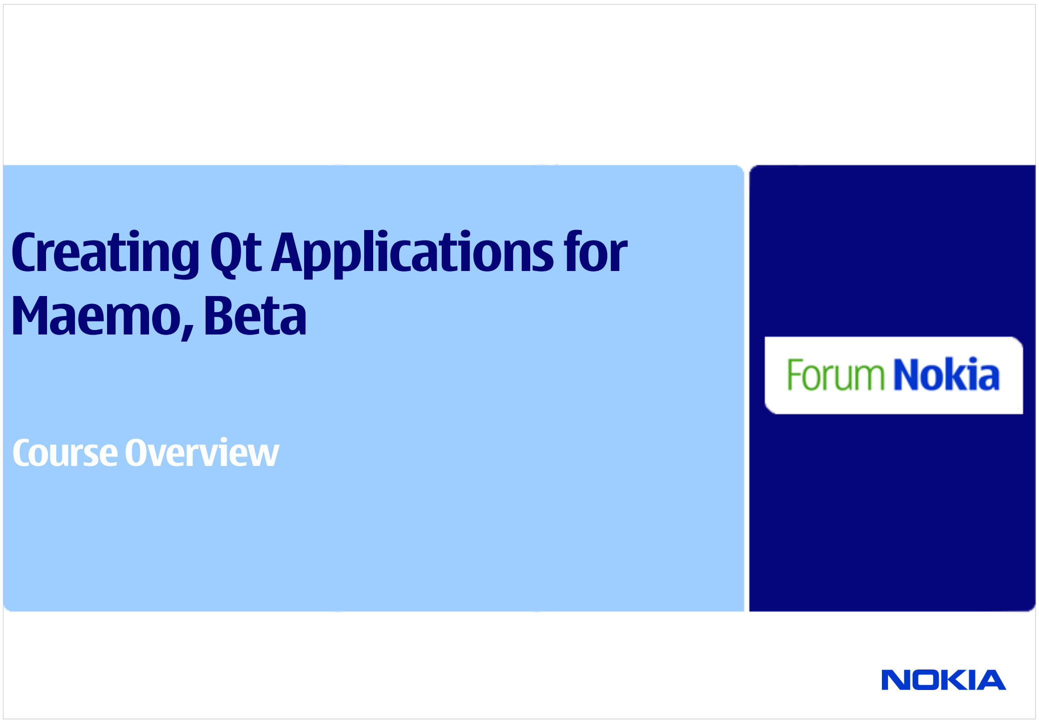
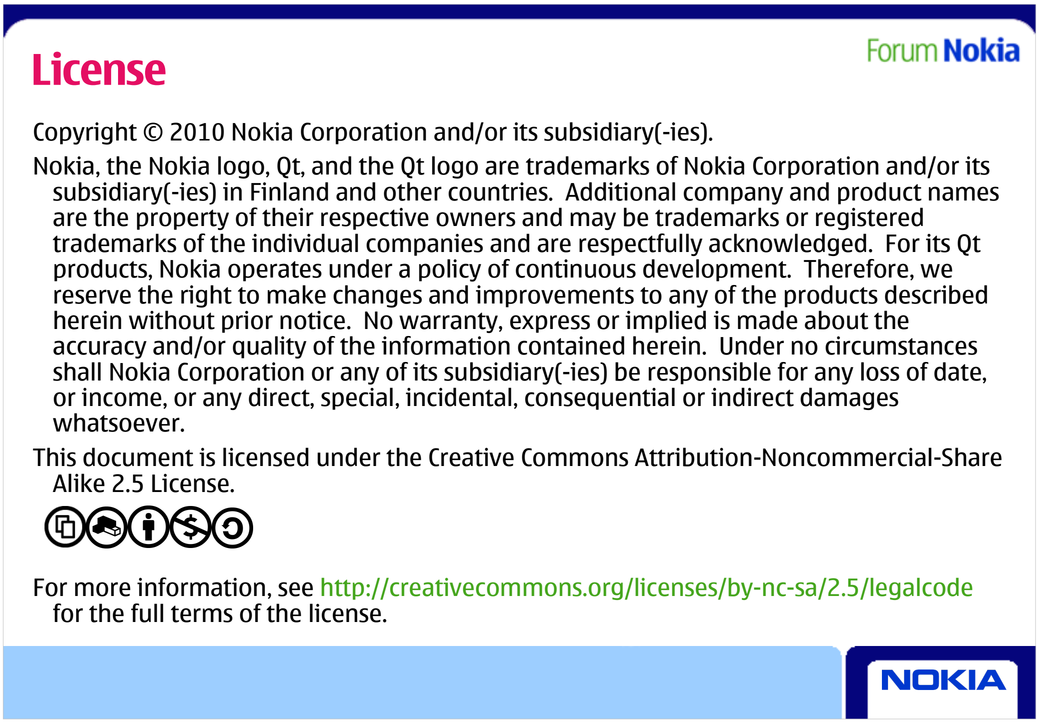


Creating Qt Applications for Maemo, Beta

Course Overview

Forum Nokia

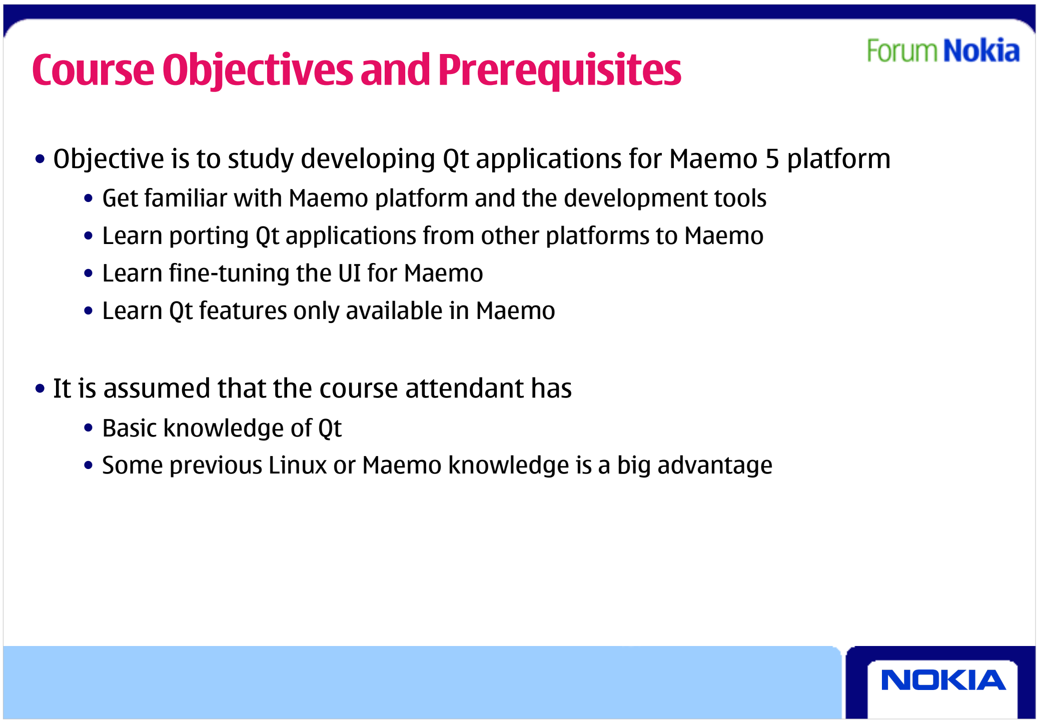
NOKIA



License

Forum Nokia

NOKIA



Course Objectives and Prerequisites

- Objective is to study developing Qt applications for Maemo 5 platform
  - Get familiar with Maemo platform and the development tools
  - Learn porting Qt applications from other platforms to Maemo
  - Learn fine-tuning the UI for Maemo
  - Learn Qt features only available in Maemo
- It is assumed that the course attendant has
  - Basic knowledge of Qt
  - Some previous Linux or Maemo knowledge is a big advantage

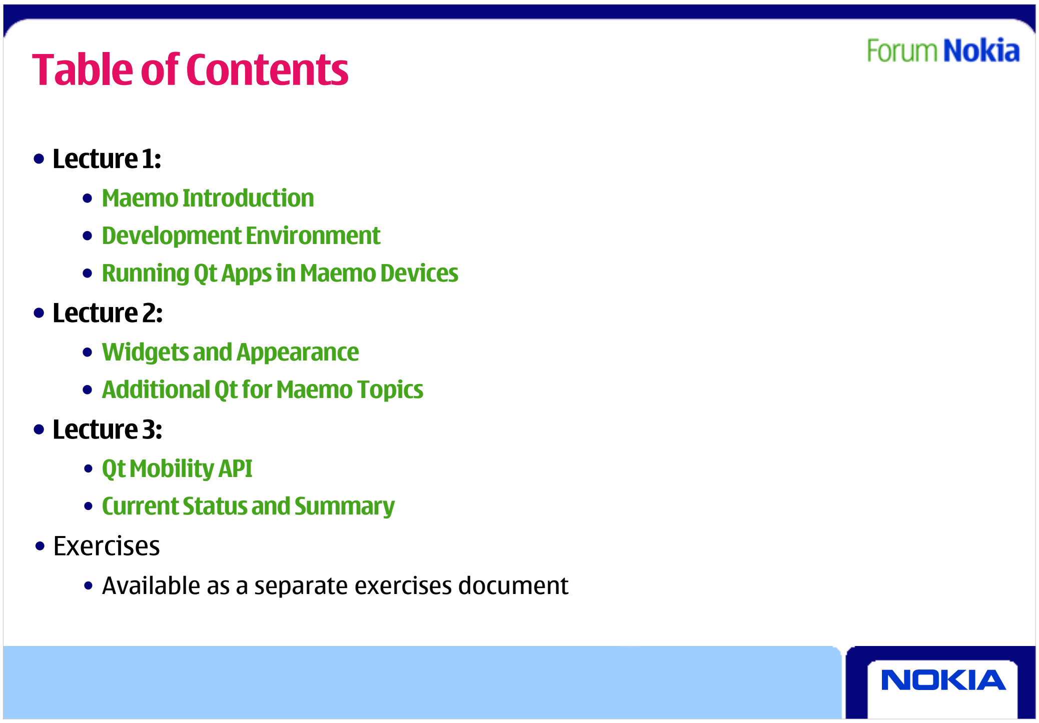
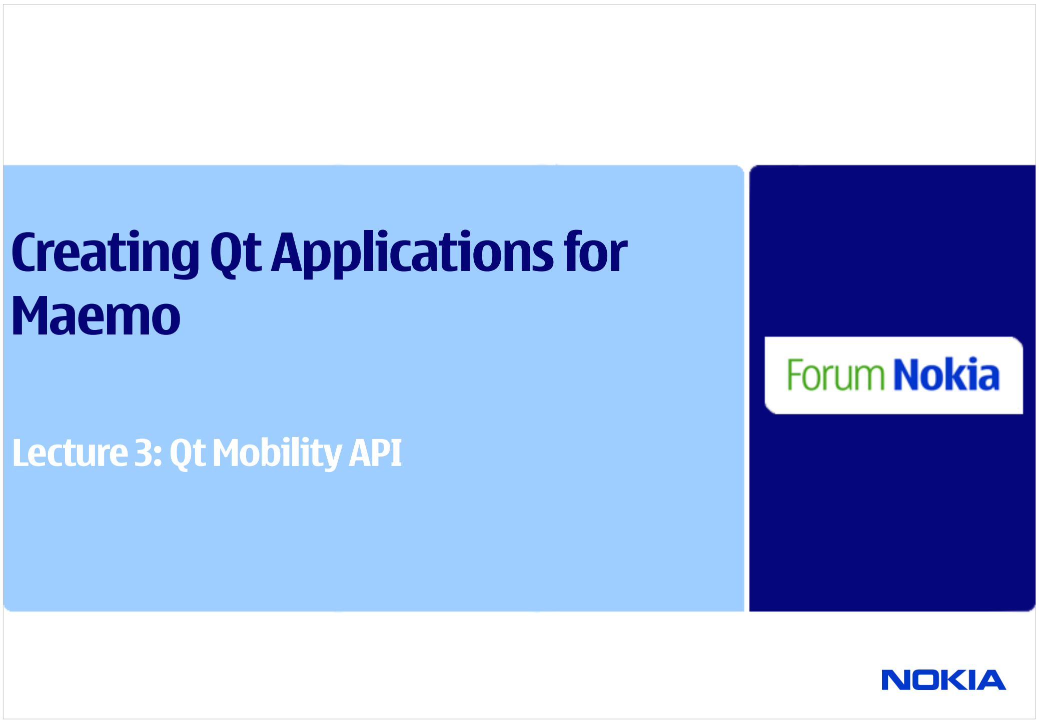


Table of Contents

- **Lecture 1:**
  - **Maemo Introduction**
  - **Development Environment**
  - **Running Qt Apps in Maemo Devices**
- **Lecture 2:**
  - **Widgets and Appearance**
  - **Additional Qt for Maemo Topics**
- **Lecture 3:**
  - **Qt Mobility API**
  - **Current Status and Summary**
- Exercises
  - Available as a separate exercises document

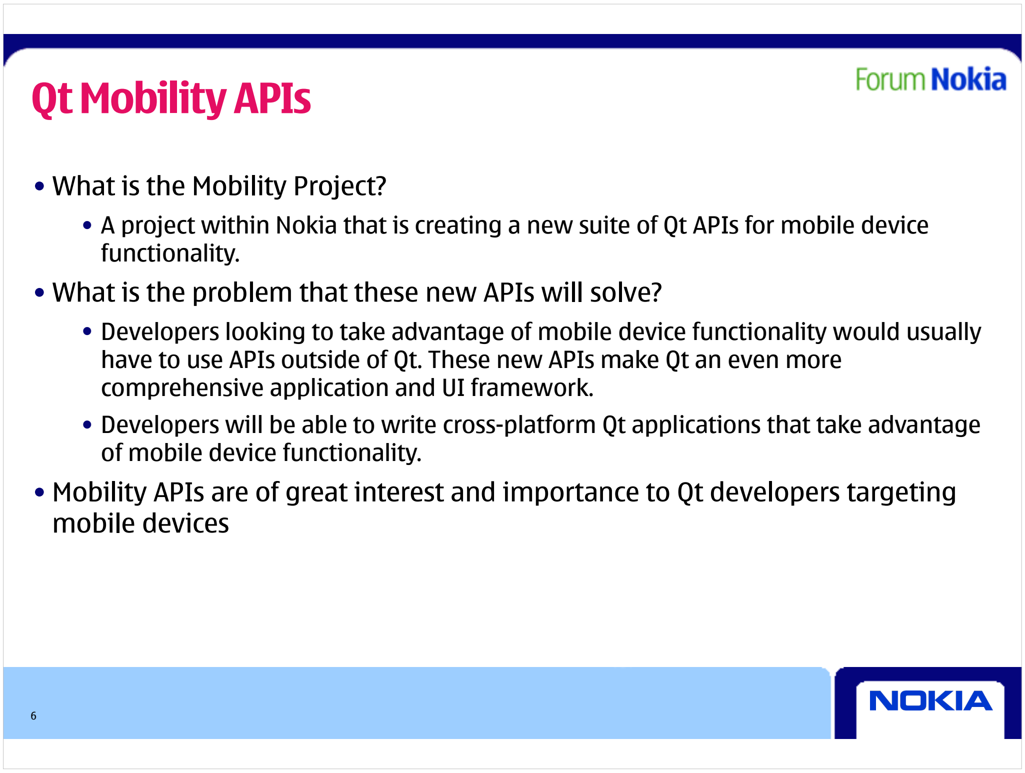


Creating Qt Applications for Maemo

Lecture 3: Qt Mobility API

Forum Nokia

NOKIA



Qt Mobility APIs

Forum Nokia

- What is the Mobility Project?
  - A project within Nokia that is creating a new suite of Qt APIs for mobile device functionality.
- What is the problem that these new APIs will solve?
  - Developers looking to take advantage of mobile device functionality would usually have to use APIs outside of Qt. These new APIs make Qt an even more comprehensive application and UI framework.
  - Developers will be able to write cross-platform Qt applications that take advantage of mobile device functionality.
- Mobility APIs are of great interest and importance to Qt developers targeting mobile devices

Installing Qt Mobility Libraries on a Device

- The Qt Mobility APIs are published in the Nokia Application software repository.
- This means that when installing a package, which depends on some Qt Mobility API, also the needed binary packages are installed.
- You can make sure that the Qt Mobility APIs are installed by installing qt-mobility-examples package from Extras-devel:
  - `sudo gainroot`
  - `apt-get update`
  - `apt-get install qt-mobility-examples`

Bearer Management API

Forum Nokia

- Provides way to control which connection is used
- For example, the program can choose to use either 3G or WLAN in different situations
- With this API, also a smooth switch from one bearer to another is possible

```
QNetworkConfigurationManager manager;
const bool canStartIAP = (manager.capabilities() &
QNetworkConfigurationManager::BearerManagement);
QNetworkConfiguration cfg = manager.defaultConfiguration();
if (!cfg.isValid() || !canStartIAP)
    return;
switch(cfg.type()) {
    case QNetworkConfiguration::UserChoice: {
        // automatic IAP resolution by system or by asking user as
        // part of QNetworkSession::open() call
        QNetworkSession *session = new QNetworkSession(cfg);
        session->open(); break; }
    ...
}
```

NOKIA

Contacts API

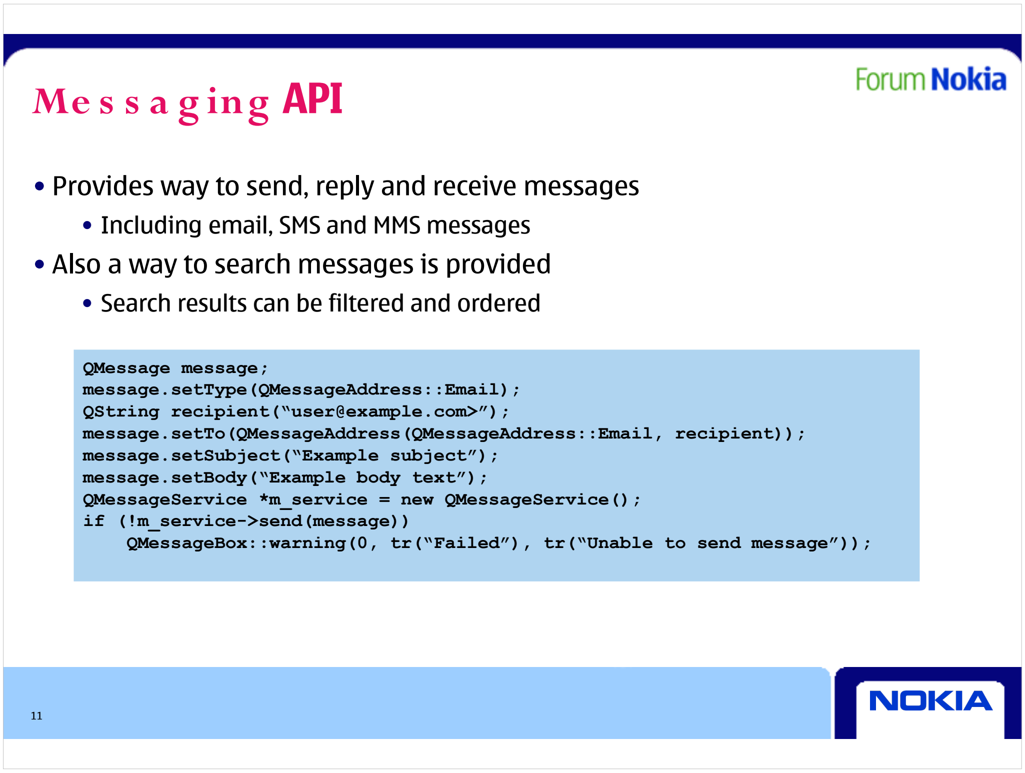
- Provides way to
  - Add, remove and update contacts and contact groups
  - Search for contacts and contacts groups

```
// cm is a QContactManager object
QContact alice;
QContactName aliceName;
aliceName.setFirst(“Alice”);
aliceName.setLast(“Jones”);
alice.saveDetail(&aliceName);
/* Add a phone number */
QContactPhoneNumber number;
number.setContexts(QContactDetail::ContextHome);
number.setSubTypes(QContactPhoneNumber::SubTypeMobile);
number.setNumber(“12345678”);
alice.saveDetail(&number);
alice.setPreferredDetail(”DialAction”, number);
cm->saveContact(&alice);
```

Location API

- Provides easy way to access location information
- Location data can be fetched with the following ways
  - one-time fetch
  - periodically
  - on position change
- QGeoAreaMonitor provides also a way to obtain notification when a specified region is entered or exited

```
QGeoPositionInfoSource *source =
QGeoPositionInfoSource::createDefaultSource();
if (source) {
    connect(source, SIGNAL(positionUpdated(QGeoPositionInfo)),
            this, SLOT(positionUpdated(QGeoPositionInfo)));
    source->startUpdates();
}
// slot positionUpdated is now called each time the position is updated
```



Messaging API

- Provides way to send, reply and receive messages
  - Including email, SMS and MMS messages
- Also a way to search messages is provided
  - Search results can be filtered and ordered

```
QMessage message;
message.setType(QMessageAddress::Email);
QString recipient(“user@example.com>”);
message.setTo(QMessageAddress(QMessageAddress::Email, recipient));
message.setSubject(“Example subject”);
message.setBody(“Example body text”);
QMessageService *m_service = new QMessageService();
if (!m_service->send(message))
    QMessageBox::warning(0, tr(“Failed”), tr(“Unable to send message”));
```

Multimedia API

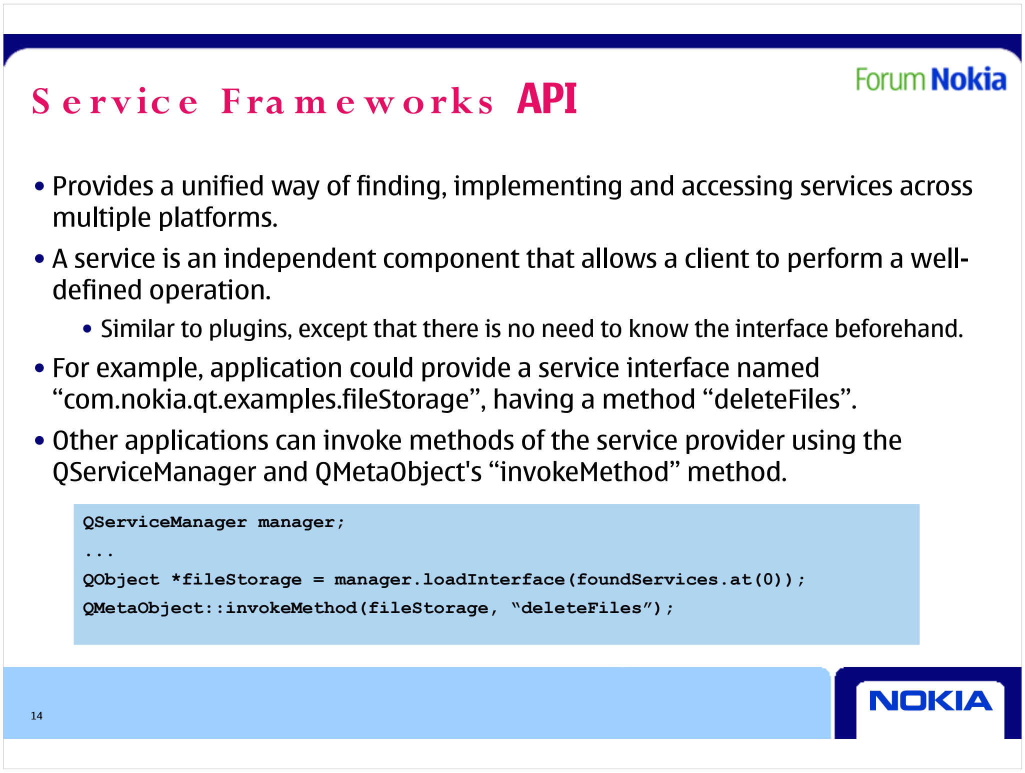
- Provides way to
  - show pictures and play music, video files and multimedia stream
  - record to record sound
  - listen to the radio

```
QMediaPlayer *player = new QMediaPlayer;
...
player->setMedia(QUrl::fromLocalFile(”test.raw”));
player->setVolume(50);
player->play();
```

Publish and Subscribe API

- Provides way to publish items, read item values, navigate through and subscribe to change notifications.
- Data is stored as a tree-like data structure. When a child of a node is changed, also the node itself is changed. This allows applications to be notified always when any child is changed for some node.
- Values are fetched with a path string, for example,
  - /Device/Network/Interfaces/eth0/Name
- In Maemo, the backend is provided with GConf, the GNOME Configuration database system.

```
QString valueSpace = “/Device/Network/Interfaces/eth0/Name”;
QValueSpaceSubscriber *subscriber = new QValueSpaceSubscriber(valueSpace);
QObject::connect(subscriber, SIGNAL(contentsChanged()),
this, SLOT(infoChanged()));
```



Service Frameworks API

- Provides a unified way of finding, implementing and accessing services across multiple platforms.
- A service is an independent component that allows a client to perform a well-defined operation.
  - Similar to plugins, except that there is no need to know the interface beforehand.
- For example, application could provide a service interface named “com.nokia.qt.examples.fileStorage”, having a method “deleteFiles”.
- Other applications can invoke methods of the service provider using the QServiceManager and QMetaObject's “invokeMethod” method.

```
QServiceManager manager;
...
QObject *fileStorage = manager.loadInterface(foundServices.at(0));
QMetaObject::invokeMethod(fileStorage, “deleteFiles”);
```

To use the interface object, you could also cast it dynamically with the qobject_cast. For example,

```
QServiceManager manager;
QServiceFilter filter(”com.nokia.qt.examples.FileStorage”);
filter.setServiceName(”FileManager”);
QList<QServiceInterfaceDescriptor> foundServices;
foundServices = manager.findInterfaces(filter);
QObject *object;
object = manager.loadInterface(foundServices.at(0));
MyFileStorage *fileStorage = qobject_cast<MyFileStorage *>(fileStorage);
if (fileStorage != 0) {
  fileStorage->deleteFiles();
}
```

The downside is that in this case you need to know the specific class of the returned object already when compiling.

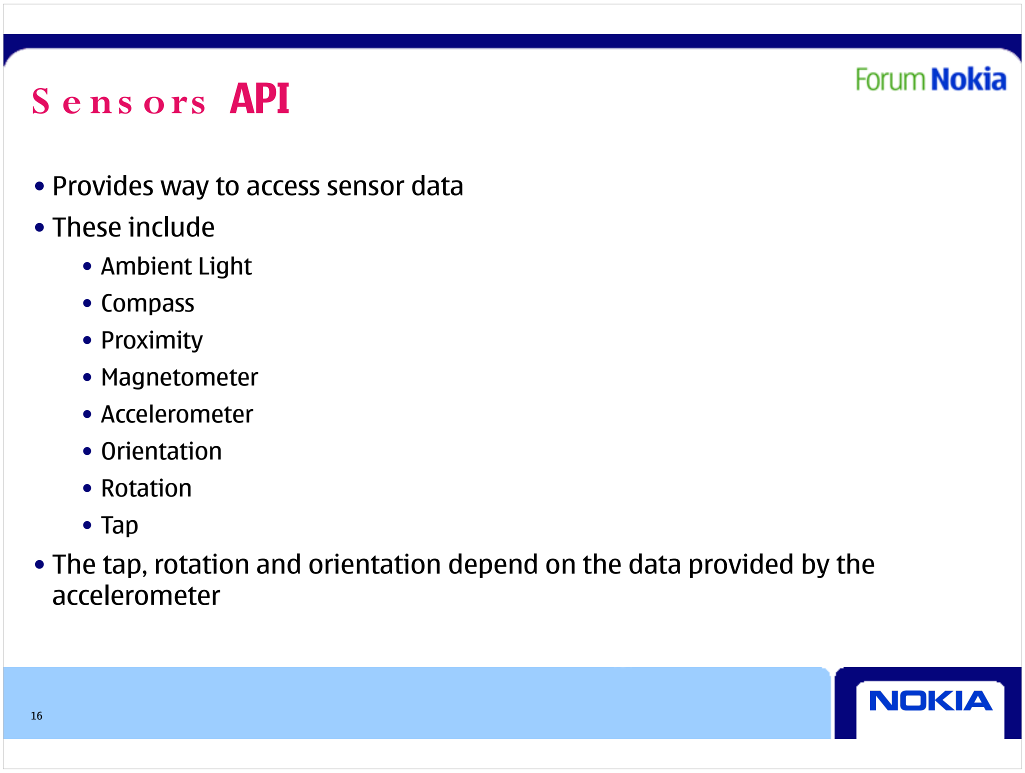
System Information API

Forum Nokia

- Provides way to obtain information about the features and versions
    - Versions of Operating system, firmware, WebKit, Qt
    - Device information like battery level, screen size, SIM, input methods
    - Display information
    - Features such as GPS, bluetooth, wlan, camera, fm radio etc

```
// Battery level (range 1 - 100)
QSystemDeviceInfo di;
di.batteryLevel();

// returns Qt lib version
QSystemInfo systemInfo;
QString qtVersion = systemInfo.version(QSystemInfo::QtCore);
```



Sensors API

- Provides way to access sensor data
- These include
  - Ambient Light
  - Compass
  - Proximity
  - Magnetometer
  - Accelerometer
  - Orientation
  - Rotation
  - Tap
- The tap, rotation and orientation depend on the data provided by the accelerometer

16

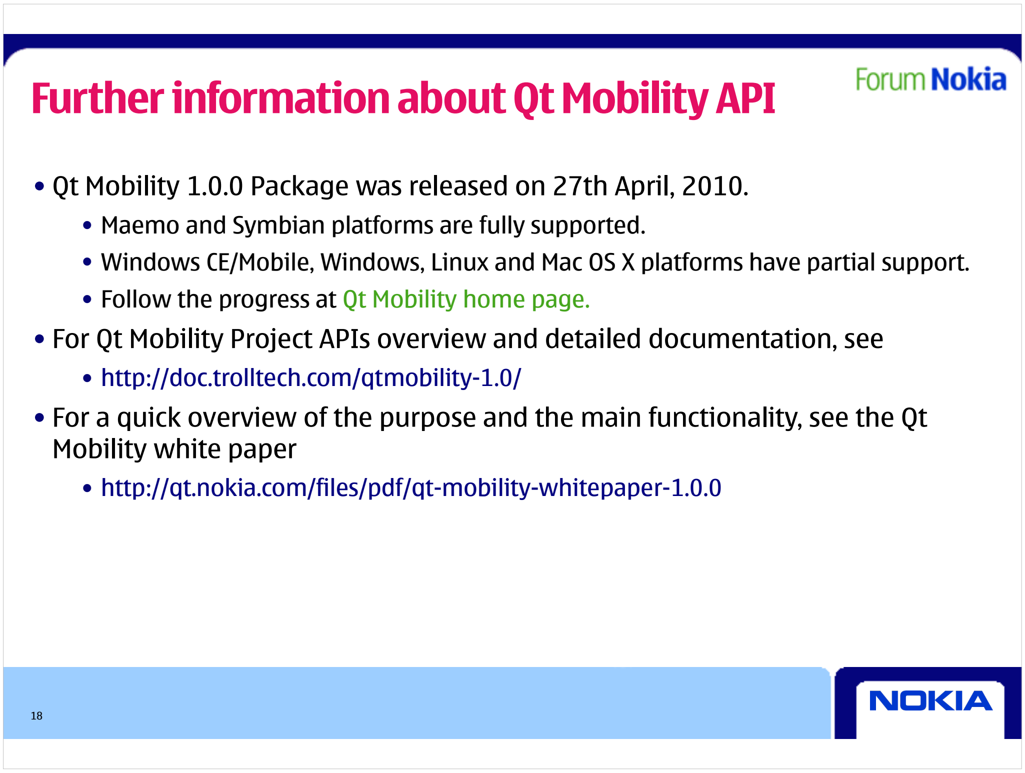
For example, to get the acceleration to the right of the device:

```
QAccelerometer accSensor;
QAccelerometerReading reading = accSensor.reading();
// x is to the right of the device
qreal x = reading.x();
```

Versit© API

- Provides way to read and write Versit© documents such as vCards and vCalendar
- Allows applications to easily map QContacts to vCards and vice versa
- Currently vCards are supported

```
// Parse the input into QVersitDocuments
QVersitReader reader;
reader.setDevice(&input);
reader.startReading(); // Remember to check the return value
reader.waitForFinished();
// Convert the QVersitDocuments to QContacts
QList<QVersitDocument> inputDocuments = reader.results();
QVersitContactImporter importer;
if (!importer.importDocuments(inputDocuments))
    return;
QList<QContact> contacts = importer.contacts();
```



Further information about Qt Mobility API

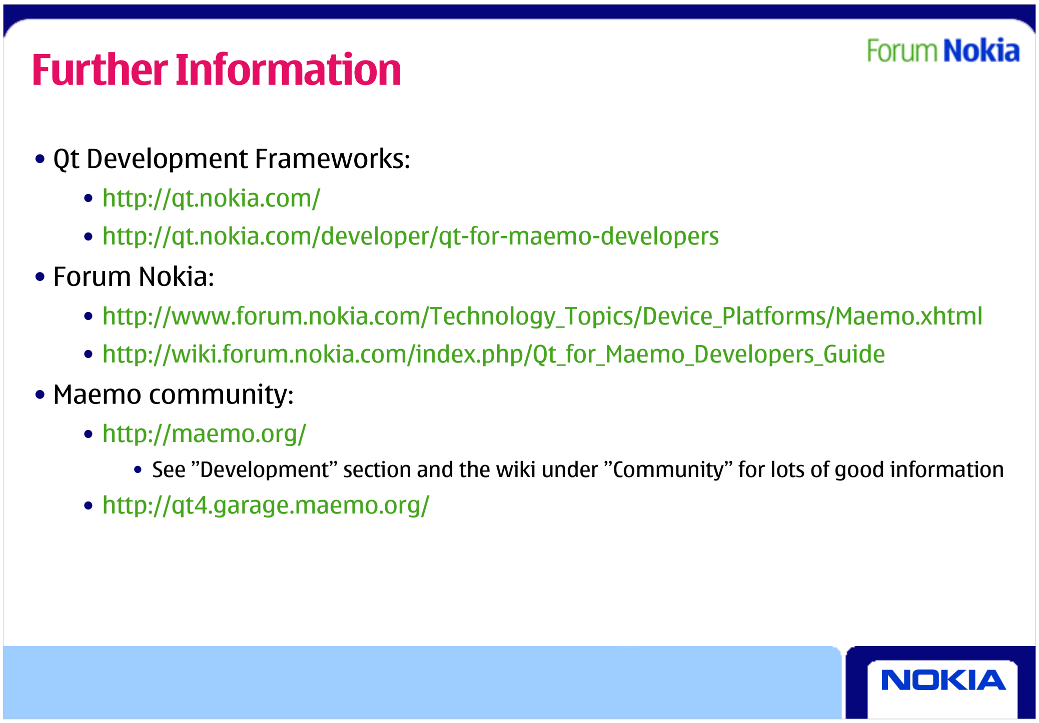
- Qt Mobility 1.0.0 Package was released on 27th April, 2010.
  - Maemo and Symbian platforms are fully supported.
  - Windows CE/Mobile, Windows, Linux and Mac OS X platforms have partial support.
  - Follow the progress at Qt Mobility home page.
- For Qt Mobility Project APIs overview and detailed documentation, see
  - http://doc.trolltech.com/qtmobility-1.0/
- For a quick overview of the purpose and the main functionality, see the Qt Mobility white paper
  - http://qt.nokia.com/files/pdf/qt-mobility-whitepaper-1.0.0

Creating Qt Applications for Maemo

Current Status & Summary

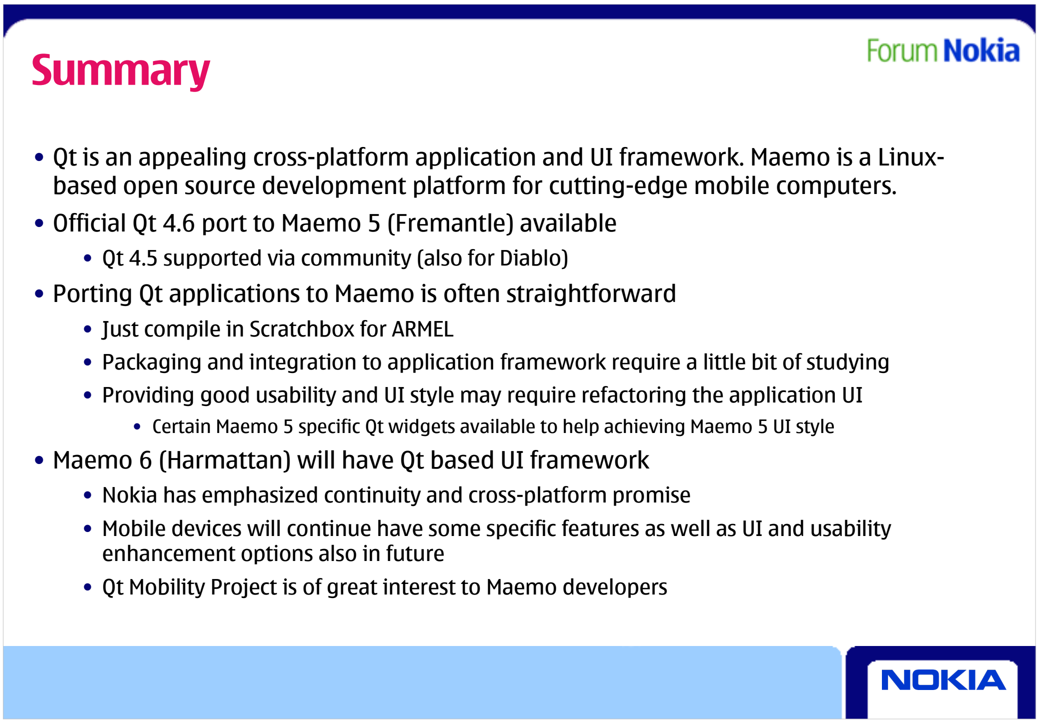
Forum Nokia

NOKIA



Further Information

- Qt Development Frameworks:
    - http://qt.nokia.com/
    - http://qt.nokia.com/developer/qt-for-maemo-developers
- Forum Nokia:
    - http://www.forum.nokia.com/Technology_Topics/Device_Platforms/Maemo.xhtml
    - http://wiki.forum.nokia.com/index.php/Qt_for_Maemo_Developers_Guide
- Maemo community:
    - http://maemo.org/
        - See ”Development” section and the wiki under ”Community” for lots of good information
    - http://qt4.garage.maemo.org/



Summary

- Qt is an appealing cross-platform application and UI framework. Maemo is a Linux-based open source development platform for cutting-edge mobile computers.
- Official Qt 4.6 port to Maemo 5 (Fremantle) available
  - Qt 4.5 supported via community (also for Diablo)
- Porting Qt applications to Maemo is often straightforward
  - Just compile in Scratchbox for ARMEL
  - Packaging and integration to application framework require a little bit of studying
  - Providing good usability and UI style may require refactoring the application UI
    - Certain Maemo 5 specific Qt widgets available to help achieving Maemo 5 UI style
- Maemo 6 (Harmattan) will have Qt based UI framework
  - Nokia has emphasized continuity and cross-platform promise
  - Mobile devices will continue have some specific features as well as UI and usability enhancement options also in future
  - Qt Mobility Project is of great interest to Maemo developers