Archived:Mobile Extensions

Archived: This article is **archived** because it is not considered relevant for third-party developers creating commercial solutions today. If you think this article is still relevant, let us know by adding the template {{ReviewForRemovalFromArchive|user=~~~~|*write your reason here*}}.

Mobile Extensions are generally not available, and where available not supported. Use Qt Mobility APIs instead.

Qt Mobility replaces Mobility Extensions API

15 Mar 2009

You can extend your Qt applications on S60 with the Mobile Extensions. **The Mobile Extensions were a** technology preview **and have been replaced by the Qt Mobility project at Qt**. Newest version of Mobility is 1.1, see f.ex http://qt.nokia.com/products/qt-addons/mobility/

Qt Mobility is also included in the (Nokia) Qt SDK available in http://www.developer.nokia.com/Develop/Qt/Tools/. Notice the name change between Nokia Qt SDK 1.0 and Qt SDK 1.1 (that contains both desktop and mobile development environments, covering Symbian and Maemo).

Even though mobile extensions are officially deprecated we still want to hear your valuable feedback. If you have any questions, issues or improvement suggestions please utilize our discussion board: http://www.developer.nokia.com/Community/Discussion/forumdisplay.php?196-Qt-for-Symbian It is actively monitored by the team behind the Extensions and Qt Mobility.

Mobile Extensions

The delivery package

The Mobile Extensions are delivered as source code as well as binaries. This package contains nineteen APIs and example code illustrating the use of each one. There are also five demonstration applications. The demonstration applications are available by installing the launcher application included in the delivery package. The demonstrations and examples are built against Qt 4.6.3: Symbian source and binary zip

Tip: The latest Qt SDK is for Qt 4.72; you may need to rebuild the source for this SDK

- **\README.txt**: Brief notes about the Mobile Extensions
- **\doc**: Documentation in HTML format (unzip the zip file to view it). The API documentation is also available as part of standard eclipse
- **\install_to_device**: Pre-built extension libraries and demos and examples for installing to a device.

Note that this does *not* include the Qt binaries. They need to be installed separately. See Qt SDK

Obsolete Release notes for Mobility Extensions

2.7.2009 - Third Technology Preview of the Mobile Extensions:

- New API features:
  - New API added for setting device alarms: Alarms API.
  - New API added for accessing the calendar on the device: Calendar API
  - Messaging API now supports emails
  - Settings Manager API now includes several new classes for more control.
- Simpler installation: The libraries do not need to be built since the binaries are now included
- Simpler usage:
  - "system include" directory containing all public headers added
  - Only LIBS += xq* is needed in .pro files
- Carbide templates removed

Qt must be installed on your S60 device so that you can run applications. Qt must be installed in the SDK(s) that you will be

developing with.

Note: Some of the APIs require restricted capabilities which means your application installation files need to be Symbian Signed. See Signing an application in the Qt Developer's library for more information about this.

1. Unzip the Mobile Extensions package (qt_s60_mobile_extensions.zip) into a new directory. There are no installers to run.
2. Install the development package and emulator libraries by unzipping the qt_mobile_ext_libs_epoc32.zip package into the EPOCROOT directory. This package contains pre-built extension binaries and libraries as well as the SDK plugins. The EPOCROOT directory is the same directory that contains the epoc32 directory, for example C:\S60\devices\S60_5th_Edition_SDK_v1.0\.
3. The examples and demos can now be built and the extensions can be used in your own applications.

26.5.2009 - Qt Mobility Project announced:

In February we published the first versions of Mobile Extensions for Qt that provide the capability to extend your Qt applications to utilize mobile features on S60 devices.

We've gotten great feedback from our developers and now we're happy to announce that the first set of these APIs are being evolved for cross-platform use, not just S60. The API interfaces and use cases of mobile extensions have been refined based on the valuable feedback we've gotten. The project is called the **Mobility project** (http://labs.qt.nokia.com/category/qtmobility/) and the work is starting with Bearer Management API and Contacts API.

In addition there will be other enablers for Qt developers that can be used on mobile devices like the announced Service Framework (http://labs.qt.nokia.com/2009/05/26/qt-service-framework/). In the longer run we're aiming to get more cross-platform solutions from the mobile extensions. It is recommended to always use the cross-platform Qt APIs when available.

9.4.2009 - Second Technology Preview of the Mobile Extensions:

- Improved documentation. In addition to the Eclipse plugin all the documentation is available as HTML. The .chm document file is removed.
- Some fixes in the APIs.
- One installer package for the demonstration applications that is compatible with Qt release S60 3rd Edition and S60 5th Edition binaries.
- The installer package checks whether your device supports S60 sensor plugin or S60 sensor framework and based on that installs the correct version of the Bricks application.
- The demonstration applications usability has been improved.
- Simplified examples.
- Example UIs done with Qt designer.

Getting started

To test the demonstration applications, see the demonstration section of this page. All that's needed is an S60 3rd Edition, Feature Pack 1 (FP1) or later device. If you are unsure which S60 Edition device you have, check the Nokia Developer device specifications page. Note that as Qt is a technology preview the usability and look and feel of the applications may be imperfect.

The precondition for developing with these extensions is that [[Qt SDK]] must be installed. More information regarding Qt development is also available here.

Should you have any questions please utilise the Qt discussion board.

Useful resources

- Qt Developer's library
- Nokia Developer Qt discussion board
- Qt Developers for Symbian at Qt
- Information about S60 devices

Want to try the demonstration applications?

In addition to the simple example applications provided, there are five more advanced demonstration applications featuring some of the possibilities that the Mobile Extensions provide.

- Flickr demonstration: The demonstration uses your location to fetch pictures taken near you.
- Camera & MMS demonstration: The demonstration captures an image using the onboard camera. The picture is sent to a selected contact (as a multimedia messaging service, MMS) and added as your image, which is shown once you call the contact (in vcard terminology called the 'logo', so the demonstration could be called the 'remote logo setter').
- Contact Photo demonstration: The demonstration adds an image to a contact that is shown once the contact calls you (in vcard terminology called the 'logo').
- Bricks demonstration: A game that demonstrates how sensors can be used as input controls.
- Landmarks demonstration: Lists landmarks around the current position (10 km radius). Landmarks are also added to the database of the mobile device. Demonstration also offers a chance to make phone call if a number can be parsed from the name of the landmark.

Documentation

The API documentation in HTML format in 'http://doc.qt.nokia.com/qtmobility-1.0/index.html. *After unzipping the package open index.html to view the documentation in your browser.*

Obsolete Developing with mobile extensions - use Qt Mobility

Mobile Extensions are located under the directory .*extensions**src*\. Simple example applications demonstrating the APIs are located in directory .*extensions**examples*\. There are also more sophisticated demonstration applications that use several extensions in the .*extensions**demos*\.

To use an extension you need to add the appropriate Extension library to your project file and add the include path to the Extensions header files. Then, in your code, you need to include the appropriate header file for the Extension you want to use. The header file always has the same name as the main Extension class. In many cases you will also need to add some security information to your project file since the Extensions use restricted features of the device.

For example, if you want to use the Access Point Manager API, the Access Point Manager API implementation notes show that you need to add the following lines to your .pro file:

```
INCLUDEPATH += [my relative extensions path]\extensions\include

symbian:LIBS += -lxqaccesspointmanager

symbian:TARGET.CAPABILITY = ReadUserData
```

Then you need to include the header for the Access Point Manager API class in your application source. The main class is XQAccessPointManager so the name of the corresponding header file is xqaccesspointmanager.h.

```
#include <xqaccesspointmanager.h>
```

Note about platform security

Starting with S60 3rd Edition Platform Security applies all C/C++ applications on S60 devices.

The same capability and certification rules apply to Qt applications as for Symbian C++ applications. Some of the extensions require either a developer certificate or the use of Symbian online signing. For more information about platform security, see Symbian OS Platform Security.

User-grantable capabilities are **LocalServices**, **ReadUserData**, **WriteUserData**, **NetworkServices**, and **UserEnvironment** (and **Location** beginning with S60 3rd Edition, Feature Pack 2). The APIs that use only user grantable capabilities only (APIs that you can utilise without having an application Symbian Signed) are shown in the API listing below. If your application uses these APIs, the application can be self signed and installed to a "Qt-enabled" S60 mobile device.

Building the examples

The easiest way to get familiar with the extensions is to play around with the examples. You can start using the examples simply by importing the examples .pro file in to Carbide.c++ (instructions available in the Qt developer's library). If you just want to compile the complete examples, you can do so also from the command prompt:

Building the Contacts example for the emulator

```
cd extensions\examples\contactsex
qmake contactsex.pro
make debug-winscw
```

Running the example on the emulator

```
make run
```

Building and installing to a target device

```
make release-gcce
createpackage -i contactsex_gcce_urel.pkg
```

This will create a self-signed *contactsex_gcce_urel.sis* and install it to your device assuming you have Nokia PC Suite installed and a compatible device connected to your PC.

You can also specify your own certificate and key files as additional parameters to *createpackage* if the default ones are not suitable, for example when self signing is not adequate due to capability requirements.

Note that if you want to compile the flickr or landmarks demonstration applications remember to set your Flickr API key to the following files:

- .\demos\flickrdemo\Flickrdemo.cpp
- .\demos\landmarks\Landmarks.cpp

APIs in the current release

The Mobile Extensions package contains pre-built versions of the Mobile Extensions libraries. You can install them with the .\install_to_device\qt_mobile_ext_libs.sisx file. NOTE: the Qt 4.5.2-Tower binaries must be installed in your device before you install the Mobile Extensions libraries.

The source code for the APIs is located in the .*extensions**src*\ folder. There is no need to build them yourself since pre-built binaries are supplied but, if do you want to build them yourself, they must be built with the Qt 4.5.2-Tower release - older Qt releases are not supported.

Examples using the APIs are in the .\extensions\examples\ folder. Prebuilt versions of them are also included in the Mobile

Extensions package in the .\install_to_device\demos_and_examples\ folder.

For detailed API documentation, refer to the documentation in the delivery packages .\doc\ folder.

1. Access Point Manager API

- Main class `XQAccessPointManager`
- Main use cases: Listing Internet Access Points (IAPs), setting the used access point, listing available access points, scanning for available WLANs

2. Alarms API

- Main class `XQAlarms`
- Main use cases: Listing alarms, creating alarms, deleting alarms.

3. Audio API

- Main class `XQAudioRecord`
- Main use cases: Recording audio

4. Calendar API

- Main class `XQCalendar`
- Main use cases: Listing appointments (and todos, anniversaries, reminders and events), creating appointments, adding and removing alarms, deleting appointments.

5. Camera API

- Main class `XQCamera`
- Main use cases: Using device's onboard camera for taking photos. Also supports viewfinder, i.e., continues camera view.

6. Contacts API

- Main class `XQContacts`
- Main use cases: Access contacts database to find, read, modify, and add contacts

7. Installer API

- Main class `XQInstaller`
- Main use case: Provides the ability to install an S60 application from the installer package; can be used, for example, to update your application

8. Landmarks API

- Main class `XQLandmarksManager`
- Main use cases: Listing available landmarks and adding a landmark

9. Location API

- Main class `XQLocation`
- Main use case: Reading devices GPS location information

10. Media API

- Main class `XQMedia`
- Main use cases: Retrieving lists of music, image, video, and sound files located in the gallery

11. Messaging API

- Main class `XQMessaging`
- Main use cases: Sending and receiving SMS and MMS messages

12. Profile API

- Main class `XQProfile`
- Main use cases: Reading devices profile information and setting the active profile

13. Resource Access API

- Main class `XQResourceAccess`
- Main use case: Accessing Symbian resource files. Eases porting of S60 Symbian C++ applications because existing resource files can be used in Qt applications.

14. Sensor API

- Main classes `XQAccelerationSensor` and `XQDeviceOrientation`
- Main use cases: Provides devices sensor information about acceleration and orientation

15. Settings Manager API

- Main class `XQSettingsManager`
- Main use cases: Accessing Central Repository and Publish & Subscribe items for getting notifications about changes in system settings.

16. System Information API

- Main class `XQSysInfo`
- Main use cases: Accessing system information, such as used language, battery information, etc.

17. Telephony API

- Main class `XQTelephony`
- Main use cases: Making circuit switched calls and getting call status notifications

18. Vibra API

- Main class `XQVibra`
- Main use cases: Turning the device vibra on and off

19. Utils API

- Main class `XQUtils`
- Main use cases: Launching a file in a default viewer application and resetting the system timer to keep the backlights on

Demo applications

In addition to providing the APIs and simple examples, we offer fancier demonstration applications that show some of the possibilities that the Mobile Extensions provide. Note that because both Qt and the Mobile Extensions are in the technology preview phase, there may be certain issues with the demos. We welcome your feedback on the discussion board.

Install the following packages:

- Install the Qt SDK
- Go through this article: Getting started with Qt
- Install the MobileExtDemos application (.*install_to_device**qt_mobile_ext_demos.sisx*) that contains all the demonstration applications.

Be aware that because the demos use Internet connections (and the Camera MMS demonstration sends an MMS), your operator will charge you for data traffic and messages.

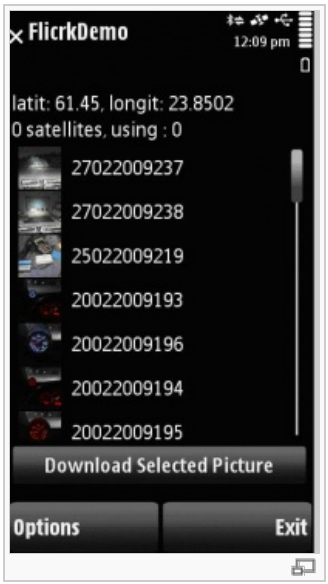
MobileExtDemos application

The MobileExtDemos application can be used to launch the demonstration applications. The following screenshot shows this application running on the Nokia 5800 XpressMusic device:

Flickr demo

The demonstration uses your location to fetch pictures taken near you using the Flickr service.

The following screenshot shows the demonstration running on the Nokia 5800 XpressMusic device:



- Once you've launched the application, give it some time to get the location; when the location has been retrieved, clicking the Download Picture List button will retrieve the pictures taken near you.
  - If the location has not yet been retrieved, hard-coded test coordinates will be used. You will see a prompt notifying you of this.
- After the download starts, the grid will be populated with pictures and thumbnails of the pictures.
- Once you select a picture from the list you can download it by clicking the Download Selected Picture button.
- When the download starts, you will see a progress bar. Once the download has finished, the picture will be shown on another window. Clicking OK will add the picture to the gallery; Delete will remove the picture.

To compile the Flickr demonstration application, remember to set your own API key to the following file:

- \demos\flickrdemo\Flickrdemo.cpp

Camera MMS demo

The demonstration captures an image using the on-board camera. The picture is sent to a selected contact (as an MMS) and added as an image that is shown once you call the contact (in vcard terminology, the 'logo').

Remember to have the application running on the device to which you're about to send your picture (the receiving end)!

- First, take a picture using the Capture button. Once you click the button you will see the picture that's been taken. If you want

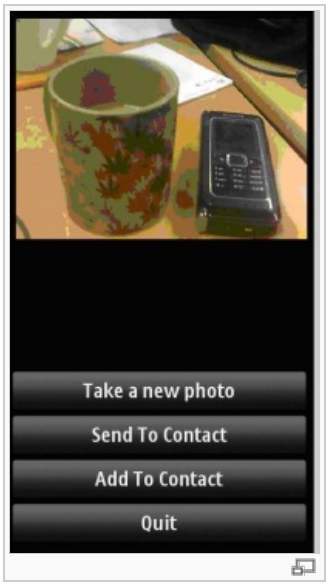
to take a new picture, select Take A New Photo. When you're satisfied, click the Send to Contact button.

- Select a contact from your contacts list by clicking the Send To Contact button.
- Once the contact has been selected, click the Send button. Note that after the 'Picture was successfully sent' dialog it still takes some time for the picture to actually be sent and received on the other device.

Once you hear the message tone, the MMS has arrived at the other mobile device and you will be able to see that the application on the receiving end is showing your contact information and the received picture.

The picture can be added as your logo just by clicking the Add button. You can easily test to see if the demonstration works by calling the receiver. Note that the MMS message is deleted automatically from the receiver end. You don't have to do anything with the application on the receiving end — just have it running.

Screenshot of the demonstration running on the Nokia 5800 XpressMusic device:

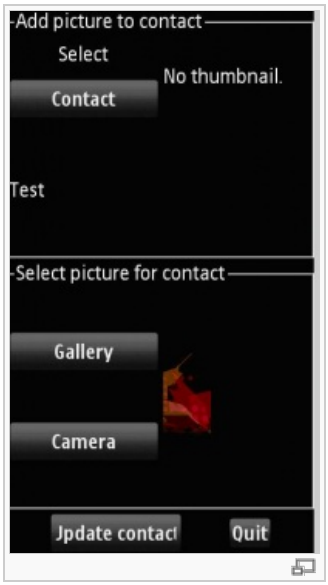


Contact photo demo

The demonstration adds an image to a contact that is shown once the contact calls you (in vcard terminology called the 'logo').

- First, select a contact from your contacts list by clicking the Contact button. Once the contact has been selected, click the OK button.
- Next, select an image from the gallery or take a picture using the Camera button. Once you click the camera button you will see a viewfinder; take a picture by clicking the Capture key. When the picture has been taken, click the OK button.
- Now you've selected the contact and the picture. Click Update Contact and you're done.

When the contact calls you, you will see the picture you set as the caller's logo! Screenshot of the demonstration running on Nokia 5800 XpressMusic:



Bricks demo

The Bricks demonstration is a simple arcade-style game that demonstrates the use of sensors to control the player object.

Screenshot of the demonstration running on the Nokia 5800 XpressMusic device

Landmarks demo

The Landmarks demonstration lists landmarks around the current position (10 km radius). Landmarks are also added to the database of the mobile device. The demonstration offers the opportunity to make a phone call if a number can be parsed from the name of the landmark.

Once the application starts, it will read the GPS location. When the location has been retrieved, the Download Landmarks button will start retrieving landmarks near you. If the location has not yet been retrieved, a hard-coded test location is used.

When the landmarks have been retrieved, they are listed in a listbox. If the landmarks have a phone number included, the number will be displayed in the Call To button and a phone call can be made to the defined phone number.

To compile the landmarks demonstration application remember to set your own API key to the following file:

- \demos\landmarks\Landmarks.cpp

Archive

- The Mobile Extensions first preview package
- The Mobile Extensions second preview package (now replaced by the Third preview package)

Feedback

We welcome your feedback and suggestions. If you have any questions or feedback, please use our discussion board at Qt Discussion board

Have fun with the extensions and stay tuned for updates!