Advanced Modern OpenGL: Pipeline and Performance

Based on OpenGL 4.5, created on June 13, 2018

KDAB

The Qt, OpenGL and C++ Experts
Advanced Modern OpenGL with Qt

- What is OpenGL?
- Terminology
- Qt helper classes

Performance, Profiling and Debugging

- Debugging OpenGL
- Synchronization Mechanisms
- Measuring
- Performance
  - Culling
  - Optimizing state changes
- Advanced Context Usage
- Optimising Memory Transfers
- Threading
- Buffering Schemes

The OpenGL Memory Model

- Introduction
- Memory coherence
- Memory barriers
- Atomics

Advanced Buffer Usage

- Instanced Rendering
- Uniform Buffer Objects
- Multi-draw Indirect
- Shader Storage Buffer Objects
- Shaders and Shader Programs introspection
- Shader Subroutines
- Image Load/Store

- Geometry Shaders
  - OpenGL Pipeline Revisited
  - Geometry Shader Basics
  - Shader Interface Blocks
  - Removing Geometry
  - Modifying Geometry
    - Augmenting Geometry Data
    - Changing the Primitive Type
- Tessellation
  - Introduction to Tessellation
  - Tessellation in Detail
  - Bezier Curves and Patches
  - Mesh Refinement

- Transform Feedback
  - Capturing Vertex Data
  - Setting Up Buffers
  - Using Transform Feedback
  - Feedback Rendering
  - Transform Feedback Object
- Compute Shaders

- Recommended Books