demos.g3.newt</td>
<td>6.5 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.jogamp.opengl.test.junit.jgcl.gis</td>
<td>13 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>15</td>
<td>15</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.jogamp.opengl.test.junit.jgcl.giu</td>
<td>4.3 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.jogamp.opengl.test.junit.jgcl.math</td>
<td>3.2 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>104</td>
<td>104</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.jogamp.opengl.test.junit.jgcl.newt</td>
<td>11 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.jogamp.opengl.test.junit.jgcl.offscreen</td>
<td>16 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>9</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.jogamp.opengl.test.junit.jgcl.perf</td>
<td>1 min 42 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>21</td>
<td>21</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.jogamp.opengl.test.junit.jgcl.tlb</td>
<td>2 min 33 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>46</td>
<td>46</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.jogamp.opengl.test.junit.jgcl.util</td>
<td>27 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>15</td>
<td>15</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.jogamp.opengl.test.junit.jgcl.util.texture</td>
<td>2 min 11 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>89</td>
<td>89</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.jogamp.opengl.test.junit.newt</td>
<td>47 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>34</td>
<td>34</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.jogamp.opengl.test.junit.newt.event</td>
<td>3 min 15 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>38</td>
<td>38</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.jogamp.opengl.test.junit.newt.mm</td>
<td>2 min 32 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>21</td>
<td>21</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>com.jogamp.opengl.test.junit.newt.parenting</td>
<td>1 min 28 sec</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>30</td>
<td>30</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Continuous Progress

No performance regressions...

Doing it..
Continuous Progress

No performance regressions ...

Doing it..
Continuous Progress

Total Code Growth (incl. comments)

Code, Comments and Blank Lines

Zoom 1yr 3yr 5yr 10yr All

1,500k

1,000k

500k

0k


SIGGRAPH 2014

Doing it..
Continuous Progress

Total Code Growth (incl. comments)

LOC by Language

Zoom 1yr 3yr 5yr 10yr All

0k 250k 500k


SIGGRAPH 2014

Doing it..
Continuous Progress

Continuous Code Growth ...

Commits per Month

Zoom 1yr 3yr 5yr 10yr All

Doing it..
Continuous Progress

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

<table>
<thead>
<tr>
<th>Module</th>
<th>v2.0-rc2</th>
<th>v2.0.2</th>
<th>v2.2.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>nativewindow</td>
<td>4326</td>
<td>+80%</td>
<td>7797</td>
</tr>
<tr>
<td>jogl</td>
<td>60982</td>
<td>+63%</td>
<td>99205</td>
</tr>
<tr>
<td>newt</td>
<td>11916</td>
<td>+91%</td>
<td>22799</td>
</tr>
<tr>
<td>Source total</td>
<td>77224</td>
<td>+68%</td>
<td>129801</td>
</tr>
<tr>
<td>Test</td>
<td>10839</td>
<td>+329%</td>
<td>46544</td>
</tr>
<tr>
<td>Test / Source</td>
<td>14%</td>
<td></td>
<td>36%</td>
</tr>
</tbody>
</table>

Doing it..
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
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
opencal

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

Lifecycle: From the lower right to the top, left step where the GLContext is created with a suitable GL implementation matching the GLProfile.
OpenGL Profiles

Lifecycle: From the lower right to the top, left step where the GLContext is created with a suitable GL implementation matching the GLProfile.

Programmable Shader

Desktop/Mobile
Common Interfaces

AbstractGraphicsConfiguration
Capabilities
GLCapabilities
GLProfile

GLProfile

GLContext

GLBase

Common GL

GL

GLDrawable

GL2ES2

GL2ES3

GL3ES3

GL4ES3

GL2GL3

GLES2

GLES3

GLES3Impl

GL3

GL4

GL4dImpl

desktop final public OpenGL Interfaces

mobile final public OpenGL Interfaces

NativeWindow

1 chosen requested
Windowing Toolkits

Native Window

- NEWT (Window)
- AWT (AWT Canvas)
- SWT (SWT Canvas)

Native Surface

- X11 (Unix)
- GDI (Windows)
- Android
- Coco (MacOSX)

- GLX
- WGL
- EGL
- CGL

- GL
NEWT

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

**Backends**
- X11 (Unix)
- GDI (Windows)
- Android
- Coco (MacOSX)

**Input Events**
- Keybd
- Pointer

**Output Events**
- Monitor

**Windowing Features**
- Create / Destroy
- Native Parenting
  - NEWT
  - AWT, SWT, ...
- Fullscreen & Transparency
- Monitor
  - Multiple Devices
  - Mode Change

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

Napplet Demo

Graph
Resolution Independent Curve Rendering API

- Based on Paper:
- **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, consectetur adipiscing elit. aliquam iaculis urna cursus ornare. Nullam

In hac habitasse platea dictumst. Vivamus vulputate lorem.

Mauris quis bibendum nibh. Donec lacinia, neque a

Donec ut dolor et nulla tristique varius. in lorem. Maecenas in ipsum ac justo

Graph
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
Julien Gouesse
Andres Colubri

Jens Hohmuth
Xerxes Ranby
Harvey Harrison
Wade Walker

Sven Gothel
Emmanuel Puybaret
Mark Raynsford

... all the many contributors & users