

---

# Programming with Qt 3D

Based on Qt 5.15, created on March 16, 2023

The logo for KDAB, featuring a stylized white lightning bolt icon to the left of the text "KDAB" in a white, sans-serif font, all contained within a blue rectangular box with a white diagonal cutout at the bottom left corner.

**KDAB**

*The Qt, OpenGL and C++ Experts*

---

- Feature Set
- Entity Component System? What's that?
- Hello Donut
- Qt 3D ECS Explained

No part of this publication may be made available to others than the named licensee which is shown on every page by any means, electronic, mechanical, photocopying, recording or otherwise, or translated into any language, without the prior written permission of the publisher.

- Introduction
- Geometries and Materials
- Lights

- Picking
- Basic Input Handling
- Logical Devices
- Accumulators

## Integrations and Helpers

- Dynamic QML Scenes
  - The NodeInstantiator Element
  - The EntityLoader Element
  - The SceneLoader Element
- Starting the Qt 3D Engine
- Painted Textures
- Integrating Qt Quick with Qt 3D

## The Qt 3D Scene Graph

- The Simplified Graphics Pipeline
- Shader Programs
  - Introduction to Shader Programs
  - Configurable shader programs
- Geometry and Coordinate Systems
  - How to Specify Geometries?
  - Coordinate Systems
- Texturing
  - Texturing Basics
  - Texturing Geometry
  - Texture Sampling
  - Using Multiple Textures

## The Qt 3D Frame Graph

- Viewports and Layers
- Transparency
- Selecting Shaders at Runtime
- Image-Based Techniques
  - Rendering to a Texture
  - Post-Processing Effects
    - Edge Detection
    - Television Effect
    - Gaussian Blur
    - Effect Chains

## Animation with Qt 3D

- Key Frame Animations
- Combining Animations

## Qt 3D with Qt 6

- API Changes
- Renderer Changes

## Optional Topics

- Instanced Rendering
- Procedural Texturing
- Capturing the Rendering
- Level of Detail
- Displaying Text