KUESA

Getting started with KUESA™ 3D

KDAB

1. Introduction

KUESA™ 3D consists of two parts, KUESA™ 3D Runtime and KUESA™ 3D Studio.

KUESA™ 3D works with both Qt3D or Serenity. KUESA™ 3D works with Qt3D out of the box, however, Serenity is a Vulkan renderer you can use with KUESA™ 3D or standalone if you just need a Vulkan renderer.

Serenity allows the use of KUESA™ 3D with one of the other supported 2D frameworks such as Rmlui, Imgui or any other 2D framework (as long as it supports Vulkan).

1.1 KUESA™ 3D Runtime

This is the developer facing part of the Kuesa workflow.

KUESA™ 3D Runtime is a software library that provides support for loading and displaying glTF2 3D assets. It is to be used either with the Qt 3D renderer library of with a KUESA™ 3D's own Vulkan based renderer (also referred to as Serenity). In addition to plain glTF2 file handling, it has support for additional custom glTF2 extensions that can provide enhancements (Layer tagging for filtering scenes, optimized Iro materials for embedded targets, draco compression to reduce asset size, ...).

1.2 KUESA™ 3D Studio

This is the designer facing part of the Kuesa workflow.

KUESA™ 3D Studio is a set of accompanying tools that help with the export, optimization and previsualization of glTF2 3D assets.

It provides glTF2 exporters for the major 3D Design softwares (Blender, Maya) that handle the custom KUESA™ 3D glTF2 extensions. When it comes to previsualization, the glTFInspector tools allow designer to preview their glTF2 scenes as see them as they would be rendered by the KUESA™ 3D Runtime. This allows to check that content matches without the need of integrating the content in a 3D application.

2. Prerequisites

Please make sure to remove all former KUESA™ 3D and Serenity installations from your system. Otherwise Cmake may find it which may result in configure or build errors:

- System wide Kuesa and Serenity installations
- Kuesa installations inside any Qt versions

See Section 8 Uninstalling KUESA™ 3D for more details.

2.1 Hardware

2.1.1 Desktop

Anything with OpenGL 3.2 support (or more recent) is adequate.

Currently supported operating systems are:

- Windows
- Linux
- Mac OS

2.1.2 Embedded

KUESA™ 3D Runtime supports anything with OpenGL ES 3.2 or above. OpenGL ES 3.1 or lower is supported but with a limited feature set depending on available OpenGL extensions:

- For HDR support, GL_OES_texture_half_float is required
- For anti aliasing, ARB_texture_multisample and GL_EXT_color_buffer_half_float are required

A list of Chipsets/Devices KUESA™ has successfully been tested on:

- Apple iPad 5 (PowerVR GT7600)
- Apple iPhone 7
- OnePlus 3T (Android, Qualcolmm Snapdragon 821/ Adreno 430)
- NVidia Tegra K1 (embedded Linux)
- iMX8 (embedded Linux)
- Raspi
- Android

2.2 Software

2.2.1 Windows

Required software for KUESA™ :

- Microsoft Visual Studio C++ 2019 compiler (or later)
- git - https://gitforwindows.org
- CMake - https://cmake.org/, make sure it is in your PATH

For the KUESA™ 3D Studio Renderer (Serenity) you'll need:

- Vulkan SDK - https://www.lunarg.com/vulkan-sdk/
    - Make sure to install the additional debug libraries when prompted during the install process
    - Make sure VULKAN_SDK is set accordingly in your environment
- vcpkg - https://vcpkg.io/en/getting-started.html
    - Ensure that the path to the install directory is in your PATH

2.2.2 Linux

Required software for KUESA™ 3D:

- C++17 compiler (GCC / Clang)
- git
- CMake

For the KUESA™ 3D Studio Renderer (Serenity) you'll need:

- Vulkan SDK - https://www.lunarg.com/vulkan-sdk/
    - Make sure VULKAN_SDK is set accordingly in your environment
    - **Note:** We recommend installing the Vulkan SDK directly from LunarG - https://www.lunarg.com/vulkan-sdk/ rather than relying on your distribution packages as those might not provide the shaderc library which Serenity relies on.

2.2.3 MacOS

Required software for KUESA™ :

- Xcode - latest version required
    - this installs command line tools and other packages such as git.
- CMake - https://cmake.org/,
    - Make sure that command line integration is included in the installation process
    - Make sure it is in your PATH

For the KUESA™ 3D Studio Renderer (Serenity) you'll need:

- Vulkan SDK - https://www.lunarg.com/vulkan-sdk/
    - Make sure to install the additional debug libraries when prompted during the install process
    - Make sure VULKAN_SDK is set accordingly in your environment

2.3 Required Dependencies

Most dependencies will be automatically fetched and downloaded during the configuration process.

However, either Qt or Serenity must be installed before attempting to build KUESA™ 3D Runtime.

2.3.1 Qt

KUESA™ 3D Runtime 2.0's optional Qt integration requires Qt 3D from Qt 5.15 or Qt 6 to be built. This is under Additional Libraries when installing Qt using the maintenance tool. The install has been tested against Qt 6.4.2.

The KUESA™ 3D Qt 3D integration library will be automatically enabled if Qt5 or Qt6 is found.

Ensure that you choose the appropriate compiler for your operating system - eg msvc2019_64 for Windows.

You might have to specify the path to the Qt install with CMAKE_PREFIX_PATH - this is covered in the installation section.

Note: Latest Qt 3D sources for 5.15 can be freely obtained from https://invent.kde.org/qt/qt/qt3d

2.3.2 Serenity

For the KUESA™ 3D Studio Renderer (Serenity) you'll need Freetype to be installed for its rmlUI overlay and support.

The recommended way to install Freetype on Windows is to use vcpkg - https://github.com/microsoft/vcpkg and do

```
vcpkg install freetype
```

2.4 Optional dependencies

2.4.1 QtCreator

If you wish to use KUESA™ 3D with Qt3D, QtCreator will be automatically installed with the version of Qt you have chosen.

2.4.2 Draco Compression

KUESA™ 3D supports the ``KHR_draco_mesh_compression`` glTF extension as defined at https://github.com/KhronosGroup/glTF/blob/master/extensions/2.0/Khronos/KHR_draco_mesh_compression/. Draco can be used to dramatically decrease the size of glTF files.

KUESA™ 3D comes with its own embedded version of the Draco library but by default, KUESA™ 3D will check if a system Draco (https://github.com/google/draco) mesh compression library is found and enable Draco support accordingly.

To build KUESA™ 3D with its own embedded version of the Draco library, run cmake like this:

```
cmake -DKUESA_FETCH_DRACO=ON ...
```

Note that KUESA™ 3D has last been tested with Draco 1.5.2

The glTF Inspector, provided with KUESA™ 3D Studio, is able to compress existing glTF 2.0 assets with Draco.

2.4.3 Documentation

If you wish to build the documentation locally, you'll need to install the following dependencies

- Doxygen - https://doxygen.nl
- doxybook2 - https://github.com/matusnovak/doxybook2
- mkdocs - https://www.mkdocs.org/
- mkdocs-material - https://squidfunk.github.io/mkdocs-material/

3. Introduction to KUESA™ 3D Runtime

KUESA™ 3D Runtime is divided into three main libraries: KUESA™ 3D Core, KUESA™ 3D Qt3D and KUESA™ 3D Serenity.

Both Kuesa Qt3D and KUESA™ 3D Serenity libraries link against KUESA™ 3D Core and provide similar APIs for loading a glTF2 model and rendering it.

When creating a KUESA™ 3D application, the application needs to build and link against a single library (either KUESA™ Qt3D or KUESA™ 3D Serenity). That decision is to be made based on whether you prefer to use Qt 3D and a deeper integration with Qt or if you'd rather use KUESA™ 3D's Serenity for the rendering.

3.1 KUESA™ 3D Core

The KUESA™ 3D Core library handles glTF 2 imports / exports. You can use it with either KUESA™ 3D Qt3D or KUESA™ 3D Serenity.

3.2 KUESA™ 3D Qt3D

KUESA™ 3D Qt3D provides wrappers to easily integrate with Qt and Qt 3D, along with:

- A QML plugin for integration with QtQuick
- The gltfViewer tool to preview glTF 2.0 files, including selecting cameras and animations

KUESA™ 3D Qt3D works with Qt 6 or Qt 5.15 (included for legacy purposes). If both are found, KUESA™ 3D will be built with Qt 6.

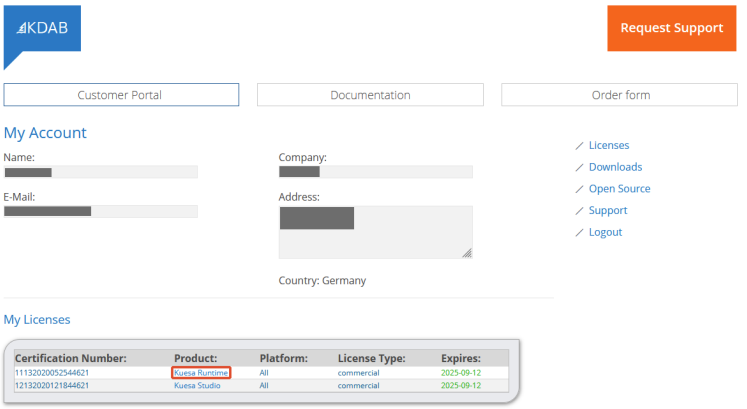
3.3 KUESA™ 3D Serenity

KUESA™ 3D Serenity provides wrappers to easily integrate with KUESA™ 3D Studio's 3D Renderer (Serenity) using any 2D technology as long as it supports Vulkan - eg Imgui, Rmlui.

4. Building and Installation of KUESA™ 3D Runtime

4.1 Location of KUESA™ 3D Runtime

In the customer portal (https://customers.kdab.com/), the list of 'My Licences' will have a link to the download for KUESA™ Runtime. Click the name of the product you wish to download:



You will be prompted to log into the site - use the username and password that you used for the customer portal.

This then opens a page to show you the list of the available downloads - choose the most recent one, unless you are developing with an older version and do not wish to migrate to a newer release.

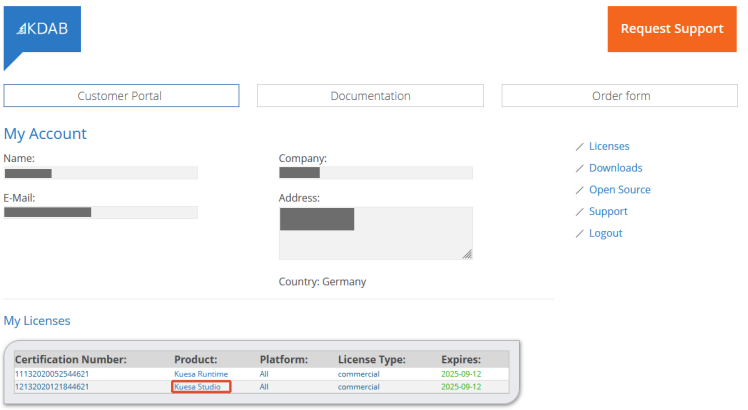
Index of /kuesa/runtime/releases

Name	Last modified	Size	Description
Parent Directory		-	
kuesa-runtime-1.2.0.zip	2020-06-11 11:02	513M	
kuesa-runtime-1.3.0.zip	2021-03-01 18:21	456M	
kuesa-runtime-2.0.1.zip	2022-09-14 12:59	1.0G	
kuesa-runtime-2.0.zip	2022-09-08 11:45	955M	

KUESA™ 3D builds with CMake. Unpack the KUESA™ sources in the directory of your choosing.

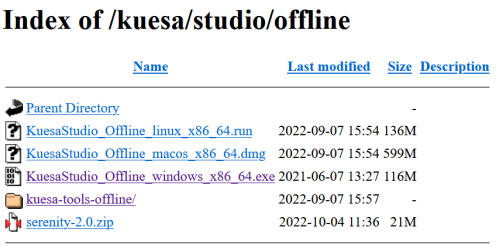
4.1.1 Location of Serenity zip file

As Serenity is included with the KUESA™ 3D Studio Licence, the zip will be found under the KUESA™ 3D Studio option:



You will be prompted to log into the site - use the username and password that you used for the customer portal.

This then opens a page to show you the list of the available downloads - this will allow you to download the zip from the following screen:



4.2 Building KUESA™ 3D Runtime

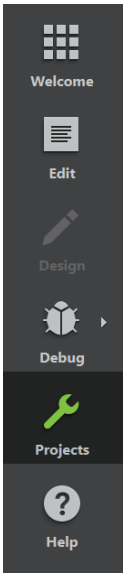
If you wish to build KUESA™ 3D Runtime against Qt3D only, then it is possible to build using QtCreator or using the command line.

If you wish to build KUESA™ 3D Runtime against Serenity, then it is only possible to build using the command line.

4.2.1 Building and installing with QtCreator (Qt3D with/without Serenity)

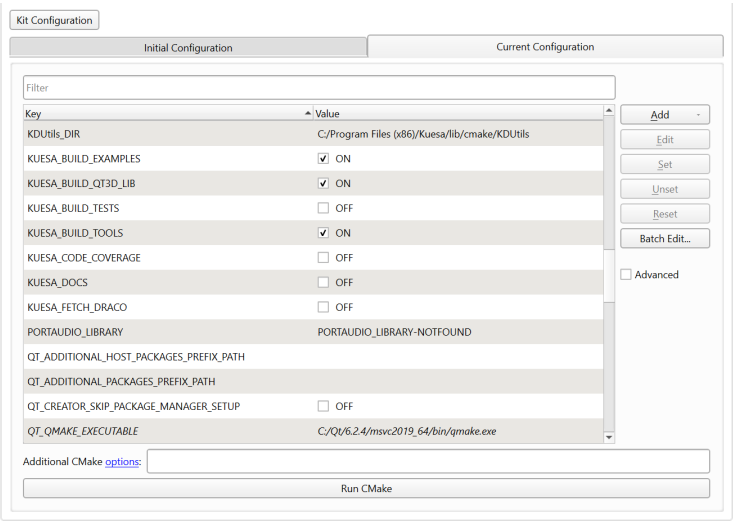
To open KUESA™ 3D Runtime in QTCreator, open CMakeLists.txt in QtCreator by clicking “Open File or Project...“, and configure the project using the version of Qt that you wish to use with KUESA™ 3D.

Project configuration tab selected in QtCreator:



4.2.1.1 Build Configuration

On the project configuration tab, you can select options to configure the build:



Available options are:

KUESA_BUILD_EXAMPLES=ON to enable building of the examples

KUESA_BUILD_TESTS=ON to enable building of the tests

KUESA_FETCH_DRACO=ON to fetch the Draco library

KUESA_DOCS=ON to build the documentation

CMAKE_INSTALL_PREFIX=/path/to/install to override the default installation path

CMAKE_PREFIX_PATH=/path/to/install directories that CMake will look to find the built dlls in once installed - can have more than one, separated by semi-colon

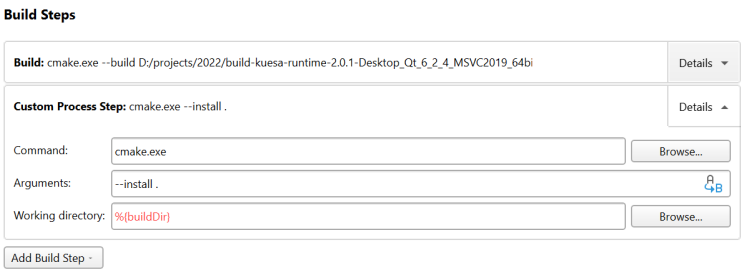
SERENITY_TARBALL=/path/to/serenity.zip to enable building Serenity and the Kuesa Serenity library

Note: SERENITY_TARBALL will need to be added to the options if it is to be used, as it is not currently listed.

One further build step is required to install KUESA™ 3D Runtime - under the Build Steps a Custom Process Step should be added with the following code:

```
cmake --install .
```

The build steps section should then look like this:



4.2.1.2 Environment variables

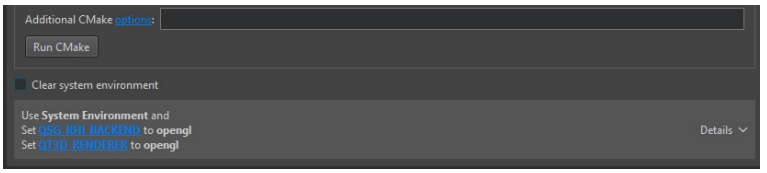
In addition, there need to be two environment variables set to “opengl”:

QSG_RHI_BACKEND
QT3D_RENDERER

This can be done one of two ways, depending on your version of QtCreator.

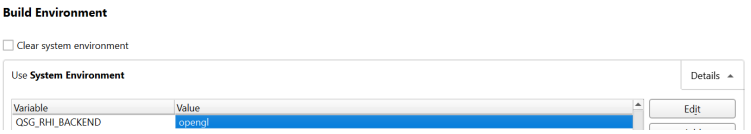
Option 1:

Underneath the build configuration there may be an option to set environment variables for CMAKE:



Option 2:

If the previous options are not available, these values can be set in the Build Environment further below the build configuration:



These will then be set as additional environment variables within the Qt project.

4.2.1.3 Running the build

Once you have chosen the configuration you wish to build, go to the Edit tab and run CMake.

4.2.2 Building and installing with the command line (all configurations)

Using the command line it is possible to build KUESA™ 3D Runtime for either Qt3d or Serenity.

Open a terminal console and navigate to the directory that you unpacked Kuesa Runtime to.

Note: On Windows you want to run the Native Tools command prompt for VS.

Then run the commands in order of the sections below depending on what you want to build.

Note: You might need to run your console as administrator in order to successfully install KUESA™ 3D to its default location.

Note: The configure step might take some time as missing dependencies will be fetched online.

4.2.2.1 Creating the build directory

This should be done for all installs:

```
mkdir build
cd build
```

4.2.2.2 Build configuration

Kuesa Runtime is configured for building by running CMake. The following CMake options can be set:

KUESA_BUILD_EXAMPLES=ON to enable building of the examples

KUESA_BUILD_TESTS=ON to enable building of the tests

KUESA_FETCH_DRACO=ON to fetch the Draco library

SERENITY_TARBALL=/path/to/serenity.zip to enable building Serenity and the Kuesa Serenity library

KUESA_DOCS=ON to build the documentation

CMAKE_INSTALL_PREFIX=/path/to/install to override the default installation path

CMAKE_PREFIX_PATH=/path/to/install directories that CMake will look to find the built dlls in once installed - can have more than one, separated by semi-colon

CMAKE_BUILD_TYPE=Release to ensure the correct version is built

Note: options are passed to cmake by prefixing the option by -D

Note: The configure step might take some time as missing dependencies will be fetched online.

Examples of some common CMake configurations:

Configuring KUESA™ 3D Runtime to build the KUESA™ 3D Qt 3D library only

```
cmake -DCMAKE_PREFIX_PATH=<your qt path> -DKUESA_
BUILD_EXAMPLES=ON ..
```

Configuring KUESA™ 3D Runtime to build the KUESA™ 3D Serenity library only

```
cmake -DSERENITY_TARBALL=<your serenity.
zip> -DKUESA_BUILD_EXAMPLES=ON -DCMAKE_BUILD_
TYPE=Release ..
```

Configuring KUESA™ 3D Runtime to build both the KUESA™ 3D Qt 3D and KUESA™ 3D Serenity libraries

```
cmake -DCMAKE_PREFIX_PATH=/path/to/Qt -DKUESA_
BUILD_EXAMPLES=ON -DCMAKE_BUILD_TYPE=Release
-DKUESA_FETCH_DRACO=ON -DSERENITY_TARBALL=/path/
to/serenity.zip ..
```

Note: On Windows, if you have used vcpkg to install Freetype, you can set -DCMAKE_TOOLCHAIN_FILE and -DVCPKG_TARGET_TRIPLET like:

```
cmake -DCMAKE_PREFIX_PATH=/path/to/Qt -DCMAKE_
TOOLCHAIN_FILE=path/to/vcpkg/scripts/buildsystems/
vcpkg.cmake -DVCPKG_TARGET_TRIPLET=x64-windows
-DCMAKE_BUILD_TYPE=Release -DKUESA_BUILD_
EXAMPLES=ON -DKUESA_FETCH_DRACO=ON -DSERENITY_
TARBALL=/path/to/serenity.zip ..
```

4.2.2.3 Installing KUESA™ 3D Runtime

Once KUESA™ 3D Runtime has been successfully configured for building, it can be built and installed by running the following commands. *This might need administrator rights when executed*:

```
cmake --build . --config Release
cmake --install .
```

4.2.3 Building with the command line - example

The following commands would build and install KUESA™ 3D Runtime for Qt 3D on Windows, where the Qt Path for version 6.4.2 is C:\Qt\6.4.2\msvc2019_64 - adjust the path to match the version of Qt that you have installed:

```
mkdir build
cd build
cmake -DCMAKE_PREFIX_PATH=C:\Qt\<your Qt
version>\msvc2019_64 -DKUESA_BUILD_EXAMPLES=ON ..
cmake --build . --config Release
cmake --install .
```

Example of command using version 6.4.2:

```
cmake -DCMAKE_PREFIX_
PATH=C:\Qt\6.4.2\
msvc2019_64 -DKUESA_
BUILD_EXAMPLES=ON ..
```

4.2.4 Building Serenity - standalone

It is possible to build Serenity as a standalone, without KUESA™ 3D - the commands would be as follows:

```
mkdir build
cd build
cmake .. -DCMAKE_TOOLCHAIN_FILE=path/to/vcpkg/
scripts/buildsystems/vcpkg.cmak -DCMAKE_BUILD_
TYPE=Release -DCMAKE_FIND_DEBUG_MODE=ON >log-
config.txt

cmake --build . --config Release
cmake --install .
```

5. Deployment of KUESA™ 3D Runtime

Once KUESA™ Runtime has been built, it will need to be deployed.

5.1 Clearing Stale Caches

If you've updated KUESA™ 3D or have used KUESA™ 3D previously, it is recommended to clear the Qt 3D cached shader files.

The cache is located at Qt::TempLocation - e.g. "C:/Users/<USER>/AppData/Local/Temp" on Windows and "/tmp" on Linux.

Remove all the .qt3d files present at that location to ensure KUESA™ 3D rebuilds all the shaders with all the updates.

5.2 KUESA™ 3D Qt3D

Binaries build against the KUESA™ 3D Qt3D library will require:

- The KuesaQt3D QML module to be in the qml import path, it can be set with the *QML_IMPORT_PATH* environment variable:
    - e.g. for Windows the install path is C:/Program Files (x86)/Kuesa/qml

```
QML_IMPORT_PATH=/path/to/Kuesa/install/qml
```

- The KUESA™ install path (e. g. `C:/Program Files (x86)/Kuesa/bin`) needs to be in the CMAKE_PREFIX_PATH variable for the build configuration of any project that uses KUESA™ 3D - this can have more than one directory, separated by a semi-colon.
- The KUESA™ install path (e. g. `C:/Program Files (x86)/Kuesa/bin`) need to be in PATH (Windows) / LD_LIBRARY_PATH (Linux/macOS)

- The location of Qt dlls to be in the PATH (Windows) / LD_LIBRARY_PATH (Linux/macOS)
    - e.g. <your qt path>\msvc2019_64\bin for Windows, macos\bin for macOS

Troubleshooting QML imports can be done by setting the environnement variable **QML_IMPORT_TRACE=1**

5.3 KUESA™ 3D Serenity

No deployment is required for KUESA™ 3D Serenity as there are no caches as there are for the QML plugin.

5.4 Running the examples

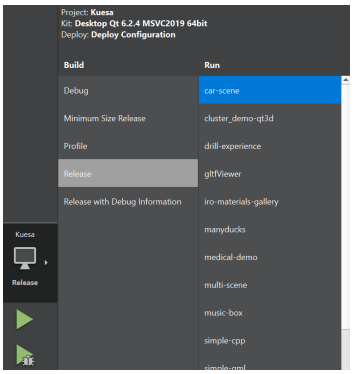
5.4.1 KUESA™ 3D Qt3D

Once you have completed the steps in the deployment of KUESA™ 3D Runtime, open up CMakeLists.txt in QtCreator.

Ensure that you configure the project to use the version of Qt that KUESA™ 3D was compiled with.

Select the Release version of the build and select the example that you wish to run in the run Menu.

Selecting an example to run in QtCreator:



5.4.2 KUESA™ 3D Serenity

If KUESA™ 3D Runtime with the Serenity backend was built with -DKUESA_BUILD_EXAMPLES=ON, the examples can be found in build/bin

6. Introduction to KUESA™ 3D Studio

KUESA™ Studio makes the unique KUESA™ 3D workflow complete, whether you are working with Blender or Maya. It allows you full access to the range of features provided with KUESA™.

KUESA™ Studio for Maya / Blender is a plugin that consists of

- An extension for the glTF 2.0 exporter to convert 3D scenes to glTF 2.0 files including the accompanying binary buffers and textures, supporting KUESA™ 3D specific features like Iro materials and KUESA™ layers.
- Integration of KUESA™ 3D materials for Eevee (Blender) or Viewport 2.0 (Maya) in order to provide real-time WYSIWYG during modeling and animation. The designer can check the visual result inside of the 3D application without exporting or waiting for an engineer build an app beforehand. This allows all team members to stay focused in their area of expertise.
- Additional utilities that support the KUESA™ 3D workflow.

The aim of KUESA™ 3D is to ensure the fastest workflow possible. Since KUESA™ 3D moves the work 'up the pipeline' you can create your scene entirely in your 3D application and use the tools you are used to without going the extra mile via a proprietary editor.

If you want to learn more about the KUESA™ 3D workflow or KUESA™ 3D in general, please checkout out https://www.kuesa.com/

KUESA™ is a tool designed to boost your 3D design-to-code workflow,

Check out the KUESA™ 3D videos : www.kuesa.com/videos

7. Installation Of KUESA™ 3D Studio

KUESA™ 3D Studio comes with a common installer for all KUESA™ 3D Studio components.

You can choose between two options:

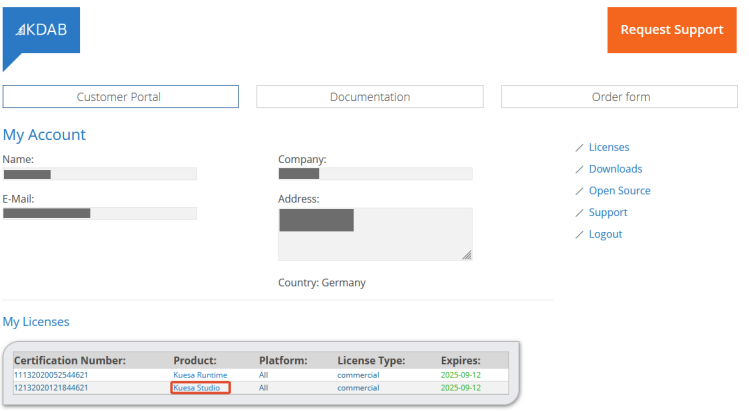
- An online installer that installs KUESA™ 3D Studio using the Qt Maintenance Tool, downloading the packages during installation.
- A standalone offline installer that is downloaded beforehand.

KUESA™ offers provides adds on for multiple 3D authoring tools such as Blender and Maya.

Both will be explained in the following sections, including slight differences for Blender and Maya

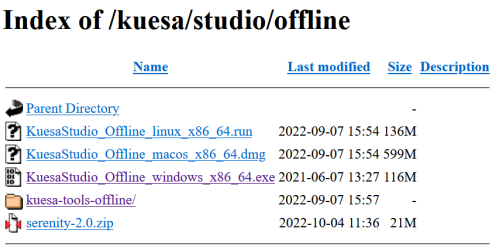
7.1 Location of KUESA™ 3D Studio Offline Installer

In the customer portal (https://customers.kdab.com/), the list of 'My Licences' will have a link to the download for KUESA™ Studio. Click the name of the product you wish to download:



You will be prompted to log into the site - use the username and password that you used for the customer portal.

This then opens a page to show you the list of the available downloads. Select the correct package for your operating system:



Unpack the KUESA™ 3D sources in the directory of your choosing.

7.2 Offline Installer for Blender

After downloading and starting the offline installer, please follow the on-screen instructions. During the installation you will be prompted to select the Blender binary. This way you can install the plugins for a specific version.

7.2.1 Choose the components to be installed.

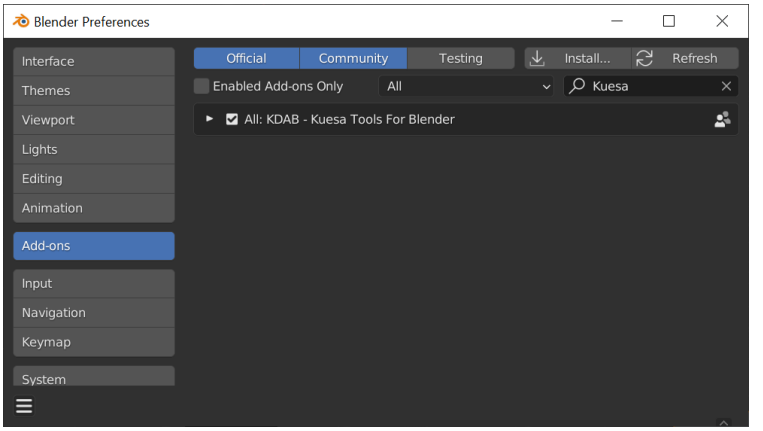
The minimum requirements would be to install the KUESA™ 3D Studio Blender add-on. To inspect the exports, you should also install the KUESA™ 3D Studio Tools which include the KUESA™ 3D Studio glTFInspector.

- Accept the license,
- Confirm the start menu shortcut options
- Browse for your Blender binary.
- Click on 'Install' to confirm.

The installer will add the KUESA™ 3D add-ons for Blender to your personal Blender add-ons folder that lives in the sub-hierarchy of your home folder.

After the installation is finished, start Blender to enable the add-on. Under Edit-> Preferences -> AddOns, search for Kuesa and ensure

that the AddOn has been loaded. It will then be available under the N-Panel.

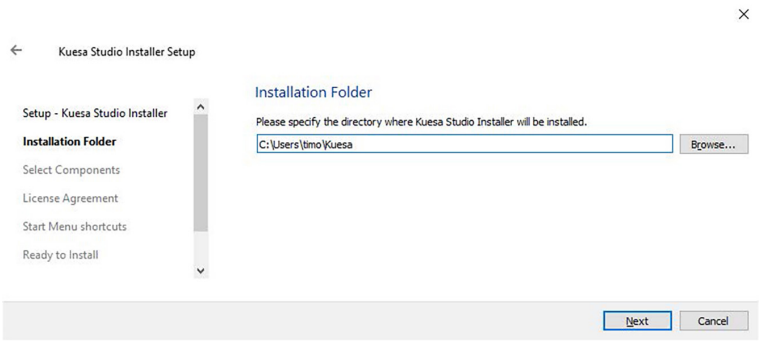


7.3 Offline Installer for Maya

After downloading and starting the offline installer, please follow the on-screen instructions.

7.3.1 Select the folder for KUESA™ 3D Studio.

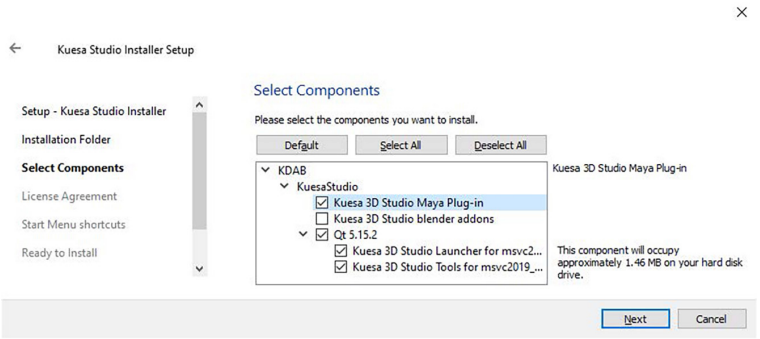
This plugin installer is not specific to Maya, so you will not be asked for your Maya environment. The Maya plugin will not be installed permanently into your Maya installation, so you are free to choose any location.



7.3.2 Choose the components to be installed.

The minimum requirements for Maya would be to install the KUESA™ 3D Studio Maya plugin and the KUESA™ 3D Studio launcher. To inspect the exports, you should also install the KUESA™ 3D Studio Tools which include KUESA™ 3D Studio glTFInspector.

The installer offers multiple components that are available to match your workflow pipeline.



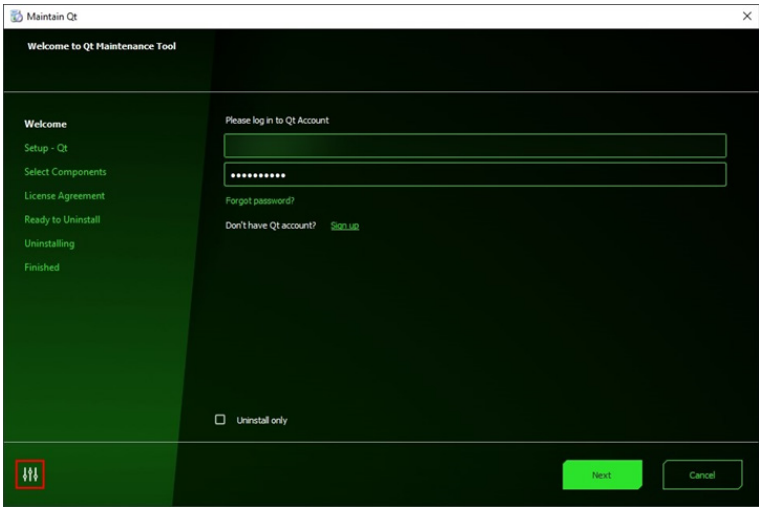
- Accept the license,
- Confirm the start menu shortcut options
- Click ‘Next’.
- Click ‘Finish’.

After confirming with ‘Finish’ you will be provided with the KUESA™ 3D Studio components inside your start menu.

In the section Launching KUESA™ 3D Studio for Maya you will learn how load Maya with the KUESA™ 3D Studio plugins.

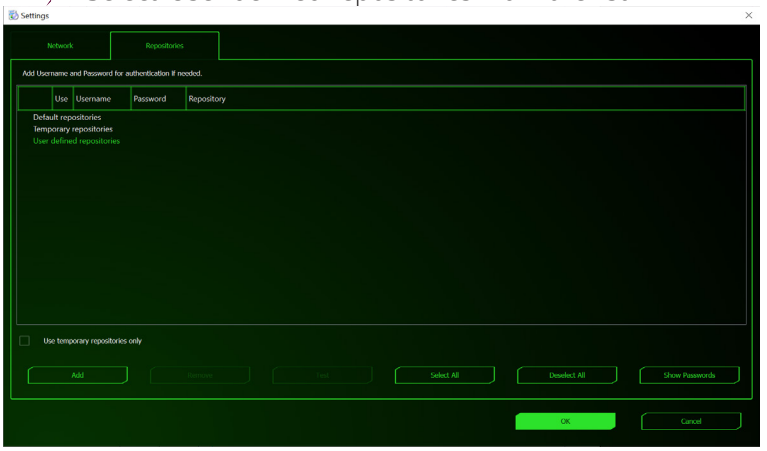
7.4 Qt Online Installer

Run the Qt maintenance tool and click on the settings icon at the bottom left corner.

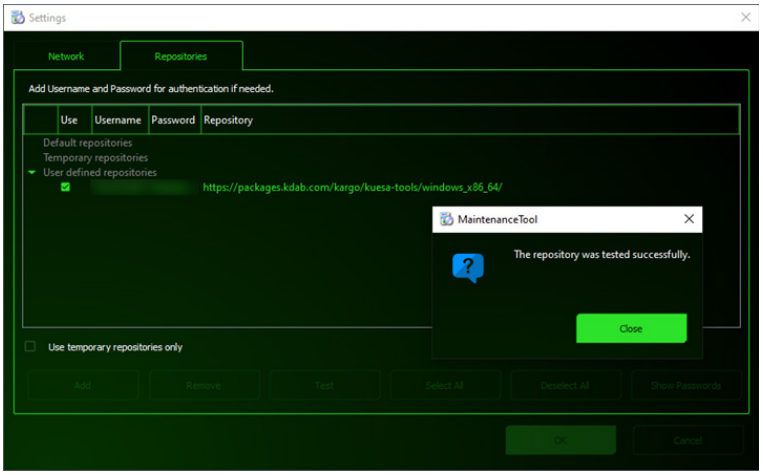


Qt is a cross-platform toolkit for creating graphical user interfaces: www.qt.io

- Open the 'Repositories' tab,
- Select 'User defined repositories' from the list



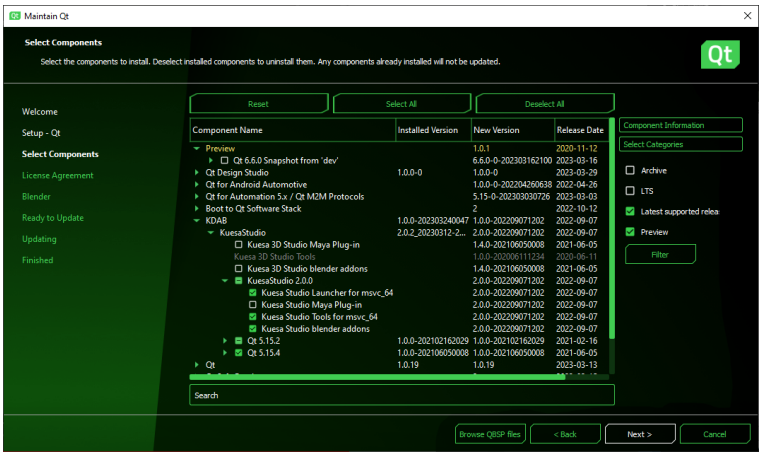
- Click 'Add'
- Add the repository: https://downloads.kdab.com/kuesa/studio/online/windows_x86_64/
- Click 'Test'
- Enter your user credentials,
- Click on 'OK' to verify that the entered information is correct, and the license is valid.



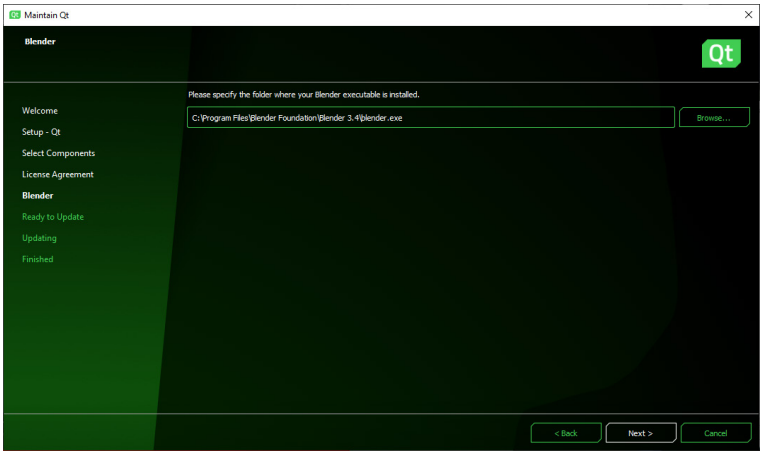
- Click ‘OK’ to go back to the welcome screen of the Qt maintenance tool,
- Click ‘Next’,
- Select ‘Add or remove components’
- Click ‘Next’.
- Wait for the repositories information to be updated.

Options for Blender

In the list under ’KDAB’ choose the components to be installed. For Blender this would be at least the KUESA™ 3D Studio Blender add-ons.

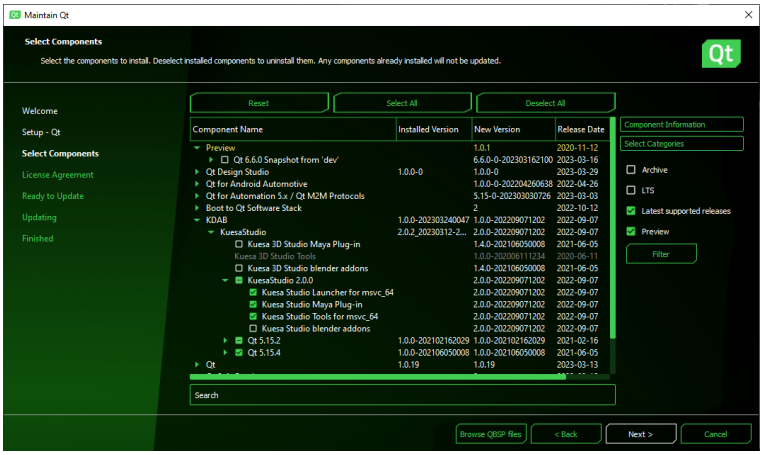


You will also be asked for the path to your Blender install:



Options for Maya

In the list under 'KDAB' choose the components to be installed. For Maya this would be at least the KUESA™ 3D Studio Maya plugin and the KUESA™ 3D Studio launcher.



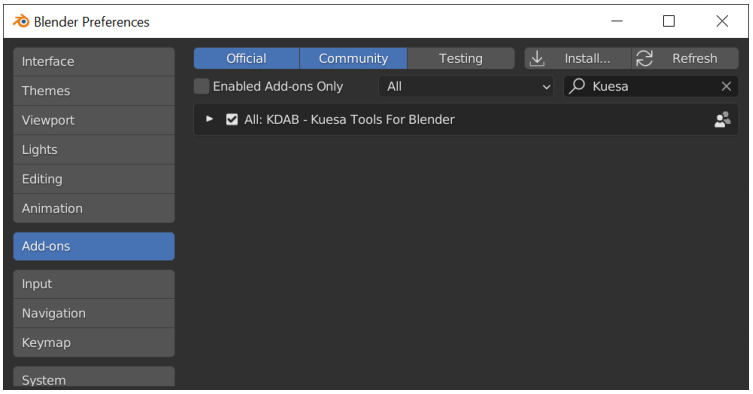
- Click on 'Next' to run the installation.

When finished, you will find KUESA™ Studio inside your Qt folder.

7.5 Launching KUESA™ 3D For Blender

Once the installation has been completed, the last step is to ensure that the AddOn in Blender has been loaded.

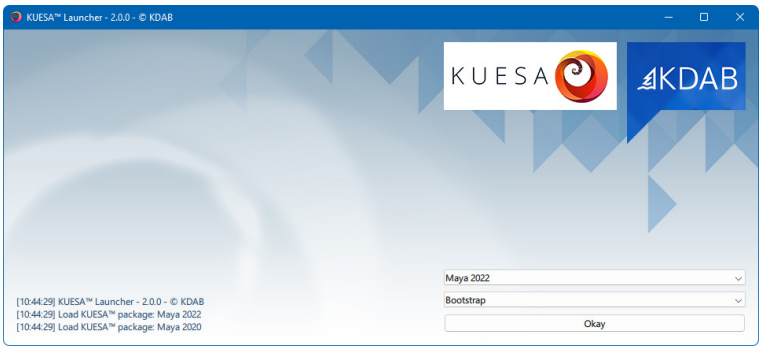
Under Edit-> Preferences -> AddOns, search for Kuesa and ensure that the AddOn has been loaded. It will then be available under the N-Panel.



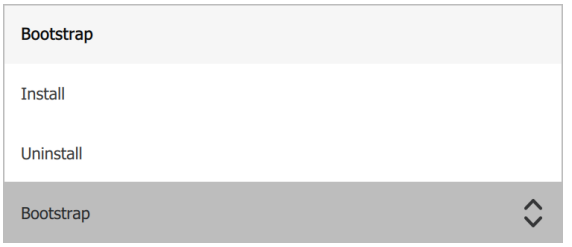
7.6 Launching KUESA™ 3D For Maya

After the installation of KUESA™ 3D Studio, the KUESA™ 3D plugins for Maya are not permanently hooked into Maya by default and therefore needs to be activated first.

To achieve this KUESA™ 3D For Maya comes with a Launcher.



After selecting the Maya version you want to work with, you are provided with three options: Bootstrap, Install and Uninstall. After choosing one of the options click Okay to apply.



7.6.1 Bootstrap

This will fire up the chosen Maya Version (if available) and load the KUESA™ 3D plugins dynamically.

Choose this option if you do not want to install the KUESA™ 3D plugins permanently into your Maya installation. The next time you start Maya without the launcher the KUESA™ 3D plugins will not be loaded again. This is useful if you are working on multiple projects in parallel or just do not want to have the KUESA™ 3D plugins around each time you are using Maya and want to keep your Maya install clean. This is useful if you are working with multiple Maya versions in parallel.

7.6.2 Install

This will permanently hook the KUESA™ plugins into your personal Maya startup script. After that, the KUESA™ 3D plugins are loaded automatically whenever you start up Maya.

Choose this option if you do not want to start Maya using the Bootstrap option or just want to have the KUESA™ 3D plugins available permanently. This is also useful when you are working with export scripts and need to be able to access KUESA™ functionality from CLI scripts.

7.6.3 Uninstall

This will permanently unhook the KUESA™ 3D plugins from your personal Maya startup script and therefore is the opposite of Install.

Choose this option if you do not need KUESA™ for Maya anymore or if you want to switch back to using the bootstrap method.

8. Uninstalling KUESA™ 3D

Removing KUESA™ 3D from your system depends on the method it was installed, and what was installed.

8.1 Uninstalling KUESA™ 3D Runtime

There are a number of places to remove KUESA™ 3D.

- Remove the install folder. On Windows this is C:\Program Files (x86)\kuesa by default, on macOS this would be under /usr/local. This could be overridden by the path set with CMAKE_INSTALL_PATH.
- Clear the Qt 3D cached shader files.
    - These cache is located at Qt::TempLocation which is usually "C:/Users/<USER>/AppData/Local/Temp" on Windows and "/tmp" on Linux.
    - Remove all the .qt3d files present at that location to ensure KUESA™ 3D rebuilds all the shaders with all the updates.
- Remove the runtime folder (including the build directory created either manually if installed via commandline, or the build directory created by QtCreator)
- Remove any Kuesa (and/or Serenity) dlls. from the folders specified in CMAKE_PREFIX_PATH - this will usually include the directory where the Qt dlls are installed (if you have installed with Qt3D). On macOS this would be under /usr/local.

8.2 Uninstalling KUESA™ 3D Studio

If KUESA™ 3D Studio was installed using the online installer, run the maintenance tool again and deselect the KUESA™ 3D Studio components. This will remove the relevant items from your system.

Otherwise, if the offline installer was used, remove the programs as normal for the operating system (e.g. for windows, using Add/ Remove programs).

8.2.1 For Blender

To remove the plugin, go to Perferences and the Add-ons tab. Search for KUESA, and onve found, expand the the box and click remove. After that, check that the plug-in directory has been correctly removed.

8.2.2 For Maya

This is only required if the tools were permanantly hooked to Maya. See Section 7.6 for more details, particularly the uninstall subsection.

9. Other Information

9.1 Reference Documentation

The reference documentation is available at: https://kdab.github.io/kuesa/

9.2 Support

Please report issues to **kuesa-support-group@kdab.com**

KDAB will happily accept external contributions; however, **all** contributions will require a signed Contributor License Agreement.

9.3 3rd-Party

KUESA™ Qt3D uses the following 3rd-party software:

- Qt 5.15 or later, licensed under LGPLv3: https://www.qt.io
- Draco, licensed under Apache License 2.0 and authored by Google Inc.: https://github.com/google/draco
- MikkTSpace, licensed under a custom license and authored by Morten S. Mikkelsen: http://mmikkelsen3d.blogspot.com/

9.4 Python Binding

A Python binding is available based on PySide 2. Please check the build instructions in ``src/python`` and the sample application in ``examples/kuesa/python``