QEO SDK INSTALLATION GUIDE

Version 0.15.0

Revised: 3 October 2013. Printed: 23 March 2010.
N/A v1.0.

Table of Contents

1 Installation Guide

About this document

This document provides all the information needed to install the Qeo SDK.

What You Need

In order to install the Qeo SDK you need:

- The Qeo SDK Archive (referred to as **QeoSDK.ZIP** hereafter)
- This document

2 Supported Platforms

Supported Platforms

The Qeo SDK supports the following target platforms:

Android

- ARM (**arm-linux-androideabi**)
- x86 (**x86-linux-androideabi**)

The following languages are supported:

- Java
- JavaScript

Linux

- x86 (**i686-linux):** binaries require glibc_2.7 (and are built with gcc 4.4.7))
- RaspberryPi using the hard-float ABI (**arm-linux-gnueabihf**); a toolchain for this platform is available from https://github.com/raspberrypi/tools (validated with arch-linux and Raspbian "wheezy")

The following languages are supported:

- C
- JavaScript (provided you do your own platform integration)

Validated Development Platforms

The Qeo SDK supports the following development environment platforms:

For Android development:

- Linux (RedHat Enterprise Edition 6, Mint 15)
- MacOSX (10.7.5 and 10.8.4)
- Windows (XP/7)

For C development:

- Linux (RedHat Enterprise Edition 6, Mint 15 + Ubuntu 32/64 bit version)

For Javascript development:

- Linux (RedHat Enterprise Edition 6, Mint 15)
- MacOSX (10.7.5 and 10.8.4)
- Windows (XP/7)

3 Installing the Qeo SDK

How to Install

To install the Qeo SDK it suffices to unzip QeoSdk.zip to a folder on your file system. This results in a ***QeoSDK*** folder which will be referred to as ***<QEO_HOME>*** hereafter. Download the SDK here:

http://wiki.developer-partnersalliance.com/display/QeoDevSpace/SDK+Download+Page

Previous SDK Versions

Refer to Migration Guide - Android or Migration Guide - C for more details on how to migrate to a newer version of the Qeo SDK. Since the previous release did not include JavaScript support, there is no issue for the JavaScript SDK.

⛔ The SDK will automatically export to a different folder than previous version's folders. However, to avoid confusion, it is good practice to remove all SDK files from a previous installation and version of the SDK.

4 Exploring the Qeo SDK

Folder Overview

The Qeo SDK contains all required resources to get you started on Qeo Development for Android or C. After installation of the Qeo SDK, you will find the following folders in your <***QEO_HOME***> folder:

- android
  - apks
  - docs
  - lib
  - samples
- C
  - bin
  - docs
  - include
  - lib
  - platform
  - samples
- js
  - docs
  - lib
- qdms
- tools

android/apks Folder

This folder contains the Qeo Service for Android that needs to be installed once on every Android device to make it Qeo-enabled. Refer to the Android Developer Guide for more information. On top of that, it contains:

- The Qeo Android Management Application. This is an add-on to the Qeo Service designed to improve policy redistribution. Refer to Updating Policy Files on Your Devices for more information.
- The Remote Devices Registration Application: This application allow you to register remote devices. Refer to The Remote Device Registration Application.

android/docs Folder

This folder contains Javadocs of the libraries you need for Qeo development on Android.

android/lib Folder

This folder contains all libraries you need for Qeo development on Android. Refer to Using the Qeo SDK for Android for more info.

android/samples Folder

This folder contains a number of Qeo samples for Android, which a developer can use as a reference. These samples are ready to be imported into Eclipse. Refer to Using the Qeo SDK for Android for more info.

c/bin Folder

This folder contains the ***SafeGuard Daemon***. This is an add-on to the Qeo Service designed to improve policy redistribution. Refer to Updating Policy Files on Your Devices for more information.

c/docs Folder

This folder contains a Qeo C API reference manual in HTML format.

c/include Folder

This folder contains all necessary header files for Qeo development in C. Refer to Using the Qeo SDK for C for more info.

c/lib Folder

This folder contains all necessary libraries for Qeo development in C for the supported platforms. Refer to Using the Qeo SDK for C for more info.

c/platform Folder

This folder contains all files related to platform portation and integration. For more information, go to Platform Integration Documents.

c/samples Folder

This folder contains a number of Qeo samples written in C, which a developer can use as a reference. Refer to Using the Qeo SDK for C for more info.

js/docs Folder

This folder contains a Qeo Javascript API reference manual in HTML format.

js/lib Folder

This folder contains the Qeo javascript library (qeo.js). You need to include this file in your own projects to make them Qeo-enabled.

qdms Folder

This folder contains the Qeo Data Model (QDM) files and related files. For more information, refer to the Qeo Data Model (QDM) Developer Guide and QDM Tooling.

tools Folder

This folder contains the Qeo code generator for generating code from a QDM file. For more information, refer to the Code Generator User Guide

5 Using the Qeo SDK for Android

Prerequisites

The following prerequisites must be met:

- An available development machine supported by Eclipse and the Android SDK.
- Basic knowledge of Eclipse and the Android SDK.
- At least one Android device at your disposal, running Android version 2.3 or higher.
- To fully leverage the Qeo SDK, we use Eclipse as IDE.
- In order to do Qeo development for Android, you first need to install the ***Android SDK***. More info on how you install the Android SDK can be found on the Android SDK page from the Android developers documentation.

If you do not have an IDE installed yet, we recommend installing the ADT Bundle for your platform. This single download contains everything you need for Android development (Eclipse, ADT plugin as well as Android platform, tools and system image for the emulator...). After installing this bundle, you will find the Android SDK (tools, platform-tools,...) installed a directory which will be referred to as <Android_HOME> hereafter.

Install the Qeo Samples

The Qeo SDK contains Android sample projects that work out of the box. As with any Android project, you can now install and run your Qeo samples on your device.

To do so, do the following:

1. In Eclipse's File > Import menu, select General > Existing Projects into Workspace.
2. Make sure Select root directory is checked.
3. Click Browse and navigate to the ***android/samples*** directory in <QEO_HOME>.
4. It is good practice to make sure the Copy projects into workspace option is checked. Eclipse will create a copy of all sample files in your workspace. This way the original files in the Qeo SDK are not modified.
5. Click Finish.

The samples will appear in the Eclipse ***Project/Package Explorer view***.

If Eclipse is not set to automatically build after import, manually start a build after importing the samples so that all sample projects are properly and successfully compiled. Depending on the Eclipse workspace settings, you may need to refresh and rebuild the sample project. Since a full description of the workspace settings is outside the scope of this document, we recommend performing this refresh and build step regardless of your settings.

Android target

All provided samples are built using a recent Android SDK. In case Eclipse generates errors such as "Unable to resolve target 'android-18'", this means that version is not installed on your development machine. This can be solved by either:

Installing that target version (eg 'android-18') on your machine using the "Android SDK Manager"

Modifying the sample target version in the project properties: see <android><Project Build Target>

Create your Own Qeo Project

You can create or use your own Android project (ref. Android documentation) and make it Qeo-enabled by proceeding as follows:

1. Copy the libs from ***<QEO_HOME>/android/lib*** into your ***<your_eclipse_project>/libs*** folder.
2. Perform a refresh in Eclipse (Optional)
3. This allows you to start using Qeo libraries in your project.

Run Your Qeo Project

As with any Android project, you can deploy, run and debug your Qeo project using common Eclipse and Android SDK tools.

Restrictions

⛔ Qeo Development is not (yet) supported on the standard Android Emulator.

6 Using the Qeo SDK for C

Prerequisites

The following prerequisites must be met:

- The Developer must have a x86 development machine running Linux.
- The Developer must have a toolchain for the target platform on which to run the Qeo application(s).
- The ***$QEO_HOME*** environment variable should point to the Qeo SDK installation location.

Install the Qeo Samples

The Qeo SDK contains C sample projects that work out of the box.

To run any of the samples, do the following:

1 To keep the original sample files untouched. You might want to copy the ***c/samples/sample-q...*** directory to another location before proceeding.

> This step is optional.

2 Go into the sample directory.

3 By default the sample will be built for a Linux x86 platform.

> Precompiled libraries for host are built for 32bit linux. If running on a 64bit machine, you'll need 32bit compatibility packages installed.

4 Type 'make'. This will result in an executable.

5 You can now run the executable. Make sure that the Qeo library can be found by setting ***LD_LIBRARY_PATH*** to the correct location ***(${QEO_HOME}/c/lib/<platform>***, where ***<platform>*** is one of the platforms listed above).

6 To compile the samples for another platform (see Supported Platforms for a list), proceed as follows:

7 Set the environment variable ***$PLATFORM*** to one of the supported platforms (e.g. "arm-linux-gnueabihf")

8 Set the environment variable ***$CROSSCOMPILER*** to the base of the compiler toolchain (e.g. "/home/myuser/Downloads/gcc-linaro-arm-linux-gnueabihf-raspbian/bin/arm-linux-gnueabihf-")

9 type the following command:

```
make -f Makefile.crosscompile
```

Create your Own Qeo Project

You can create or use your own C project and make it Qeo-enabled by proceeding as follows:

1 In your source files add ***#include <qeo/api.h>*** and the necessary code to use Qeo (see Making Your C Application Qeo Enabled).

2 In your ***Makefile*** update:

- the compiler flags by adding ***-I${QEO_HOME}/c/include***;
- the linker flags by adding ***-L${QEO_HOME}/c/lib/$(PLATFORM) -lqeo***.

3 Rebuild and run.

7 Using the Qeo SDK for JavaScript/HTML5

7.1 About the JavaScript Binding

The Qeo-Javascript language binding is not tied to a single implementation platform. Rather, the Qeo SDK offers you the necessary building blocks to build Qeo-enabled Javascript applications that run inside an Android WebView component or that run as native Linux applications backed by a similar component (e.g. QWebView or WebKitGTK+).

⚠ A custom qeo-enabled webview is required for the Qeo Javascript binding, Qeo Javascript can't be used in a regular browser (e.g chrome, firefox,...)

7.2 Android

Javascript-enabled Qeo Android applications do not differ significantly from any other Qeo Android application. See Using the Qeo SDK for Android to get up and running with Qeo Android development in general.

QWebview is a sample Android application which presents you an HTML5/Javascript implementation of the QGauge and QSimpleChat sample applications.

7.3 Native

As was the case with Android, a Javascript-enabled native Linux application is in many respects just a native Linux application, subject to the same constraints and possibilities. See Using the Qeo SDK for C to get started with native Linux Qeo applications.

As an example, we have provided a sample QWebView integration of the Qeo/Javascript language bindings. This section explains the necessary prerequisites and steps you have to take to build and run this sample integration.

Prerequisites

The following prerequisites must be met:

- The Developer must have a x86/x64 or ARM development machine running Linux.
- The Developer must have the Qt application framework installed (at least version 5.0) for the target platform on which to run the Qeo Qt application(s).
  The Qt application framework can be downloaded from http://qt-project.org/downloads.
  To build the Qeo Qt applications you may need to install some additional packages on your target platform.
  E.g. debian/ubuntu: **libgl1-mesa-dev** and **build-essential**, for Fedora**: mesa-libGL** and **"Development Tools".**
- The ***$QEO_HOME*** environment variable should point to the Qeo SDK installation location.

⚠ The Qeo SDK is currently only delivered as 32-bit binaries. In order to use these binaries with Qt, you will need to install the 32-bit version of the Qt application framework.

For x64 machines you will need additinally the ia32-libs to install the 32-bit Qt.

The QWebview Sample Application

The QWebview application is a common container for loading Qeo-enabled web pages. A web page (HTML file) is loaded as a command line parameter. This gives you the advantage that you don't need to recompile the Qt application each time you make changes to one of your web related files (HTML, CSS or Javascript).

The Qt project file for the Qt appliclation can be found in the Qeo SDK under c/samples/sample-qwebview-qt.
The Qeo sample web pages can be found in the Qeo SDK under c/samples/sample-qwebview-qt/html.

> ⚠ The Qt Web application has the Qt Web Inspector integrated into the code. You can activate it by compiling the application in debug mode. The Qt Web Inspector is a powerful debugger tool that allows you to debug and trace your web related files (HTML, CSS and Javascript).

Building and Running QWebview

There are two ways to build the Linux QWebview application:

- Via the command line
- Using the QtCreator IDE that comes with the Qt application framework
- Command-line build instructions

Compile via Command Line

1. Open a terminal and go to the directory c/samples/sample-qwebview-qt of the Qeo SDK on your machine.
2. Make sure that the Qeo library can be found: export *LD_LIBRARY_PATH* = *${QEO_HOME}/c/lib/i686-linux:$LD_LIBRARY_PATH*
3. type qmake
4. type make (or make DEFINES+="DDEBUG" if you want use the Qt Web Inspector)

> ⚠ If you don't have the gcc/bin directory of the Qt application framework in your PATH directory then you need to add that directory to the qmake command.
>
> If you want to recompile the Qt application, you can clean it by executing make clean first and then run qmake and make again.

Run via Command Line

1. To run the Qt application, you just start it up with a web page location specified on the command line.
For example:

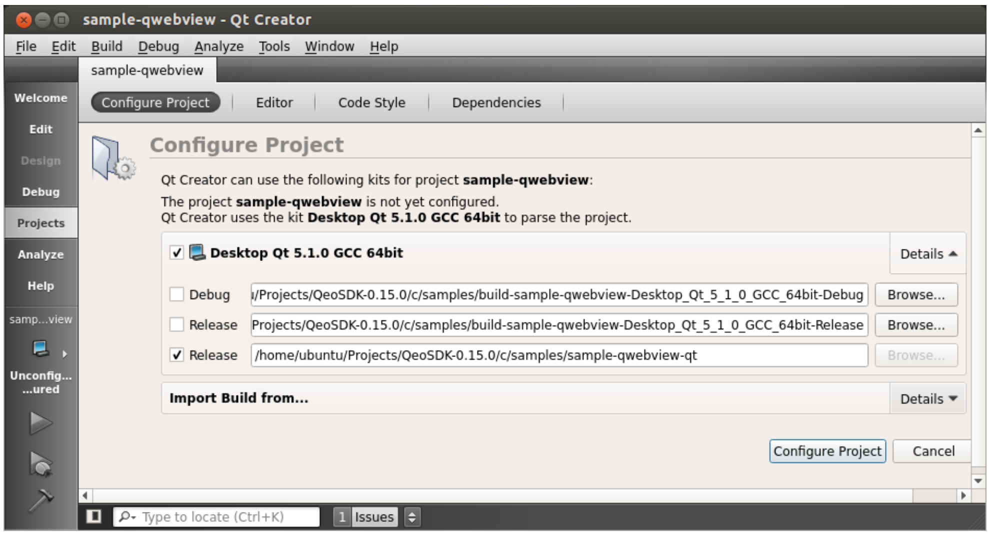
```
./sample-qwebview html/QSimplechat.html
```

Using QtCreator

You can also use the QtCreator IDE that comes with the Qt application framework.

1 Start QtCreator and open the Qt project file **sample-qwebview.pro** located in the directory **c/samples/sample-qwebview-qt** of the Qeo SDK on your machine.

2 Click on the **Configure Project** button.



Configure Project

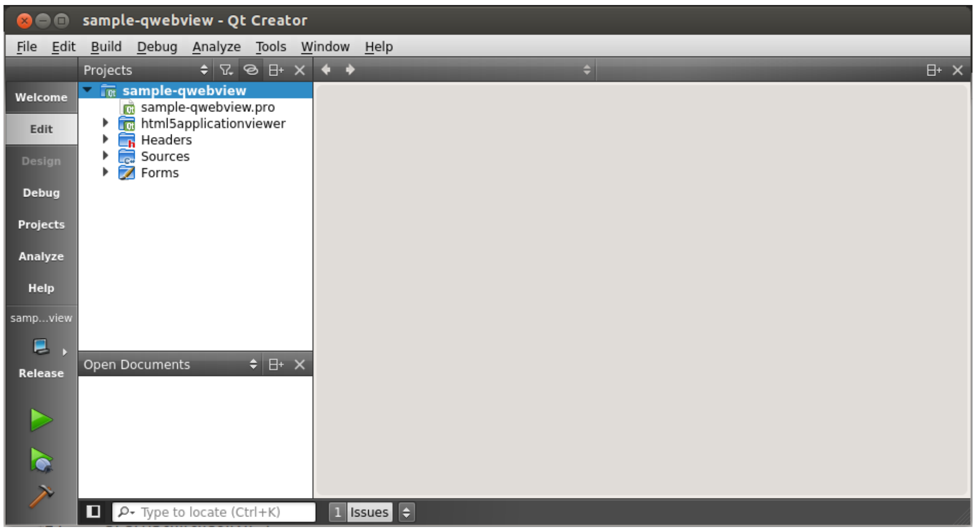
Qt Creator can use the following kits for project **sample-qwebview**:

The project **sample-qwebview** is not yet configured.
Qt Creator uses the kit **Desktop Qt 5.1.0 GCC 64bit** to parse the project.

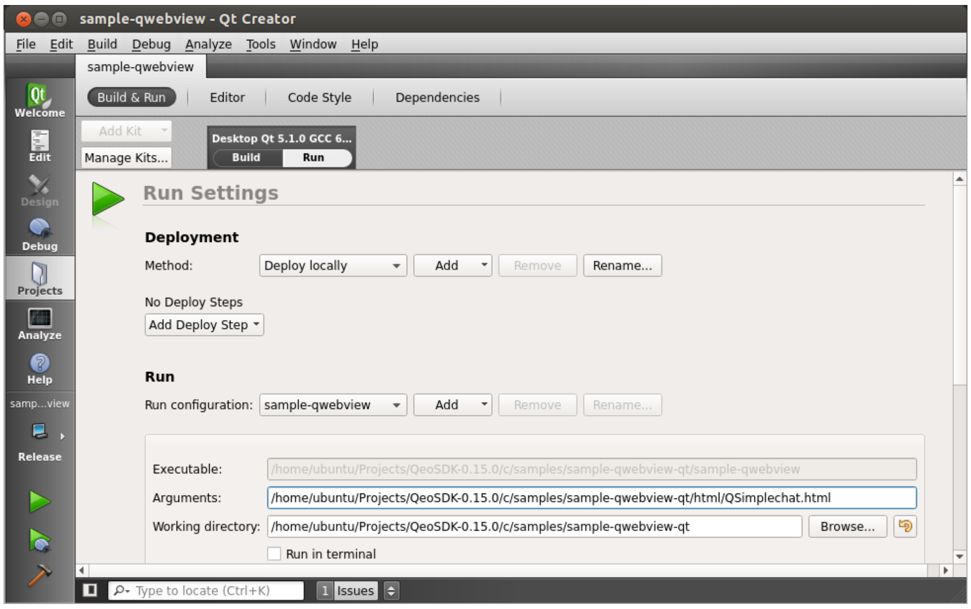
✓ **Desktop Qt 5.1.0 GCC 64bit**

☐	Debug	ı/Projects/QeoSDK-0.15.0/c/samples/build-sample-qwebview-Desktop_Qt_5_1_0_GCC_64bit-Debug
☐	Release	Projects/QeoSDK-0.15.0/c/samples/build-sample-qwebview-Desktop_Qt_5_1_0_GCC_64bit-Release
✓	Release	/home/ubuntu/Projects/QeoSDK-0.15.0/c/samples/sample-qwebview-qt

Qt IDE:



Adding a web page to the Run Settings:
Go to the Projects tab on the left side, than select **Build&Run** tab (top side) and then the **Run** (toolbar below the Build&Run) to get to the Run Settings.
Fill in the **Arguments** box with the directory path to one of the Qeo web samples.



Run Settings

Deployment

Method: Deploy locally

No Deploy Steps

Run

Run configuration: sample-qwebview

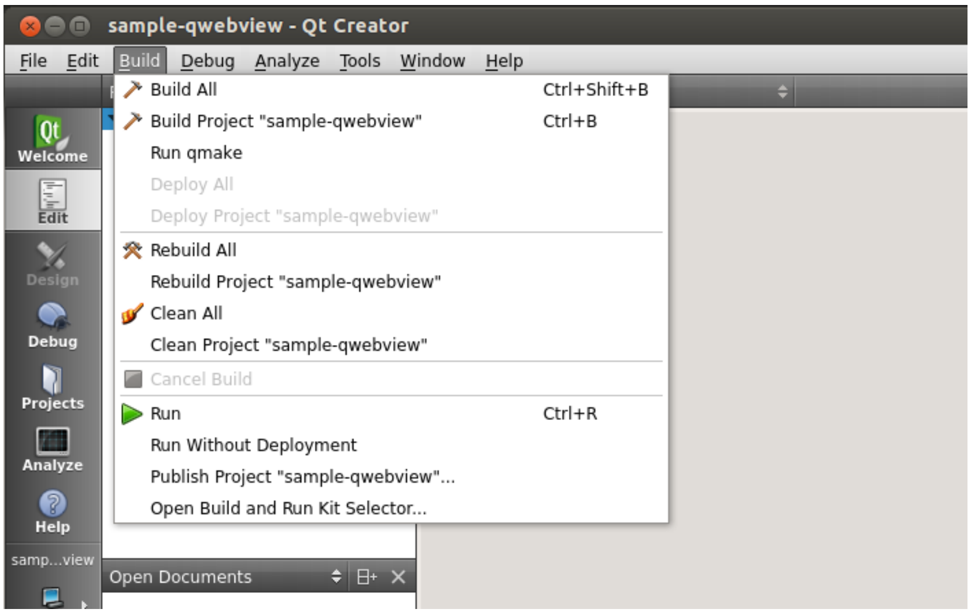
Executable: /home/ubuntu/Projects/QeoSDK-0.15.0/c/samples/sample-qwebview-qt/sample-qwebview

Arguments: /home/ubuntu/Projects/QeoSDK-0.15.0/c/samples/sample-qwebview-qt/html/QSimplechat.html

Working directory: /home/ubuntu/Projects/QeoSDK-0.15.0/c/samples/sample-qwebview-qt

☐ Run in terminal

Build an Run:
Go to the Build menu and select **Run qmake** and once again to select **Run**.


sample-qwebview - Qt Creator
File Edit Build Debug Analyze Tools Window Help
Build All Ctrl+Shift+B
Build Project "sample-qwebview" Ctrl+B
Run qmake
Deploy All
Deploy Project "sample-qwebview"
Rebuild All
Rebuild Project "sample-qwebview"
Clean All
Clean Project "sample-qwebview"
Cancel Build
Run Ctrl+R
Run Without Deployment
Publish Project "sample-qwebview"...
Open Build and Run Kit Selector...
Welcome
Edit
Design
Debug
Projects
Analyze
Help
samp...view
Open Documents

⚠ You can switch between Release and Debug compilations using the 4th last tab on the left side.

If your gcc/bin of the QT application framework is not in your PATH directory, then you can modify the build settings to point to the location of the Qt qmake binary file.

8 Migration Guide - Android

About this document

This document provides all the information needed to migrate your Qeo development environment and devices to a newer version of the Qeo SDK for Android.

Migration steps

To migrate to the latest Qeo SDK, proceed as follows:

1. Install the latest Qeo SDK.
2. Upgrade your Android devices.
3. Upgrade your Android applications to use the latest Qeo libraries.

Install the latest Qeo SDK

Download the latest SDK and unzip it to a folder on your filesystem. After unzipping, you will find a folder like "QeoSDK-0.x.y ", this folder will be referred to as ***<QEO_HOME>***. See Installing the Qeo SDK for more details.

Upgrade your devices

To upgrade your Android devices, uninstall the existing Qeo Service and install the latest version of the Qeo Service.

Don't upgrade Qeo Service

Do not upgrade the existing Qeo Service to the latest version, as this will result in communication issues with the Security Management Server.

Upgrade your Android Applications

You can upgrade each of your Qeo enabled Android Applications as follows:

In your Eclipse workspace, delete all old Qeo libraries from the ***<your_eclipse_project>/libs*** folder.

- qeo-android-0.14.0-SNAPSHOT.jar
- qeo-android-core-0.14.0-SNAPSHOT.jar
- qeo-java-core-0.14.0-SNAPSHOT.jar

Import all new Qeo libraries from the ***<QEO_HOME>/android/lib*** folder into your ***<your_eclipse_project>/libs*** folder.

- qeo-android-0.15.0-SNAPSHOT.jar
- qeo-android-core-0.15.0-SNAPSHOT.jar
- qeo-java-core-0.15.0-SNAPSHOT.jar

To upgrade a sample that was delivered with the previous version of the SDK, it is recommended to remove it completely from your Eclipse environment and import the updated version included in the latest SDK.

9 Migration Guide - C

About this document

This document provides all information needed to migrate your Qeo development environment to the latest version of the Qeo SDK for C.

Migration steps

To migrate to the latest Qeo SDK, proceed as follows:

1. Install the latest Qeo SDK.
2. Clean your C devices.
3. Compile your C applications against the latest Qeo libraries.
4. Update your C applications to make them compatible with the latest Qeo API.

Install latest Qeo SDK

Download the latest SDK and unzip it to ***<NEW_QEO_HOME>*** as described in the Installation Guide.

Clean your devices

Remove the existing credentials of your C devices. By default they are stored in the ***.qeo*** directory of your home directory:

```
rm -rf ~/.qeo/*
```

Don't use old credentials

Don't reuse existing credentials, as this will result in communication issues with the Security Management Server.

Compile against latest Qeo libraries

If you are using ***Makefiles*** similar to those delivered with the samples, you can link to latest version of the SDK by setting the ***$QEO_HOME*** to ***<NEW_QEO_HOME>***.

If you are using a different flavor of ***Makefiles***, or a whole different build system, you need to change the library path to ***<NEW_QEO_HOME>/c/lib/<PLATFORM>***.

Make sure to clean any object files to make sure your program is recompiled against the latest version.

Update to the latest Qeo API

Some changes in the Qeo API are not fully backwards compatible. Please read the list of API changes to get a full list of changes you might need to apply to your code.

END OF DOCUMENT

Revised: 3 October 2013. Printed: 3 October 2013.