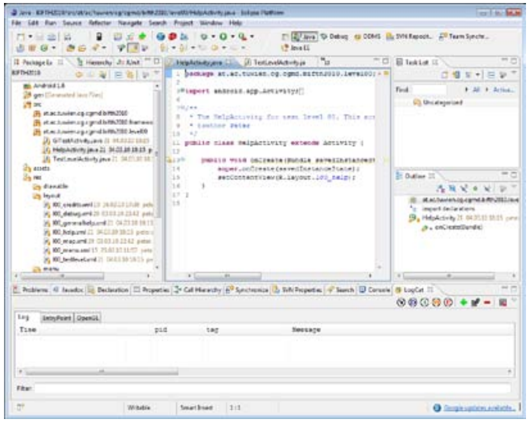
Augmented Reality on Mobile Devices

VU SS2011 4.5 ECTS

Michael Gervautz, Peter Rautek

Last Time

- Motivation
- Lab
  - VU Framework
  - Organizational
- Android Basics
  - Activity, Layouts, Service, Intent, Permission, etc.
  - Development with Eclipse
- Any Questions???



```
package at.ac.tuwien.cg.cgmd.bifth2010.level00;

import android.app.Activity;

/**
 * The HelpActivity for team level 00. This ac
 * @author Peter
 */
public class HelpActivity extends Activity {

    public void onCreate(Bundle savedInstanceSt
        super.onCreate(savedInstanceState);
        setContentView(R.layout.l00_help);
    }
}
```

Overview

- Advanced Android Topics
  - Rendering
  - Debugging OpenGL
  - Configuration, Resources and Localization
- OpenGL ES
  - History
  - Overview
- Lab Phase II

Rendering on Android

- Canvas (mainly for 2D)
- WebKit (for html)
- OpenGL ES the Java way
  - Version 1.x ever since
  - Version 2.0 since Android 2.2
- OpenGL the C(++) way
  - Version 1.x since Android 1.6
  - Version 2.0 since Android 2.0
- Renderscript
  - Since Android 3.0

OpenGL „The Java Way“

- Activity holds GLSurfaceView
  - Implements the lifecycle methods
    - onPause
    - onResume
- Extend the GLSurfaceView
  - Call the setRenderer method and pass a GLSurfaceView.Renderer
- Implement the GLSurfaceView.Renderer
  - onSurfaceCreated
  - onSurfaceChanged
  - onDrawFrame

GLSurfaceView

- Manages memory (surface), composited into the view system
- Manages OpenGL rendering to the surface
- Requires a implementation of the GLSurfaceView.Renderer interface
- Rendering runs in own thread
- On-demand vs. continuous rendering
- OpenGL debugging
- Default: 16-bit R5G6B5, 16-bit depth buffer

Renderer

- Runs in separate thread
- onSurfaceCreated
  - Called when
    - The activity is started
    - The OpenGL context was destroyed and recreated
  - Load Textures
- onSurfaceChanged
  - Called when size/orientation changes
- onDrawFrame

OpenGL „The Java Way“

- Pro
  - Very easy to implement
- Con
  - Lower performance
  - Garbage collection can lead to hickups

- Check out the API demos!

OpenGL “The C/C++ Way”

- Native development kit (NDK)
  - Native implementation
  - Generate make files
  - Build shared library
  - Use SDK tools to build application
    - Load shared library from Java

      -
        ```
        static {
                        System.loadLibrary(„mylibrary“);
        }
        ```
    - Declare native methods
      - private static native void nativeMyFunction();
    - Call native functions from GLSurfaceView
- Two demo apps in the NDK
  - san-angeles: OpenGL 1.x
  - hello-gl2: OpenGL 2.x
    - Android 2.0 and higher
    - Not running in emulator

OpenGL “The C/C++ Way”

- Pro
  - Performance
- Con
  - Harder to implement
  - Cumbersome development
  - Debugging

- Check out the NDK demos!

OpenGL “The ARMD Way”

- QCAR SDK runs native code
  - Tracking
  - Pose estimation
  - Rendering of camera output
  - OpenGL drawing could be done here
- ARMD framework on top
  - Redirects OpenGL drawing to Java
  - Exposes information about tracking targets
- Inherit from class AugmentedRealityRenderer
  - See example level00

Debugging OpenGL

- Context Wrapper (see AboutActivity, OpenGL 1.x)

```
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    mGLSurfaceView.setGLWrapper(new GLWrapper(){
          private Writer logger = new Writer(){
          //implementing the writer interface
          };

          @Override
          public GL wrap(GL gl) {
              return GLDebugHelper.wrap(
                    gl,
                    GLDebugHelper.CONFIG_CHECK_GL_ERROR|GLDebugHelper.CONFIG_LOG_ARGUMENT_NAMES,
                    logger);
          }
    });
}
```

Performance

- Always know your frame rate!
- Log (+easy, -floods your Log, -bad visibility)
- Onscreen
  - OpenGL Overlay
    - Write text to bitmap
    - Render as texture
  - GUI Overlay
    - Use FrameLayout
    - Overlay TextView

Graphical User Interface Overlays

- Multithreading
  - User interface (Activity)
  - Rendering (GLSurfaceView.Renderer)
  - Your own threads
- Synchronization
  - Necessary to avoid concurrency problems
  - Handler class provides message queue
  - Example: Display frame rate

GUI Overlay for OpenGL Activity

- Layout
  - Set/Get FrameLayout
  - Add GLSurfaceView (and renderer)
  - Add TextView
- Timer
  - Schedule a task for every second or so
  - Get the frame count from the renderer
  - Send a message to the handler
- Handler
  - Update UI (TextView)

Configurations

- Configurations depend on
  - Device
    - Screen size
    - Keyboard
  - User preference
    - Language
  - Situation
    - Orientation
- Solution
  - Multiple resources in one apk

Resources and Configurations

- Change of configuration
  - Destroying activity
  - Restarting activity with new configuration
- Resource folders
  - Loading of resource in appropriate folder
  - Depending on new configuration

Example: Different Languages

- Localization
  - res/values/string.xml
    The default must contain all strings
  - res/values-de/string.xml
    The de folder may contain localized strings for German version
  - res/values-fr/string.xml
    The fr folder may contain localized strings for French version

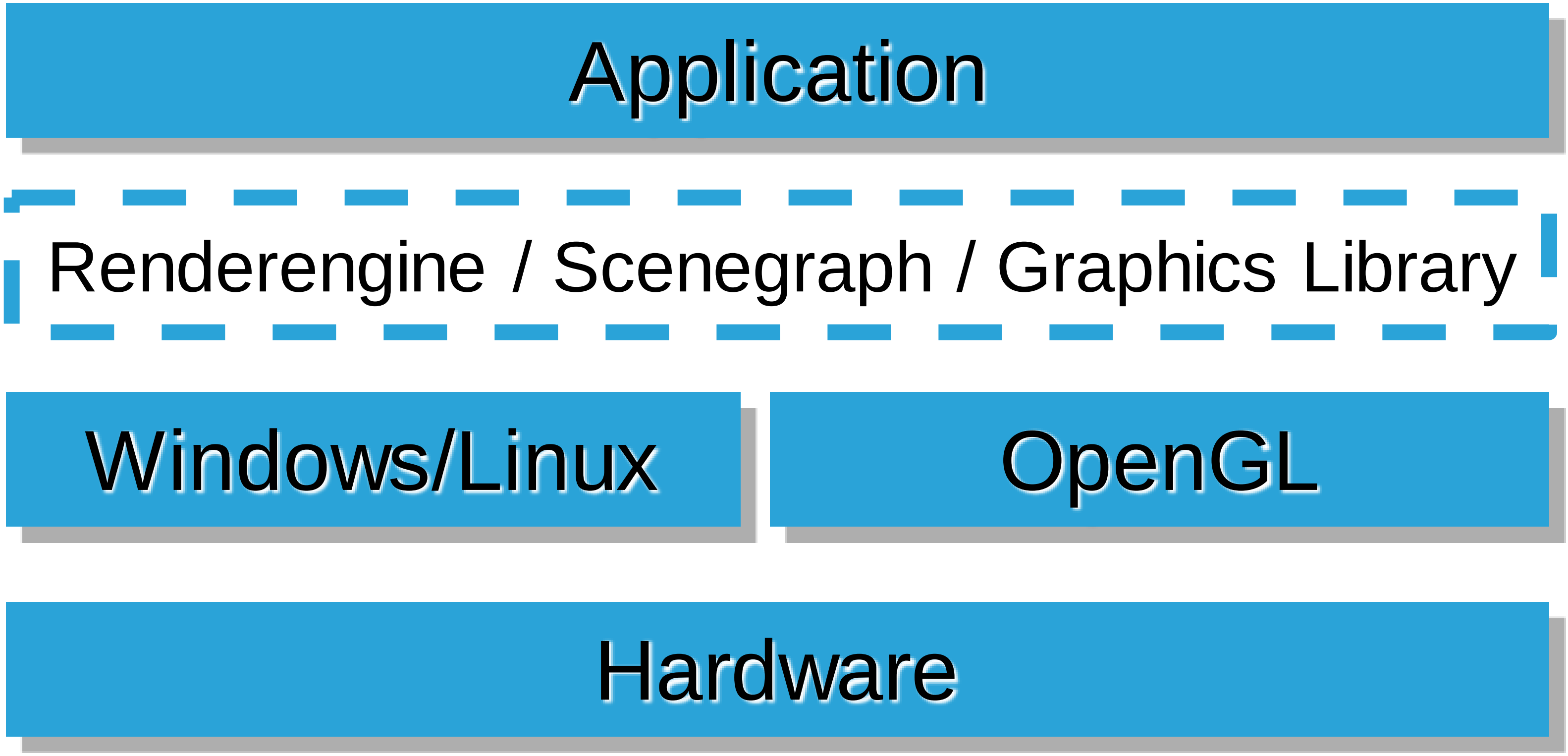
Example: Switching Orientation

- Handle orientation change
  - Automatic
    - res/layout-port/main.xml
    - res/layout-land/main.xml
  - Fixed orientation (no change occurs)
    - Specifiy in AndroidManifest file
  - Custom behavior
    - Specify in AndroidManifest file
    - Implement onConfigurationChanged()

Overview

- Advanced Android Topics
  - Rendering
  - Debugging OpenGL
  - Configuration, Resources and Localization
- OpenGL ES
  - History
  - Overview
- Lab Phase II

OpenGL – Who, What, and Most Importantly Why?


Application

Renderengine / Scenegraph / Graphics Library

Windows/Linux

OpenGL

Hardware

OpenGL Specification

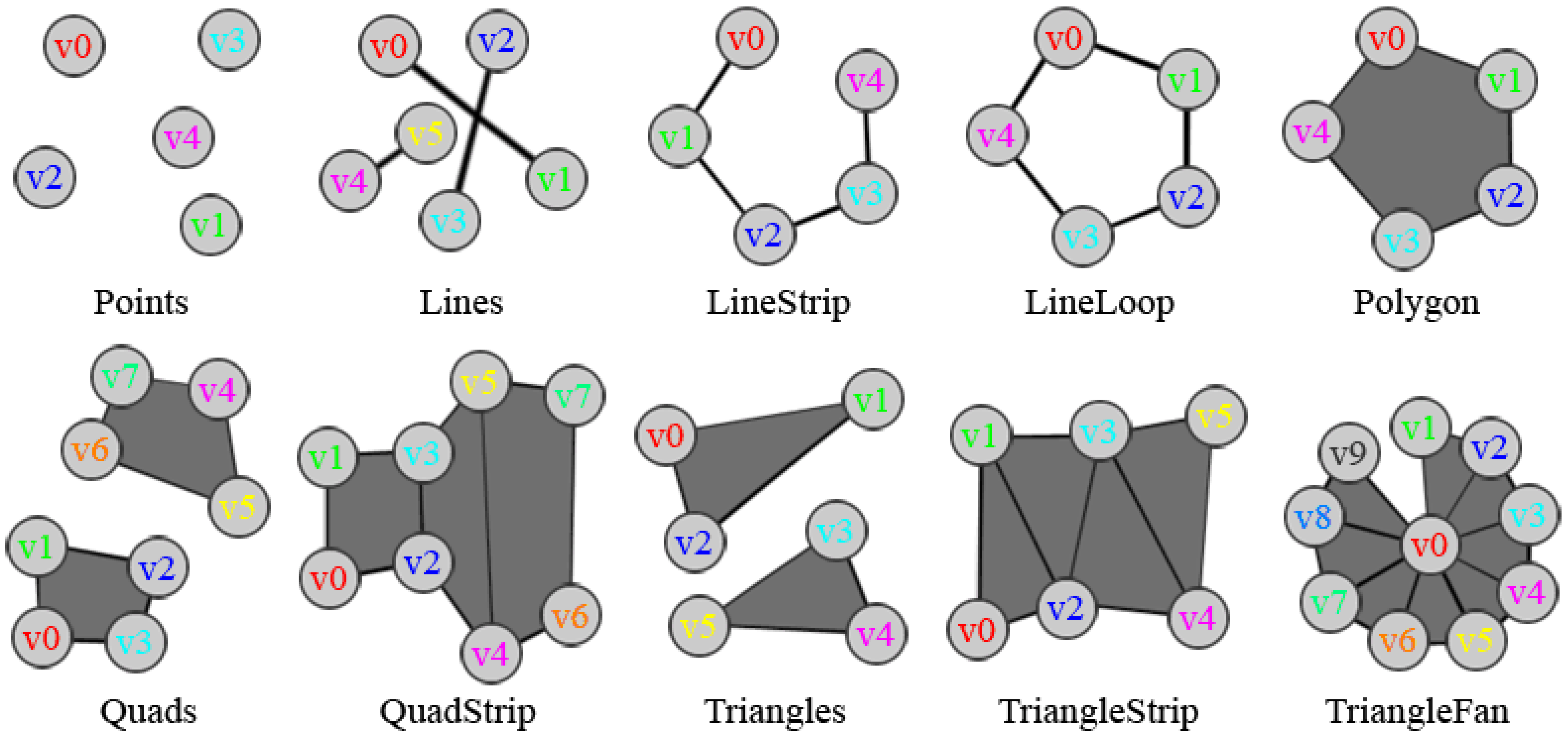
- Since 2006 under control of Khronos Group
  - Non profit consortium
  - Open standards
  - Royalty free
- Working Groups
  - OpenGL, OpenGL ES, OpenCL, COLLADA, OpenVG, OpenSL ES, EGL, WebGL, etc.
- Members
  - AMD, Apple Inc., ARM Holdings, Creative Labs, id Software, Ericsson, Intel Corporation, Motorola, Nokia, Nvidia, Samsung Electronics, Sony Computer Entertainment, Sun Microsystems, Texas Instruments, etc.
- Links: www.khronos.org, www.opengl.org

OpenGL Design Goals

- Platform independent
- Language independent
- Consistency
  - Conformance tests and required verification
  - Not pixel exact, but invariant across passes
- Complete implementations
  - Missing features emulated in software
- Clean interface
  - State machine
  - Most states are orthogonal
- Extensibility
  - Favors innovation

OpenGL in a Nutshell

- Small number of primitives
- Defined by vertices



Points

Lines

LineStrip

LineLoop

Polygon

Quads

QuadStrip

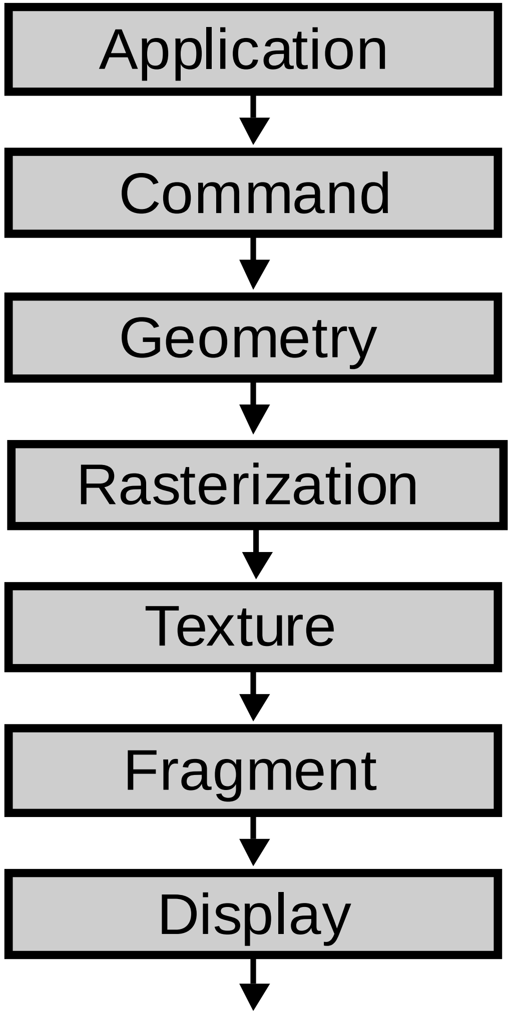
Triangles

TriangleStrip

TriangleFan

Graphics Pipeline

- Application
  - Scene traversal
  - Object, and camera movement
  - Animation
  - Occlusion/Visibility Culling
  - Level of Detail (LoD) selection
- Geometry
  - Transformation (model, world, view)
  - View Projection
  - Culling (e.g., back-face)
  - Perspective Division
  - Clipping
  - Screen space transformation
- Triangle Setup
- Rasterization
- Texturing
- Fragment
  - Shading
  - Depth Buffering



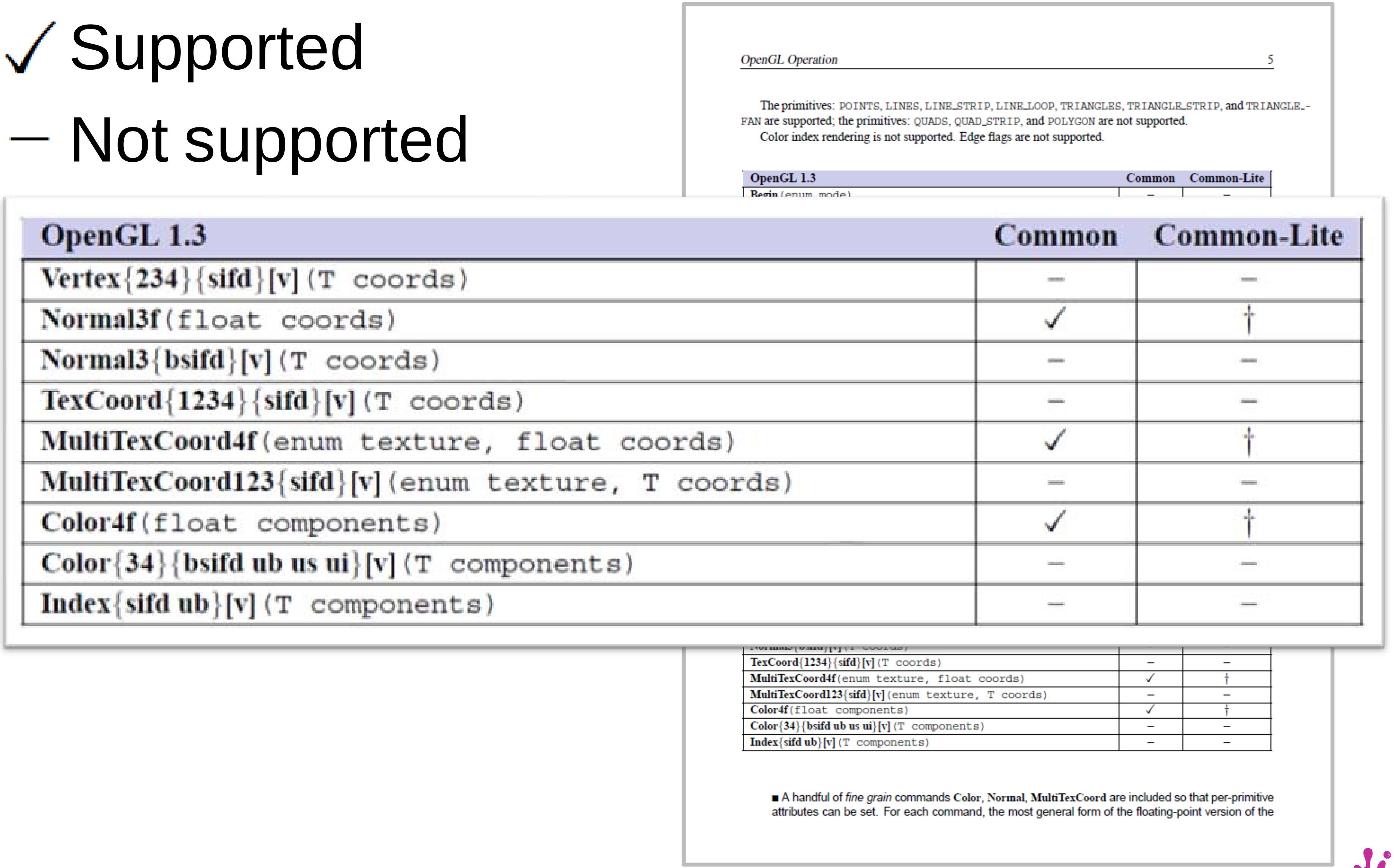
OpenGL vs. OpenGL ES

- Specification
  - OpenGL ES 1.0 written against OpenGL 1.3
  - OpenGL ES 1.1 written against OpenGL 1.5
  - OpenGL ES 2.0 written against OpenGL 2.0
- Differences
  - Single vs. Double
  - Fixed vs. Floating
  - No glBegin(), glEnd(), glVertex()
  - No display lists
  - etc.

Example

✓ Supported

– Not supported



OpenGL 1.3	Common	Common-Lite
Vertex{234}{sifd}[v] (T coords)	–	–
Normal3f(float coords)	✓	†
Normal3{bsifd}[v] (T coords)	–	–
TexCoord{1234}{sifd}[v] (T coords)	–	–
MultiTexCoord4f(enum texture, float coords)	✓	†
MultiTexCoord123{sifd}[v] (enum texture, T coords)	–	–
Color4f(float components)	✓	†
Color{34}{bsifd ub us ui}[v] (T components)	–	–
Index{sifd ub}[v] (T components)	–	–

OpenGL Operation 5

The primitives: POINTS, LINES, LINE_STRIP, LINE_LOOP, TRIANGLES, TRIANGLE_STRIP, and TRIANGLE_FAN are supported; the primitives: QUADS, QUAD_STRIP, and POLYGON are not supported.

Color index rendering is not supported. Edge flags are not supported.

OpenGL 1.3	Common	Common-Lite
Begin(enum mode)	–	–

TexCoord{1234}{sifd}[v] (T coords)	–	–
MultiTexCoord4f(enum texture, float coords)	✓	†
MultiTexCoord123{sifd}[v] (enum texture, T coords)	–	–
Color4f(float components)	✓	†
Color{34}{bsifd ub us ui}[v] (T components)	–	–
Index{sifd ub}[v] (T components)	–	–

■ A handful of *fine grain* commands **Color**, **Normal**, **MultiTexCoord** are included so that per-primitive attributes can be set. For each command, the most general form of the floating-point version of the

Short History of OpenGL (ES)

1991	OpenGL ARB created
1992	OpenGL 1.0 (June 30)
1995	OpenGL 1.1
1996	OpenGL specification made public
1998	OpenGL 1.2
2000	OpenGL goes open source
2001	OpenGL 1.3
2001	OpenGL ES 1.0
2002	OpenGL 1.4
2003	OpenGL 1.5
2003	OpenGL ES 1.1
2004	OpenGL 2.0 (shaders)
2004	OpenGL ES 2.0
2008	OpenGL 3.0
2010	OpenGL 4.0

OpenGL ES 1.0 vs. 2.0

- Main novelty: shading language GLSL
- Vertex and fragment shaders
  - Replace fixed functionality
- Shader: high-level language (C-like)
- OpenGL driver: compiler and linker for shaders
- Vertex-, texture coordinates etc.: abstract input values to shader function
- Arbitrary calculations possible

Code Examples

- Android API Demos:
  com.example.android.apis.graphics
  - OpenGL 1.0: CubeRenderer
  - OpenGL 2.0: GLES20TriangleRenderer
- ARMD Framework:
  - OpenGL 1.0: AboutActivity
  - OpenGL 2.0: Level00Renderer

Overview

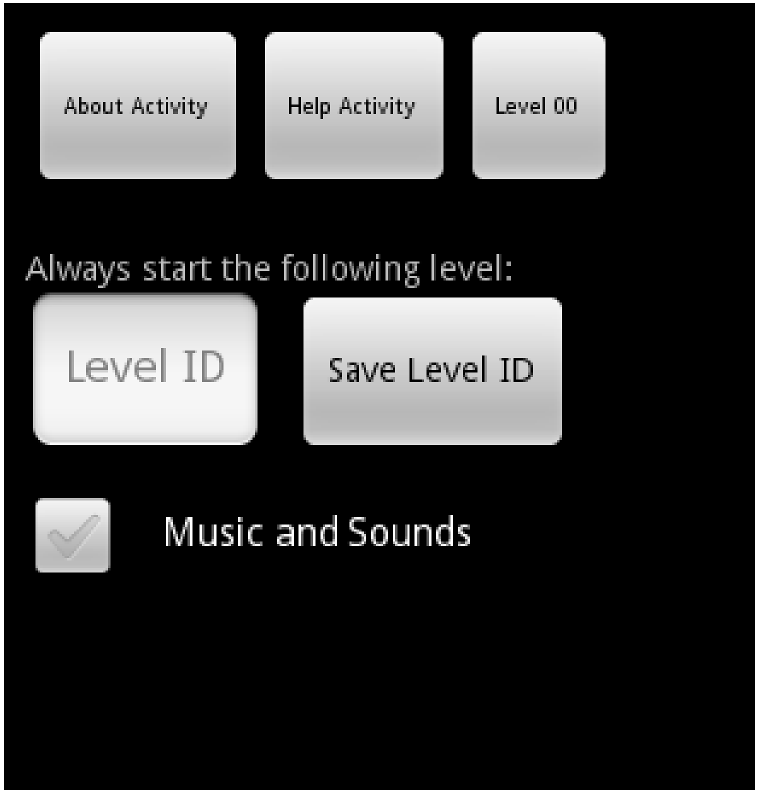
- Advanced Android Topics
  - Rendering
  - Debugging OpenGL
  - Configuration, Resources and Localization
- OpenGL ES
  - History
  - Overview
- Lab Phase II

Phase II

- Localization
  - Language: English (default) and German
- Framework Integration
  - Sound shall be configurable (on and off)
  - Replace icon template: IXX_icon
  - Value between 0-100
  - Documentation
- Implement your level
  - Functions, game play
  - Detailed instructions online

Framework - LauncherActivity

- About Activity:
  - OpenGL ES 1.0
  - Debug Wraper
- Help Activity:
  - Help of all levels
  - Icons
- Level00:
  - The AR cubes
- Save Level ID:
  - For quick debugging
  - Preferences (saving state)
- Sound on/off



Always start the following level:

☑ Music and Sounds

SVN Policy

- Look at code of others (and learn), but don’t modify it!
- Commit only if your code compiles and resources (drawables, etc.) are available
- If you break the build
  - You will be publicly punished ;)
  - The Lab-Admis might modify your code
- Write Log Messages!

Organizational

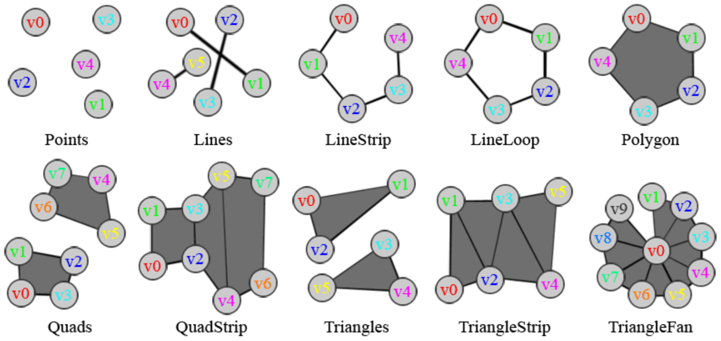
- Start now – start making troubles early!
- FAQ Event: 28.03. 17:15
- Deadline (last commit): 01.05. 23:59
- Zwischengespraech: 02.05.
- Hardware:
  - Each group one mobile (???)
  - Starting 15.03. 11:00 @secretary
  - First come first serve
  - Daily ~ 9:00 – 12:00 (details on homepage)

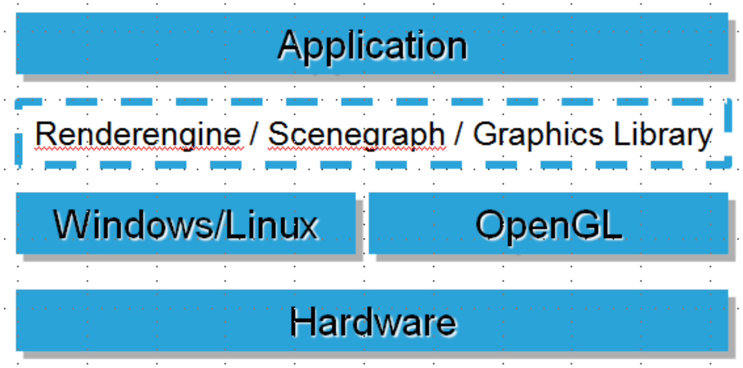
Where to Start - Links

- Open GL ES Specifications:
  - http://www.khronos.org/opengles/spec/
- Android
  - GLSurfaceView
    http://developer.android.com/reference/android/opengl/GLSurfaceView.html
  - Resources and Internationalization
    http://developer.android.com/guide/topics/resources/resources-i18n.html
  - API Demos
    http://developer.android.com/resources/samples/ApiDemos/src/com/example/android/apis/graphics/index.html
- Forum of ARMD
  - http://www.informatik-forum.at/forumdisplay.php?1378-Augmented-Reality-on-Mobile-Devices-VU

Danke für Ihre Aufmerksamkeit

Fragen?



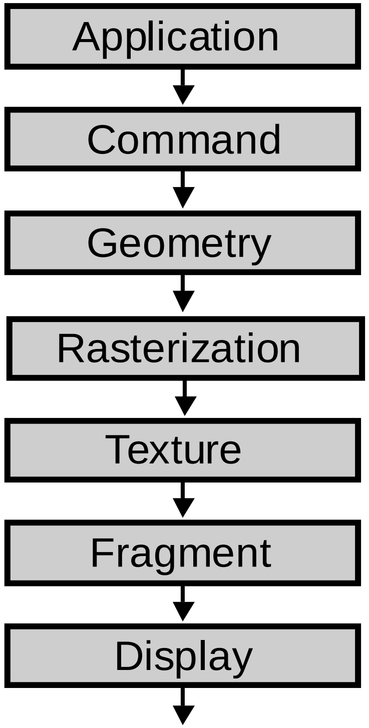

Application

Renderengine / Scenegraph / Graphics Library

Windows/Linux

OpenGL

Hardware


Application

Command

Geometry

Rasterization

Texture

Fragment

Display