Moai Qt Host
Maintenance manual

February 7, 2013

Contents

Chapter 1

Introduction

1.1 Purpose and coverage

The purpose of this document is to serve as a maintenance guide for Moai Qt Host application.

The target audience of this document is anyone who is interested in developing, modifying or using Moai Qt Host.

1.2 Product and environment

The product Moai Qt Host is a host application for Moai SDK. The primary task of the application is to offer the OpenGL drawing surface and relay user input to the Moai engine. Game developers using Moai are the primary users of Moai Qt Host.

Moai Qt Host is designed for desktop environments including Windows, OS X and Linux. It is designed for development and release of games, essentially replacing the GLUT host by offering better performance.

1.3 Definitions, terms and abbreviations

The definitions, terms and abbreviations are listed in table 1.1.

1.4 References

1. Qt Licensing. [accessed on 15.10.2012]. Available at: `http://qt.digia.com/Product/Licensing/`

Table 1.1: Terms and abbreviations

Term	Explanation
AKU	Moai SDK API for host application.
Application	In this document application refers to Moai Qt Host application.
C++	Programming language.
FMOD	Platform independent audio system.
GLUT host	An OpenGL demo Moai host that runs on Windows, OSX and Linux. It's not supposed to be used for development.
Host	In this document a host is an interface application between a device and the Moai game engine.
Lua	A scripting language
Moai / Moai SDK	2D game development environment
OpenGL	Open Graphics Library is a platform independent graphics library.
Qt	Platform independent software anf graphical user interface development environment.
Terminal	Textual command interface.
UNTZ	Open source audio library.

2. Mozilla Public License version 2.0. [accessed on 04.02.2013]. Available at: `http://www.mozilla.org/MPL/2.0/`

1.5 Document overview

Chapter two contains application overview.

Chapter three contains information on application environment.

Chapter four contains information on maintenance staff and maintenance tools.

Chapter five contains an overview or the implementation of Moai Qt Host.

Chapter six contains information on files and components.

Chapter seven contains maintenance instructions.

Chapter eight contains infromation on reporting and documents.

Chapter nine contains information on known problems, curiosities and illogicalities.

Chapter ten contains abandoned alternative solutions.

Chapter ten contains thoughts on further development.

Chapter 2

Outline

2.1 Purpose

The purpose of Moai Qt Host is to provide an open source host for Moai SDK for desktop computers.

2.2 Users

The primary users of Moai Qt Host are game developers that use Moai SDK.

2.3 Features

Moai Qt Host acts as an adapter between Moai SDK and system environment. The application provides an OpenGL drawing surface for Moai. The application also relays user input to Moai.

2.4 General restrictions

See 2.5.

2.5 Licences

The code of Moai Qt Host and related documents are MPL-2.0 licensed [2]. Qt open source –version [1] and Moai SDK licencing restrictions may apply.

Chapter 3

Application environment

3.1 Hardware environment

Any hardware capable of running supported operating systems is adequate to run Moai Qt Host. Currently Moai officially supports only OpenGL 2.0 so any OpenGL features that are on newer versions of OpenGL won't work. A keyboard and a pointer device such as a mouse are required.

3.2 Software environment

The application has tested versions for Windows 7, OS X Mountain Lion and Ubuntu Linux 12.04.

3.2.1 Programming language

The application is implemented with C++.

3.2.2 Compiler

The application was compiled with Qt Creator, so Qt MOC compiler was used. Furthermore following compilers were used on different platforms to compile Moai SDK: Microsodt Visual Studio 2010 compiler on Windows, GCC on Linux, GCC and Clang on OS X.

3.2.3 Browsers

The application requires no web browser.

3.2.4 Database

The application does not use a database.

3.3 Network environment

No network connectibility is required to run or install the application.

3.4 Third-party software and components

Third-party components are listed in table 3.1.

Table 3.1: Third-party components

Component	Version
Moai SDK	1.3 (Build 160)
Qt Creator	2.4.1
Qt libraries	4.8.1
Qt SDK	1.2.1

3.5 Compatibility

The application is compatible with above-mentioned operating systems and third-party components.

Chapter 4

Maintenance staff and tools

4.1 Maintenance staff

The maintenance of the application is done by the developers.

4.2 Planned update interval

No update schedule is in place.

4.3 Responsibilities and confidentiality

There are no specified responsibilities or confidentiality issues. Moai Qt Host is done for use and reproduction ”as is” and developers make no warranty as to its use, performance or functionality. Also developers make no warranties, conditions, representations, or terms (express or implied whether by statute, common law, custom, usage or otherwise) as to any matter including without limitation noninfringement of third party rights.

4.4 Tools and procedures

There are no specific tools or procedures concerning the maintenance of the application.

4.5 Support software

There is no specific support software concerning the maintenance of the application.

Chapter 5

Implementation overview

5.1 Architecture and design schemes

5.1.1 Software architecture

The application provides the drawing surface and callback functions for Moai SDK, to which it is connected via the AKU interface. Moai Qt Host is implemented with Qt so it uses Qt libraries in implementation. The main architecture of Moai Qt Host is shown in figure 5.1

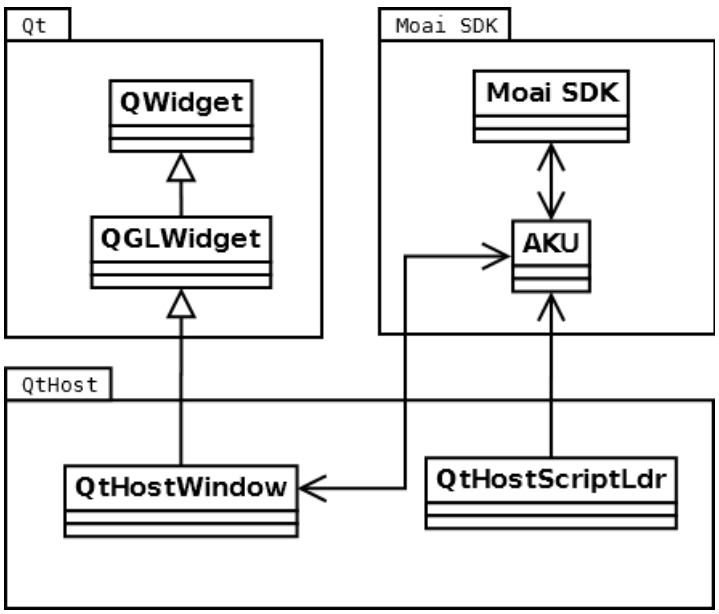

Figure 5.1: Moai Qt Host modules

5.1.2 Hardware architecture

There is no hardware architecture connected to the application.

5.1.3 Database architecture

Moai SDK does not use a database.

5.1.4 Network architecture

No network architecture is needed in the current version.

5.2 Modules

The design ideology of Moai Qt Host is modular, so new modules can be easily added. Currently there are only two modules. See figure 5.1. QtHost-Window module provides drawing surface for Moai and handles input events. QtHostScriptLdr module reads in given Lua scripts. Initialization of Moai and host modules are done in the main-function of the application.

5.3 Datastructures

Not specified.

5.4 Database

No database is used.

5.5 User interface

There is no user inferface besides the main window frame.

5.6 Printing

There is no printing functionality.

5.7 Errors and recovery

All errors are handled in-program.

5.8 Security and protection

Not specified.

Chapter 6

Files and components (Physical configuration of application)

6.1 System files and location

The folders and files associated with Moai Qt Host are shown in figure 6.1.

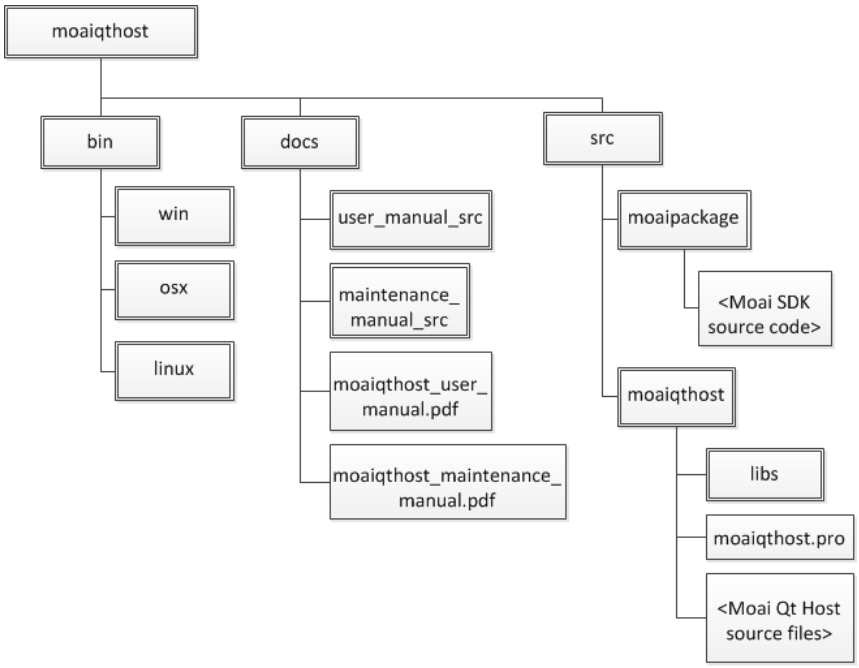

Figure 6.1: Moai Qt Host folder structure

Chapter 7

Maintenance instructions

7.1 Maintenance guidelines

In principle new features are supposed to be added only if Moai SDK needs support for new features. Only feature that may be recommended to extend is different keyboard input support because currently all keyboard buttons are not supported.

7.2 Adding a new component

New components should be added only when it simplifies the structure of Moai Qt Host.

7.3 Programs

Currently there is no reason to add new modules to Moai Qt Host. New functions may be added if they are needed for supporting new functionality on Moai SDK.

7.4 Compilation and installation

Moai Qt Host should be compiled with Qt Creator. Compilation from command line has not been tested.

7.5 Configuration files and log files

Configuration of Moai Qt Host is done with qmake configurations in the Qt-project file. For more information check the User Manual.

New configuration parameters should only be added if Moai SDK configurations change. Each configuration should include all the required modifications, such as required libraries and preprocessor macros. Different operating systems should also be noted.

When naming configurations, the name should reflect the corresponding Moai SDK configuration. Only small letters and underscore should be used in configuration names.

7.6 Database

There is no database.

7.7 Connections

7.7.1 Software connections

Moai Qt Host interacts with Moai SDK and Qt. On Linux, there are additional dependencies, which are listed in the build instructions of Moai SDK.

7.7.2 Hardware connections

All hardware interaction is handled by Qt.

7.7.3 Network connections

Currently there are no network connections.

7.7.4 Other connections

Curently there are no other connections.

7.8 Portability

7.8.1 Software environment

Moai Qt Host should be portable to new operating systems if Qt libraries and Moai SDK are available.

7.8.2 Hardware environment

Portability to new hardware environments depends on Qt because it handles all the hardware interaction.

7.9 Localization

No localization is available.

7.10 Testing

It's recommed to use Moai SDK samples for testing. If Moai Qt Host is modified all supported platforms (Windows, Linux, OS X) should be tested.

7.11 Error situations

Currently there are no known error situations.

Chapter 8

Reporting and documents

8.1 Notice of change or correction

Every issue found in the application should be announced on the *Issues* page of the GitHub repository. Issues must be well explained for the developers to be able to understand and fix them. With clear bugs the used system environment should be specified along with the required steps to reproduce the bug.

To implement a desired feature or fix a bug the repository on GitHub can be forked. Once the changes are made and pushed to the forked repository a *Pull request* should be made. After that someone with push access to the original repository can merge the changes to update the application. For more information see `https://help.github.com/articles/using-pull-requests`.

8.2 System documents and their location

This document as well as the User Manual along with their LaTeX sources can be found from the repository under the folder *docs*.

8.3 Documents that are not maintained

All of the documents should be kept up to date at all times.

Chapter 9

Known problems, curiosities and illogicalities

9.1 Unofficial Linux implementation of Moai SDK

Currently there is no officially supported version of Moai SDK on Linux platform, so an unofficial version by spacepluk is used (`https://github.com/spacepluk/moai-dev`). Because of this configuring the Moai SDK on Linux might be more difficult. Probably for the same reason mp3 music files don't work with UNTZ on Linux.

9.2 Features that have not been tested

Following Moai Qt Host qmake configurations have not been tested: audiosampler, fmod_designer, debugger.

9.3 3rd party debuggers

Using 3rd party debuggers (such as ZeroBrane Studio) with Moai Qt Host has not been tested and might not work.

9.4 Working directory

Moai Qt Host assumes that the working directory is the same where the binary is run from.

Chapter 10

Abandoned alternative solutions

10.1 Moai SDK Qt-project

The first approach taken for building Moai Qt Host was to build the whole Moai SDK with Qt Creator by creating a Qt-project for it. This was abandoned because creating the project proved to be difficult and time consuming, partly because of multiple 3rd party libraries involved. Also, even a working Qt-project would be a problem in maintenance sense since it would need to be updated every time the Moai SDK file structure changes.

10.2 Update and rendering on different timers

Initially during development the Moai Qt Host used separate timers (QTimer-class) for the update and render functions. This caused the game to sometimes update multiple times without rendering, or vice-versa. This resulted in an uneven framerate, so it was concluded that using one timer for both the update and render function would be safer.

Chapter 11

Thoughts on further development

11.1 ”Open file” –dialog

If no scripts are given as parameters when starting Moai Qt Host, the files could be inquired from the user via a dialog. Providing this feature would require changes to the script-module. The actual dialog could be easily implemented with Qt.

11.2 Mobile OS support

Qt supports several mobile operating systems, like Symbian and MeeGo. Support for Android and iOS environments is being planned. Thus, using Qt-libraries on these platforms to provide required functionality for Moai Qt Host should be possible.

The host-part of Moai Qt Host should be portable to mobile devices with relatively small changes. Larger problem is the Moai SDK itself, which must be compiled for target operating systems. Perhaps the easiest way to do this would be to compile the Moai SDK with Qt Creator (refer to section 10.1).

11.3 Separating input handling from the window-module

Mobile operating systems might require some changes to the way the input is currently handled. Required changes could be done within the current

window-module, but this might cause some unnecessary features to be included in desktop and mobile environments.

Another way would be to use different input modules for desktop and mobile environments. This would require separating the input handling from the current window-module. One way to implement this would be to create a transparent widget over the main window, and have that widget act as the input module.

11.4 Sensor support for mobile systems

If support for mobile operating systems is provided, it would also be preferable that all the related sensors (compass, location, etc.) are supported as well. This would require changes to the window-module of Moai Qt Host, but the changes themselves should be quite simple. Sensors supported by Moai SDK can be found by examining the AKU-interface.

11.5 Dynamic script reloading

GLUT-host gives user the ability to reload .lua-files when they are modified during runtime. This is a very useful feature when using the host when developing your game.

Ideally this same functionality would be implemented to Moai Qt Host by using Qt. This would keep the implementation same for all platforms. The script-module needs to be modified, but there should be no need to touch the other modules. If a timer is required, we can just make a new one inside the script-module, instead of using the one in window-module. Enabling script reloading could be done the same way as it is done in GLUT-host, which is with an e-flag in parameters when starting the host.