嵌入式计算机系统

Lecture #10

MeeGo Device and Information Management

内容来自于meego.com及MeeGo相关公开教程

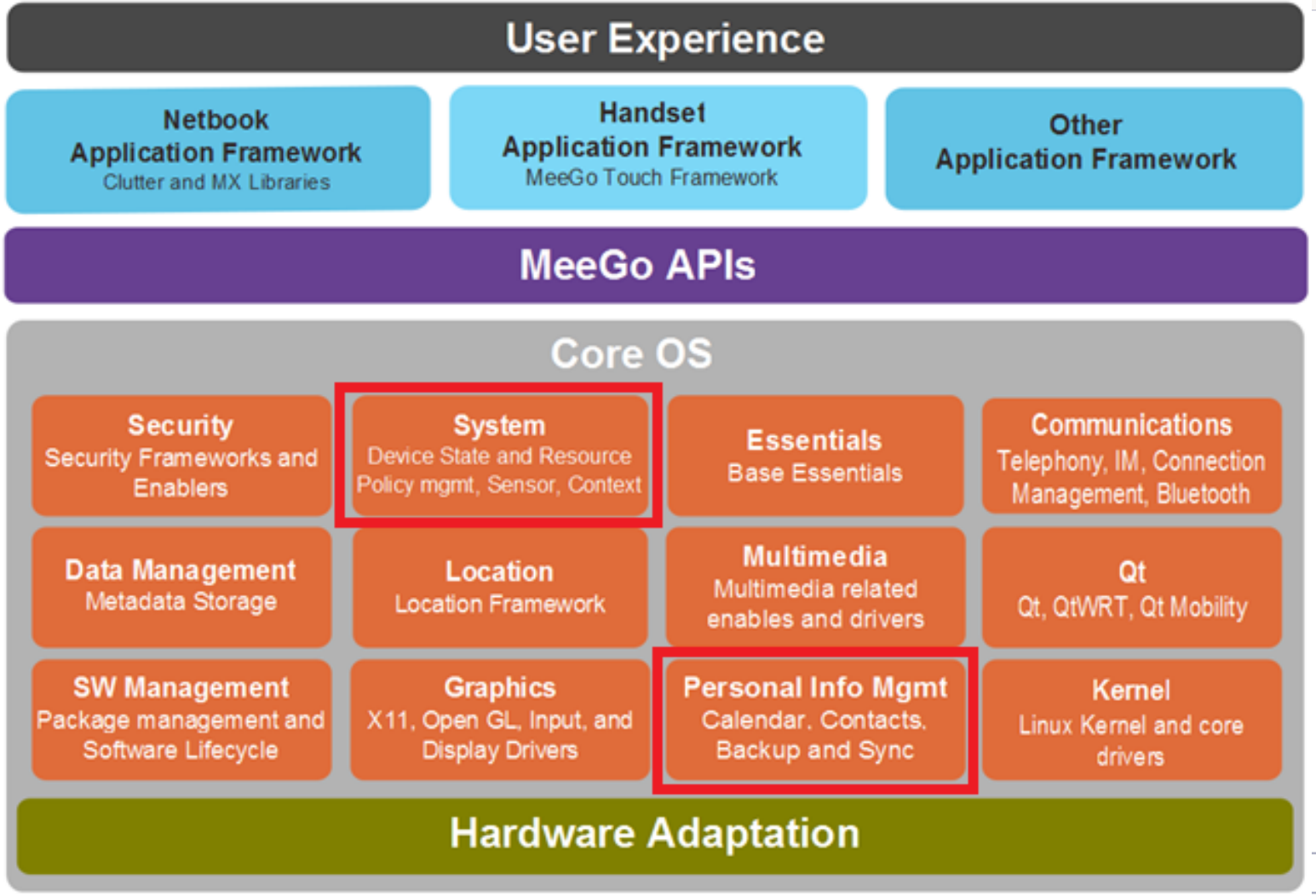
Outline

- MeeGo Device and Information Management Module Overview
- Personal Information Management Module
- System Module

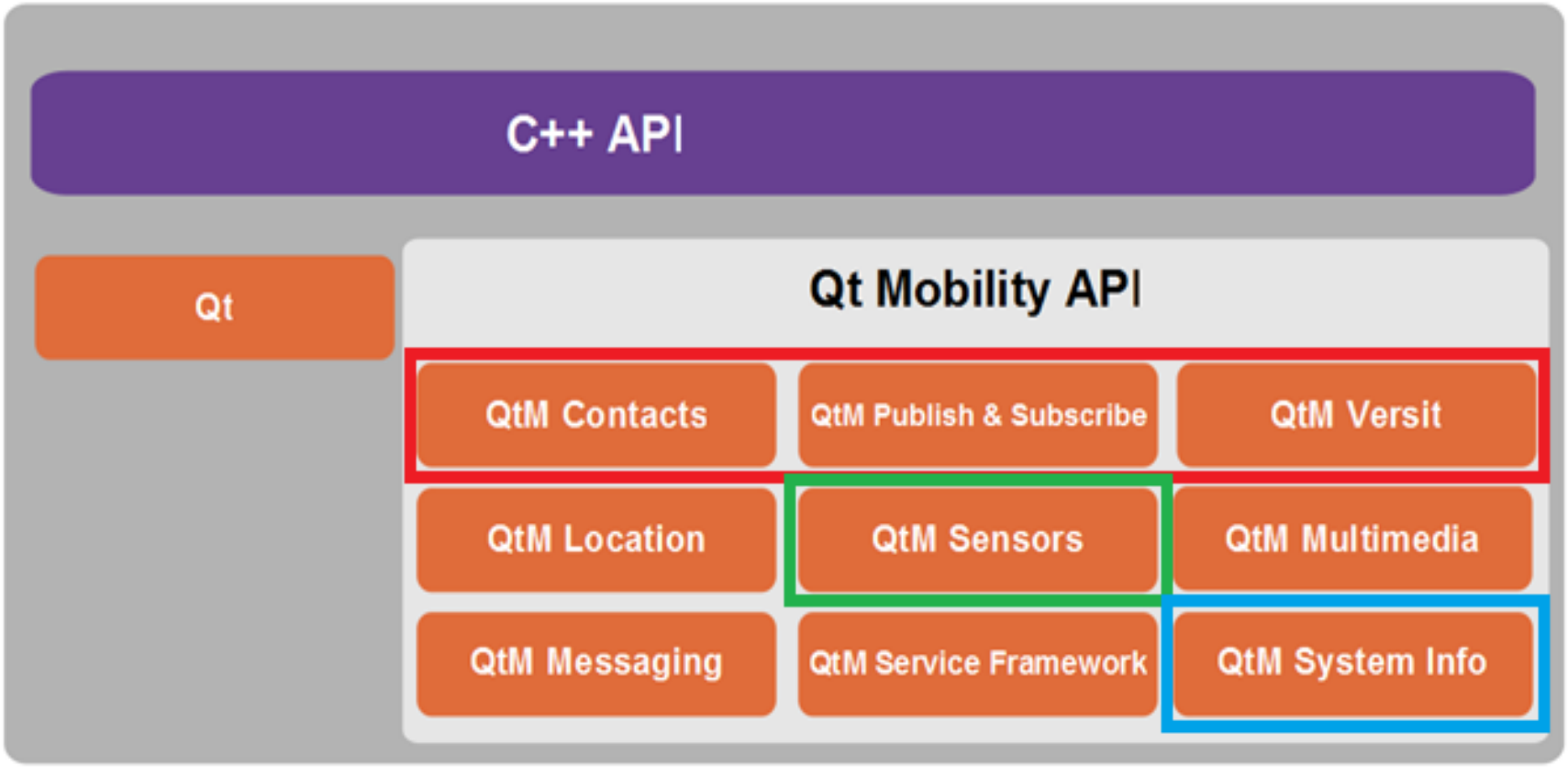
Outline

- MeeGo Device and Information Management Module Overview
- Personal Information Management Module
- System Module

Overview


User Experience
Netbook Application Framework
Clutter and MX Libraries
Handset Application Framework
MeeGo Touch Framework
Other Application Framework
MeeGo APIs
Core OS
Security
Security Frameworks and Enablers
System
Device State and Resource Policy mgmt, Sensor, Context
Essentials
Base Essentials
Communications
Telephony, IM, Connection Management, Bluetooth
Data Management
Metadata Storage
Location
Location Framework
Multimedia
Multimedia related enables and drivers
Qt
Qt, QtWRT, Qt Mobility
SW Management
Package management and Software Lifecycle
Graphics
X11, Open GL, Input, and Display Drivers
Personal Info Mgmt
Calendar, Contacts, Backup and Sync
Kernel
Linux Kernel and core drivers
Hardware Adaptation

Overview


C++ API
Qt
Qt Mobility API
QtM Contacts
QtM Publish & Subscribe
QtM Versit
QtM Location
QtM Sensors
QtM Multimedia
QtM Messaging
QtM Service Framework
QtM System Info

Outline

- MeeGo Device and Information Management Module Overview
- Personal Information Management Module
- System Module

Personal Info Management Module

- Contact
- Versit

Personal Info Management Module

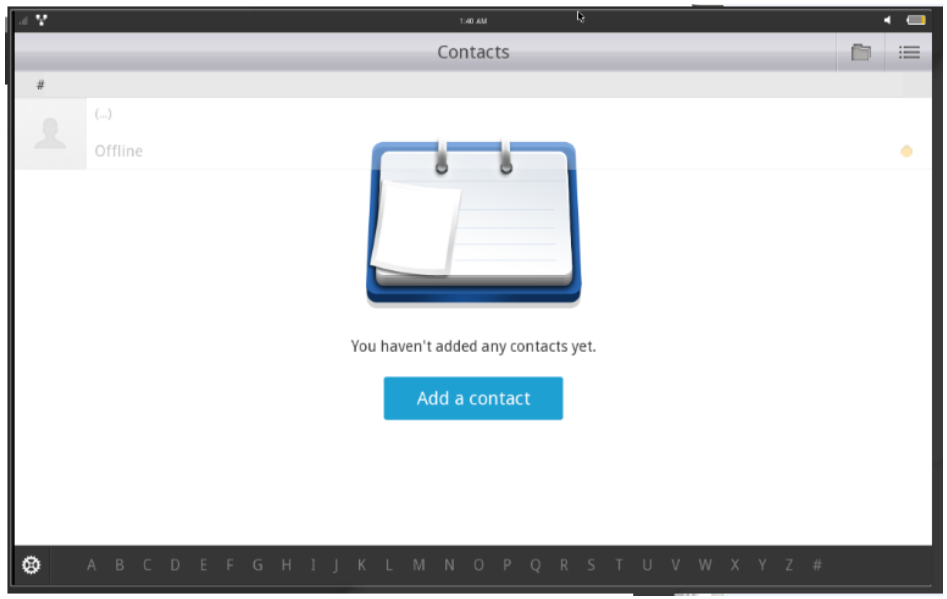
- Contact
- Versit

Why contacts is important

- Contact products
  - plaxo
- SNS contacts
  - renren.com
  - twitter
- IM contacts
  - QQ
  - MSN
- Email contacts
  - gmail
  - 126.com
- Mobile Phone contacts

Concept in Contact Module

- Manager
- Contact
- Detail
- Detail Definition
- Relationships
- Actions



Manager

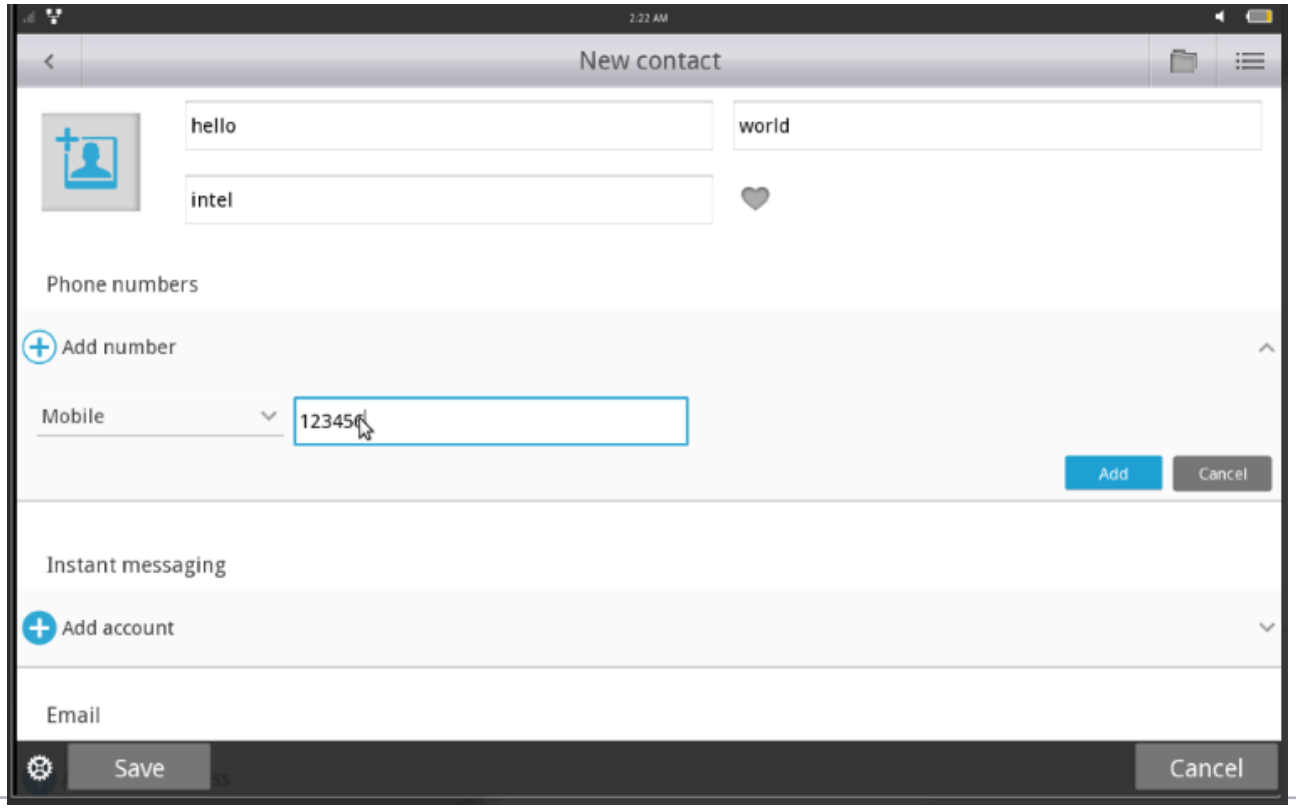
- Provides an abstraction of a datastore which contains contact information
- Provides methods to
  - retrieve and manipulate contact information
  - retrieve and manipulate relationship information
  - support schema definitions
- provides metadata and error information reporting

QContact

- QContact represents
  - individual contacts
  - groups
  - other types of contacts
- Have a collection of details
  - name
  - phone numbers
  - email address
- Have certain actions depending on details
- Have zero or more relationships with other contacts
  - hasMember

Contact

Example of creating a new contact in a manager



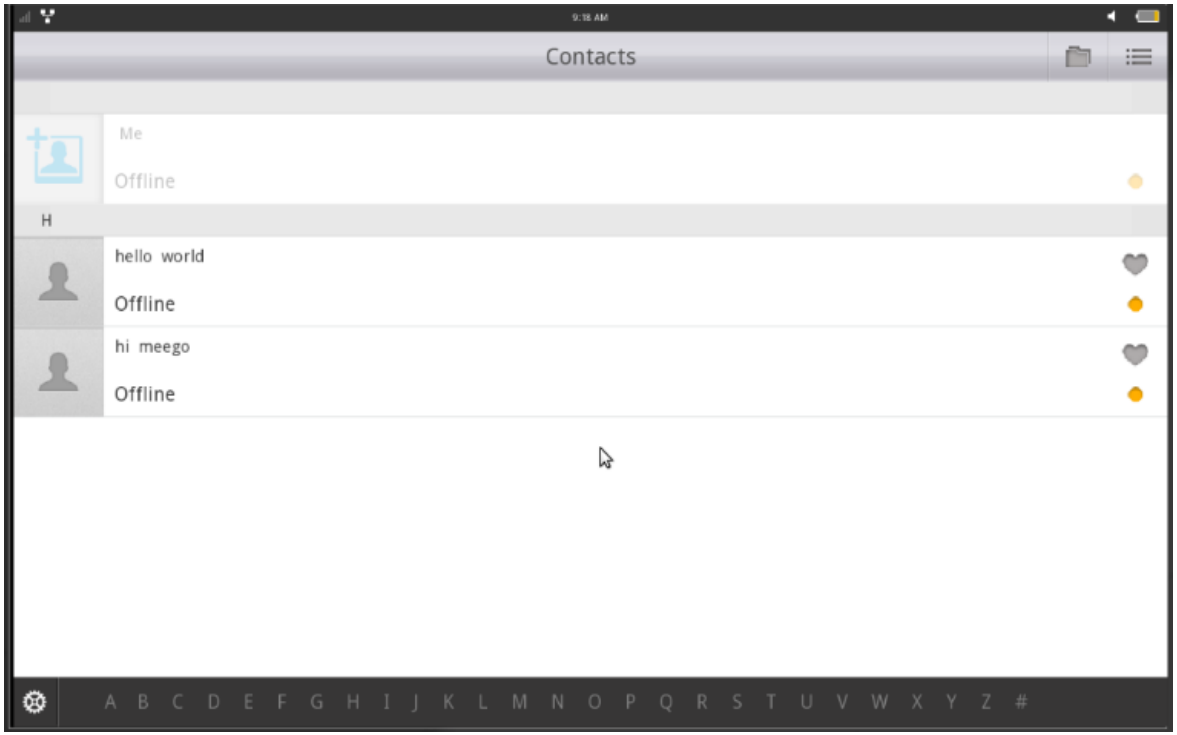
Contact

- Example of creating a new contact in a manager

```
QContact exampleContact;
QContactName nameDetail;
nameDetail.setFirstName("Adam");
nameDetail.setLastName("Unlikely");
QContactPhoneNumber phoneNumberDetail;
phoneNumberDetail.setNumber("+123 4567");
exampleContact.saveDetail(&nameDetail);
exampleContact.saveDetail(&phoneNumberDetail);
    // save the newly created contact in the manager
if (!m_manager.saveContact(&exampleContact))
    qDebug() << "Error" << m_manager.error()
            << “occurred whilst saving contact!”;
```

Contact

- Example of retrieving all contacts in a manager



Contact

- Example of retrieving all contacts in a manager

```
QList<QContact> results = m_manager.contacts(QContactPhoneNumber::match("+1234567"));
```

- Example of retrieving a special contact in a manager

```
QContact existing = m_manager.contact(exampleContact.localId());
```

Contact

- Example of updating an existing contacts in a manager

- phoneNumberDetail.setNumber("+123 9876");
exampleContact.saveDetail(&phoneNumberDetail);
m_manager.saveContact(&exampleContact);

Contact

- Example of removing a contact in a manager

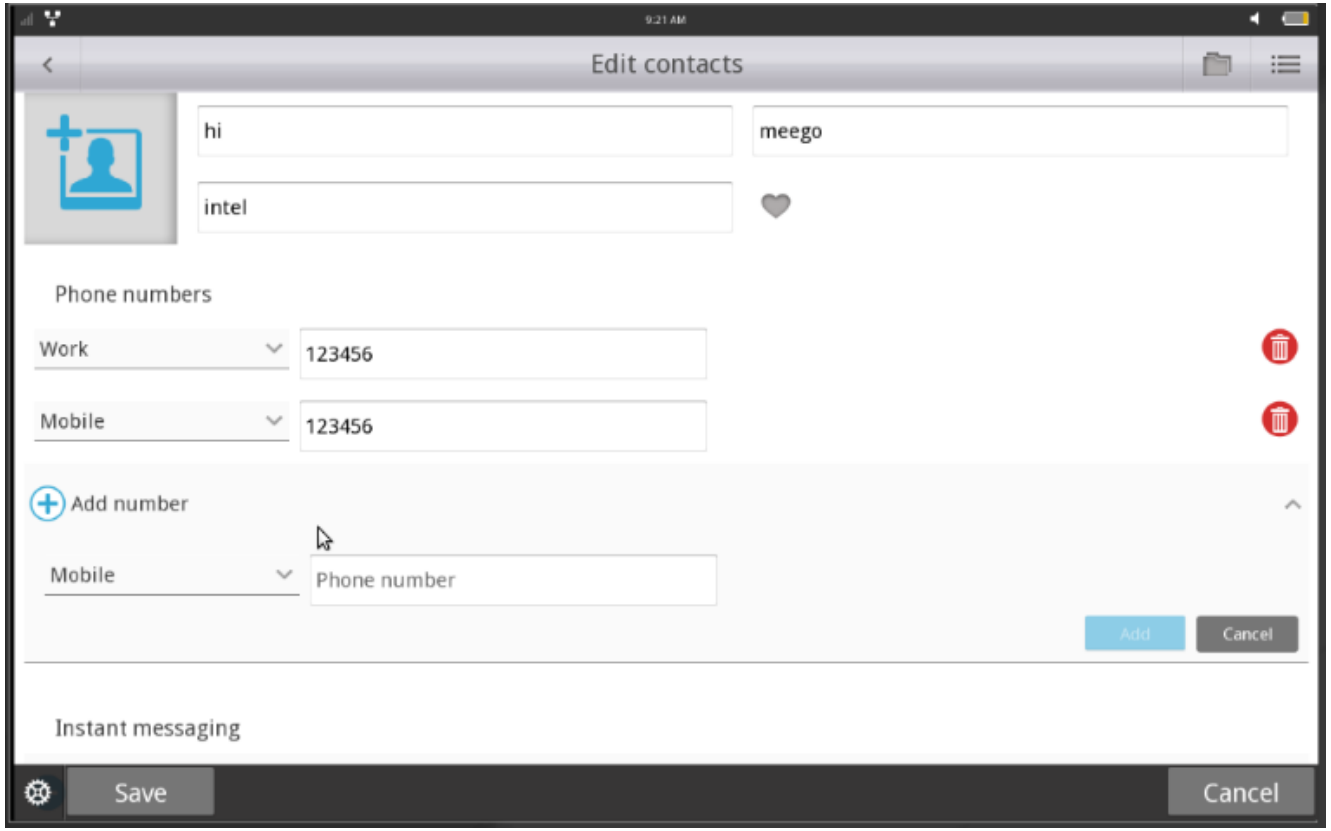
```
m_manager.removeContact(exampleContact.localId());
```

Detail

- A detail is a single, cohesive unit of information that is stored in a contact
- A detail is represented by class QContactDetail

Detail

- Example of deal with details



Detail

- Adding a detail to a contact

```
QContact exampleContact;
QContactName nameDetail;
nameDetail.setFirstName("Adam");
nameDetail.setLastName("Unlikely");
QContactPhoneNumber phoneNumberDetail;
phoneNumberDetail.setNumber("+123 4567");
exampleContact.saveDetail(&nameDetail);
exampleContact.saveDetail(&phoneNumberDetail);
```

Detail

- Updating a detail in a contact

```
phoneNumberDetail.setNumber("+123 9876");
    // overwrites old value on save

exampleContact.saveDetail(&phoneNumberDetail);
```

Detail

- Remove a detail from a contact

```
exampleContact.removeDetail(&phoneNumberDetail);
```

Detail

- View a specific detail from a contact

```
 void viewSpecificDetail(QContactManager* cm) {

        QList<QContactLocalId> contactIds = cm->contactIds();
QContact a = cm->contact(contactIds.first());
        qDebug()  << "The first phone number of"
                  << a.displayLabel()
                  << "is"

<<a.detail(QContactPhoneNumber::DefinitionName).

value(QContactPhoneNumber::FieldNumber);
 }
```

Relationships

- Relationship is represented by class QContactRelationship
- Class QContactRelationship describes a one-to-one relationship between two contacts
- Each relationship is combined with
  - first contact id
  - second contact id
  - relationship type

Relationships

- Relationships defined in QContactRelationship
  - QContactRelationship::Aggregates
    - The first contact as aggregating the second contact
  - QContactRelationship::HasAssistant
    - The second contact as being the assistant of first contact
  - QContactRelationship::HasManager
    - The second contact as being the manager of first contact
  - QContactRelationship::HasMember
    - The first contact as being a group including the second
  - QContactRelationship::HasSpouse
    - The second contact as being the spouse of first contact
  - QContactRelationship::IsSameAs
    - The first contact as being the same contact as the second

Relationships

- Example of creating a new relationship between two contacts

```
// first create the group and the group member
    QContact exampleGroup;
exampleGroup.setType(QContactType::TypeGroup);
    QContactNickname groupName;
    groupName.setNickname("Example Group");
exampleGroup.saveDetail(&groupName);

    QContact exampleGroupMember;
    QContactName groupMemberName;
groupMemberName.setFirstName("Member");
exampleGroupMember.saveDetail(&groupMemberName);
```

Relationships

- Example of creating a new relationship between two contacts

```
// second save those contacts in the manager

QMap<int, QContactManager::Error> errorMap;

QList<QContact> saveList;

saveList << exampleGroup << exampleGroupMember;
m_manager.saveContacts(&saveList, &errorMap);
```

Relationships

- Example of creating a new relationship between two contacts

```
// third create the relationship between those contacts

QContactRelationship groupRelationship;
groupRelationship.setFirst(exampleGroup.id());
groupRelationship.setRelationshipType(QContactRelationship::
HasMember);
groupRelationship.setSecond(exampleGroupMember.id());


// finally save the relationship in the manager
m_manager.saveRelationship(&groupRelationship);
```

Relationships

- Example of retrieving relationships between contacts

```
QList<QContactRelationship> groupRelationships =
m_manager.relationships(QContactRelationship::HasMember,
exampleGroup.id(), QContactRelationship::First);
QList<QContactRelationship> result;

for (int i = 0; i < groupRelationships.size(); i++) {

    if (groupRelationships.at(i).second() ==
exampleGroupMember.id()) {
result.append(groupRelationships.at(i));

    }

}
```

Relationships

- Example of removing relationships between contacts

```
m_manager.removeRelationship(groupRelationship);
```

Personal Info Management Module

- Contact
- Versit

Versit

- Convert QCatacts to and from vCards files
- Convert QOrganizerItems to and from iCalendar files

vCard

- vCard is a file standard for electronic business cards
- vCard file contains
  - name
  - address
  - phone number
  - e-mail

```
BEGIN:VCARD
VERSION:2.1
N:Gump;Forrest
FN:Forrest Gump
ORG:Bubba Gump Shrimp Co.
TITLE:Shrimp Man
TEL;WORK;VOICE:(111) 555-1212
TEL;HOME;VOICE:(404) 555-1212
ADR;WORK:;;100 Waters Edge;Baytown;LA;30314;United States of America
LABEL;WORK;ENCODING=QUOTED-PRINTABLE:100 Waters Edge=0D=0ABaytown, LA 30314=0D=0AUnited States of America
ADR;HOME:;;42 Plantation St.;Baytown;LA;30314;United States of America
LABEL;HOME;ENCODING=QUOTED-PRINTABLE:42 Plantation St.=0D=0ABaytown, LA 30314=0D=0AUnited States of America
EMAIL;PREF;INTERNET:forrestgump@example.com
REV:20080424T195243Z
END:VCARD
```

iCalendar

- A computer file format which allows Internet users to send meeting requests and tasks to other Internet users, via email

```
BEGIN:VCALENDAR
VERSION:2.0
PRODID:-//hacksw/handcal//NONSGML v1.0//EN
BEGIN:VEVENT
UID:uid1@example.com
DTSTAMP:19970714T170000Z
ORGANIZER;CN=John Doe:MAILTO:john.doe@example.com
DTSTART:19970714T170000Z
DTEND:19970715T035959Z
SUMMARY:Bastille Day Party
END:VEVENT
END:VCALENDAR
```

Versit

- Main classes in versit module
  - QVersitProperty
  - QVersitDocument
  - QVersitReader
  - QVersitWriter
  - QVersitContactImporter
  - QVersitContactExporter
  - QVersitOrganizerImporter
  - QVersitQrganizerExporter

QVersitDocument

- A container for a list of versit properties
- Abstraction of vCard and iCalendar

QVersitProperty

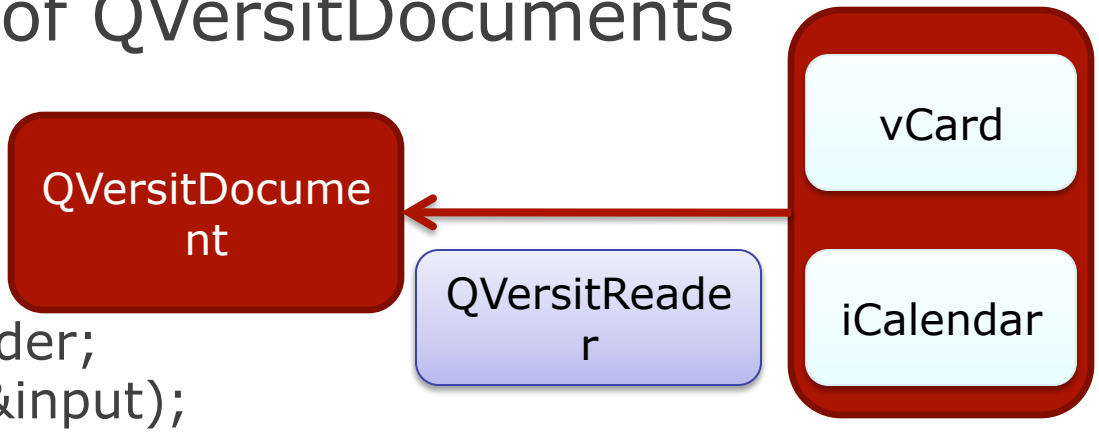
- A QVersitProperty consists of
  - a list of groups
  - a name
  - a list of parameters (key/value pairs)
  - a value
- Example

```
QVersitDocument document;

QVersitProperty property;
property.setName(QString::fromAscii("N"));
property.setValue("Citizen;John;Q;;");
document.addProperty(property);
```

QVersitReader

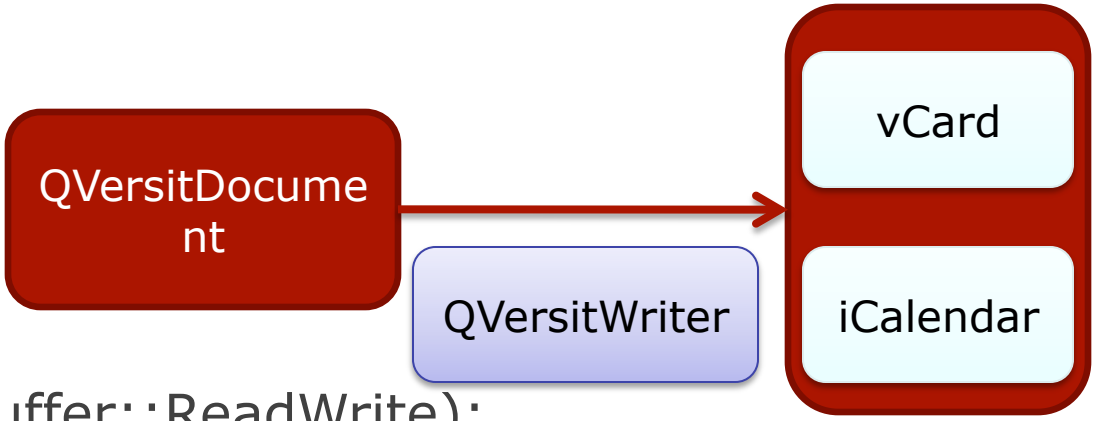
- Parse a vCard or iCalendar from an I/O device to produce a list of QVersitDocuments
- Example




```
QVersitReader reader;
reader.setDevice(&input);
reader.startReading();
reader.waitForFinished();
QList<QVersitDocument> inputDocuments =
reader.results();
```

QVersitWriter

- Writes Versit documents such as vCards to a device



- Example

```
QBuffer output;
output.open(QBuffer::ReadWrite);
QVersitWriter writer;
writer.setDevice(&output);
writer.startWriting(outputDocuments);
writer.waitForFinished();
```

QVersitContactImporter

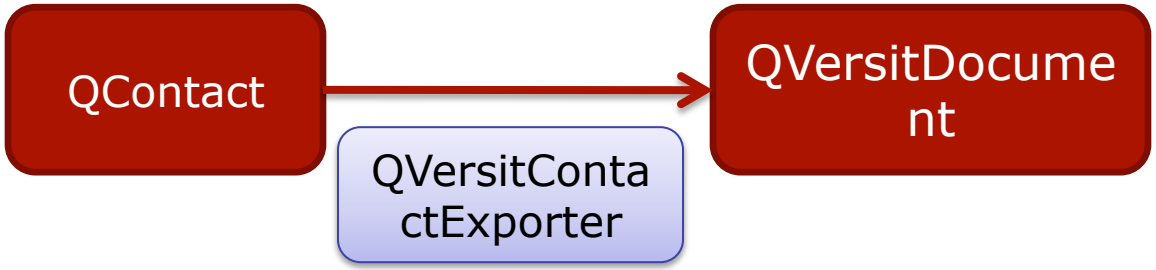
- Convert QVersitDocuments to QContacts
- Example

```
QVersitContactImporter importer;
if (!importer.importDocuments(inputDocuments))
        return;
QList<QContact> contacts = importer.contacts();
```

QVersitContactImporter

- Convert QVersitDocuments to QContacts
- Example

```
QVersitContactExporter exporter;
If (!exporter.exportContacts(contacts,
        QVersitDocument::VCard30Type))
        return;
QList<QVersitDocument> outputDocuments =
exporter.documents();
```



Outline

- MeeGo Device and Information Management Module Overview
- Personal Information Management Module
- System Module

System Module

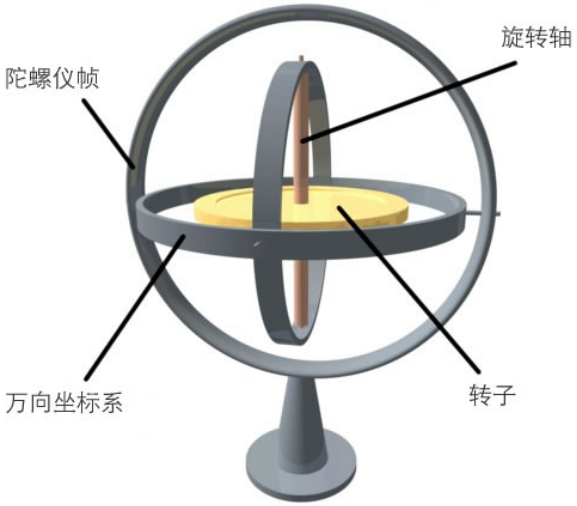
- Sensors
- System Information

System Module

- Sensors
- System Information

Sensor

- Some types of Sensor
  - Accelerometer
  - Gyroscope
  - Light Sensor
  - Orientation Sensor
  - Magnetometer
  - Proximity Sensor
  - Compass



Sensor

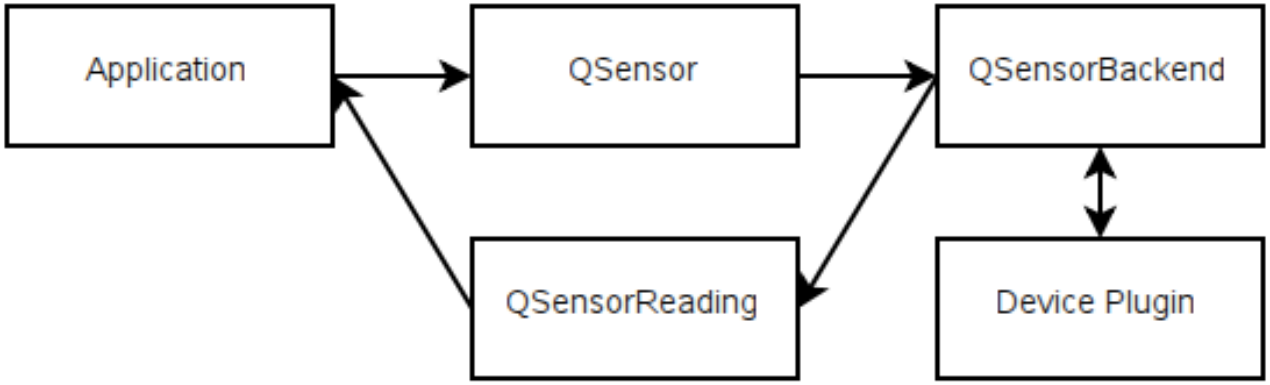
- Some applications of sensor
  - Image Sensor
    - Face identification
    - Business card identification
  - Light Sensor
    - Change backlight automatically
  - Accelerometer Sensor
    - Game
    - Pedometer
    - Alarming

Sensor

- Main Classes in Sensor Module
  - QSensor
  - QSensorFilter
  - QSensorReading

Sensor

- The QSensor class represents a single hardware sensor



QSensorReading

- Holds the readings from the sensor
- Accessed by reading() function in Qsensor
- Example:

```
QAccelerometerReading *reading = sensor.reading();

qreal x = reading->x();
```

Sensor

- The life cycle of a sensor is typically
  - Create a sub-class of QSensor on the stack or heap

    QAccelerometer sensor;
  - Setup as required by the application

    connect(sensor, SIGNAL(readingChanged()), this, SLOT(checkReading()));
  - Start receiving values

    sensor.start();

Sensor

- The life cycle of a sensor is typically
  - Sensor data is used by the application

```
void MyClass::checkReading() {
        sensor->reading()->x();
}
```

  - Stop receiving values

```
sensor.stop();
```

System Module

- Sensors
- System Information

System Information

- Provides system related information and capabilities

System Information Categories

- Version
  - Contains version information for supporting software on the device e.g. OS, firmware, Qt
- Features Supported
  - This lists the supported hardware on the device
    - e.g. camera, bluetooth, GPS, FM radio
- Network
  - State of network connection
  - Type of network e.g. gsm, cdma, ethernet

System Information Categories

- Display Information
  - ColorDepth
  - Brightness
- Storage Information
  - The presence of various storage device
- Device Information
  - Battery Status
  - Power State
  - Profile(silent, vibrating, normal)
  - Sim
  - Input Method(key, single touch screen etc)

Main Classes in System Information

- QSystemInfo
- QSystemDeviceInfo
- QSystemBatteryInfo
- QSystemNetworkInfo
- QSystemStorageInfo

QSystemInfo

- Provides access to various general information from the system
  - Feature supported
  - Version

QSystemInfo

- Example of using QSystemInfo

```
QSystemInfo s;
//print the current country code
qDebug() << s.currentCountryCode();
//print whether camera is supported
qDebug() <<
s.hasFeatureSupported(QSystemInfo::CameraFeature);
//print the version of QtMobility
qDebug << s.version(QSystemInfo::QtMobility);
```

QSystemDeviceInfo

- Provides access to device information from the system
  - BatteryStatus
  - InputMethod
  - SimStatus
  - Profile
  - PowerState
  - etc

QSystemDeviceInfo

- Example of using QSystemDeviceInfo

```
QSystemDeviceInfo deviceInfo;
qDebug << “battery status” << deviceInfo.batteryStatus();
qDebug << “power state” << deviceInfo.currentPowerState();
qDebug << “current profile” << deviceInfo.currentProfile ();
qDebug << “input method type ” << deviceInfo.inputMethodType ();
qDebug << “sim status” << deviceInfo.simStatus();
```

QSystemBatteryInfo

- Provides access to battery and power information from the system
  - BatteryStatus
  - ChargerType
  - ChargingState
  - EnergyUnit

QSystemBatteryInfo

- Example of using QSystemBatteryInfo

```
QSystemBatteryInfo batteryInfo;
qDebug() << batteryInfo.batteryStatus();
qDebug() << batteryInfo.chargerType ();
qDebug() << batteryInfo.chargingState();
qDebug() << batteryInfo.energyMeasurementUnit();
qDebug() << batteryInfo.maxBars();
qDebug() << batteryInfo.nominalCapacity ();
```

QSystemNetworkInfo

- Provides access to various networking status and signals
  - NetworkMode
  - NetworkStatus

QSystemNetworkInfo

- Example of using QSystemNetworkInfo

```
QSystemNetworkInfo networkInfo;
qDebug() << networkInfo.currentMobileCountryCode();
qDebug() << networkInfo.currentMobileNetworkCode();
qDebug() << networkInfo.currentMode();
```

Resource

- Website
  - http://meego.com/developers
  - http://doc.qt.nokia.com
  - http://doc.qt.nokia.com/qtmobility-1.2.0-beta1/contacts.html
  - http://doc.qt.nokia.com/qtmobility-1.2.0-beta1/versit.html
  - http://doc.qt.nokia.com/qtmobility-1.2.0-beta1/sensors-api.html
  - http://doc.qt.nokia.com/qtmobility-1.2.0-beta1/systeminfo.html

Legal Disclaimer

- INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL® PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECUTAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL® PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT. INTEL PRODUCTS ARE NOT INTENDED FOR USE IN MEDICAL, LIFE SAVING, OR LIFE SUSTAINING APPLICATIONS.
- Intel may make changes to specifications and product descriptions at any time, without notice.
- All products, dates, and figures specified are preliminary based on current expectations, and are subject to change without notice.
- Intel, processors, chipsets, and desktop boards may contain design defects or errors known as errata, which may cause the product to deviate from published specifications. Current characterized errata are available on request.
- Intel and the Intel logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.
- *Other names and brands may be claimed as the property of others.
- Copyright © 2010 Intel Corporation.