Get device information using Qt

This code example shows how to access device information from the system. For example to get IMEI, IMSI, Manufacturer name, Model name, Product name, Battery level, Device lock information, Battery status, Current power state, SIM status, Current profile and Supported input methods of device. Code example use `QSystemDeviceInfo` API from `QtMobility` APIs.

This article needs to be updated: If you found this article useful, please fix the problems below then delete the {{ArticleNeedsUpdate}} template from the article to remove this warning.

Reasons: hamishwillee
(16 Feb 2012)
The article explains the supported generic Qt Mobility API, but provides example/code snippets using deprecated QWidget UI. This example should be updated to not use QWidget.

Headers required

```
#include <qsysteminfo.h>
```

.pro file

Modify your **.pro** file as follows.

```
QT += core \
    gui \
    network
INCLUDEPATH += d:/qtmobility/src/systeminfo
INCLUDEPATH += D:/QtMobility/src/global

CONFIG += mobility
MOBILITY = systeminfo
symbian:TARGET.CAPABILITY = ReadDeviceData
```

Header file

```
/* GetDeviceInfo.h */

#ifndef GETDEVICEINFO_H
#define GETDEVICEINFO_H

#include <QtGui/QMainWindow>
#include <qsysteminfo.h>

QTM_USE_NAMESPACE;

class QListWidget;
class QScrollArea;

class GetDeviceInfo : public QMainWindow
{
    Q_OBJECT

public:
 GetDeviceInfo(QWidget *parent = 0);
    ~GetDeviceInfo();

private:
 void resizeEvent (QResizeEvent* event);
 void setWidgetGeometry(QSize* screenSize);
    void getDeviceInformation();

private:
    QSystemDeviceInfo* sysInfo;
 QListWidget     *listWidget;
 QScrollArea     *scrollArea;
};

#endif // GETDEVICEINFO_H
```

Source file

```
/* GetDeviceInfo.cpp */
#include "GetDeviceInfo.h"

#include <QListWidget>
#include <QScrollArea>
#include <QResizeEvent>

GetDeviceInfo::GetDeviceInfo(QWidget *parent)
    : QMainWindow(parent)
{
 setWindowTitle("Get device information");

 sysInfo = new QSystemDeviceInfo(this);
```


(C) Copyright Nokia 2013. All rights reserved.

```
 /* To show items in list view */
 listWidget = new QListWidget();
 listWidget->setObjectName(QString::fromUtf8("listWidget"));

 /* To make UI scrollable */
 scrollArea = new QScrollArea();
 setWidgetGeometry(&(size()));

 /* scrollArea takes ownership of the listWidget, so no need to delete it */
 scrollArea->setWidget(listWidget);
 scrollArea->setAlignment(Qt::AlignLeft);
 scrollArea->setWidgetResizable(true);

 /* QMainWindow takes ownership of the scrollArea, so no need to delete it */
 setCentralWidget(scrollArea);
 getDeviceInformation();
}

GetDeviceInfo::~GetDeviceInfo()
{
}

void GetDeviceInfo::resizeEvent (QResizeEvent* event)
{
    QSize widgetSize = event->size();
    setWidgetGeometry(&widgetSize);
    QMainWindow::resizeEvent(event);
}

void GetDeviceInfo::setWidgetGeometry(QSize* screenSize)
{
 resize(*screenSize);
 QRect rect(0,0,screenSize->width(), screenSize->height());
 listWidget->setGeometry(rect);
}

void GetDeviceInfo::getDeviceInformation()
{
 QString strInfo;

 /* get IMEI of device */
 strInfo.append("IMEI: " + sysInfo->imei());
 listWidget->addItem(strInfo);

 /* get IMSI of device */
 strInfo.clear();
 strInfo.append("IMSI: " + sysInfo->imsi());
 listWidget->addItem(strInfo);

 /* get manufacturer of device */
 strInfo.clear();
 strInfo.append("Manufacturer: " + sysInfo->manufacturer());
 listWidget->addItem(strInfo);

 /* get model number of device */
 strInfo.clear();
 strInfo.append("Model: " + sysInfo->model());
 listWidget->addItem(strInfo);

 /* get product name of device */
 strInfo.clear();
 strInfo.append("Product name: " + sysInfo->productName());
 listWidget->addItem(strInfo);

 /* get battery level of device*/
 strInfo.clear();
 strInfo.setNum(sysInfo->batteryLevel());
 strInfo.prepend("Battery level: ");
 strInfo.append("%");
 listWidget->addItem(strInfo);

 /* check if device locked   */
 strInfo.clear();
 strInfo.append("Device locked: ");
 if(sysInfo->isDeviceLocked())
  strInfo.append("Yes");
 else
  strInfo.append("No");

 listWidget->addItem(strInfo);

 /* get battery status of device*/
 strInfo.clear();
 strInfo.append("Battery status: ");

 switch(sysInfo->batteryStatus ())
 {
  case QSystemDeviceInfo::BatteryNormal:
  {
   strInfo.append("Normal");
   break;
  }
  case QSystemDeviceInfo::BatteryLow:
  {
   strInfo.append("Low");
   break;
  }
  case QSystemDeviceInfo::BatteryVeryLow:
  {
   strInfo.append("Very Low");
   break;
  }
  case QSystemDeviceInfo::BatteryCritical:
  {
   strInfo.append("Critical");
   break;
  }
  case QSystemDeviceInfo::NoBatteryLevel:
  default:
  {
   strInfo.append("Battery level not found");
   break;
  }
 }

 listWidget->addItem(strInfo);

 /* get current power state of device*/
 strInfo.clear();
 strInfo.append("Current power state: ");
```


(C) Copyright Nokia 2013. All rights reserved.

```
switch(sysInfo->currentPowerState())
{
 case QSystemDeviceInfo::BatteryPower:
 {
  strInfo.append("Battery power");
  break;
 }
 case QSystemDeviceInfo::WallPower:
 {
  strInfo.append("Wall power");
  break;
 }
 case QSystemDeviceInfo::WallPowerChargingBattery:
 {
  strInfo.append("Wall power charging battery");
  break;
 }
 case QSystemDeviceInfo::UnknownPower:
 default:
 {
  strInfo.append("Power error!!");
  break;
 }
}

listWidget->addItem(strInfo);

/* get SIM status of device*/
strInfo.clear();
strInfo.append("SIM status: ");

switch(sysInfo->simStatus())
{
 case QSystemDeviceInfo::SingleSimAvailable:
 {
  strInfo.append("Single sim available");
  break;
 }
 case QSystemDeviceInfo::DualSimAvailable:
 {
  strInfo.append("Dual sim available");
  break;
 }
 case QSystemDeviceInfo::SimLocked:
 {
  strInfo.append("Sim locked");
  break;
 }
 case QSystemDeviceInfo::SimNotAvailable:
 {
  strInfo.append("Sim not available");
  break;
 }
 default:
 {
  strInfo.append("Error in getting status!!");
  break;
 }
}

listWidget->addItem(strInfo);

/* get current profile of device*/
strInfo.clear();
strInfo.append("Current profile: ");

switch(sysInfo->currentProfile())
{
 case QSystemDeviceInfo::NormalProfile:
 {
  strInfo.append("General");
  break;
 }
 case QSystemDeviceInfo::SilentProfile:
 {
  strInfo.append("Silent");
  break;
 }
 case QSystemDeviceInfo::VibProfile:
 {
  strInfo.append("Vibrate");
  break;
 }
 case QSystemDeviceInfo::LoudProfile:
 {
  strInfo.append("Loud");
  break;
 }
 case QSystemDeviceInfo::OfflineProfile:
 {
  strInfo.append("Offline ");
  break;
 }
 case QSystemDeviceInfo::CustomProfile:
 {
  strInfo.append("Custom profile");
  break;
 }
 case QSystemDeviceInfo::PowersaveProfile:
 {
  strInfo.append("Powersave ");
  break;
 }
 case QSystemDeviceInfo::UnknownProfile:
 default:
 {
  strInfo.append("Error in profile!!");
  break;
 }
}

listWidget->addItem(strInfo);

/* get supported input methods of device*/
strInfo.clear();
strInfo.append("Supported input methods are: ");
listWidget->addItem(strInfo);

QSystemDeviceInfo::InputMethodFlags inputMethods = sysInfo->inputMethodType();
```


(C) Copyright Nokia 2013. All rights reserved.

```
  strInfo.clear();
  strInfo.append("  Keys: ");
  if((inputMethods & QSystemDeviceInfo::Keys))
  {
   strInfo.append("Yes");

  }
  else
  {
   strInfo.append("No");
  }

  listWidget->addItem(strInfo);

  strInfo.clear();
  strInfo.append("  Keypad: ");
  if((inputMethods & QSystemDeviceInfo::Keypad))
  {
   strInfo.append("Yes");

  }
  else
  {
   strInfo.append("No");
  }

  listWidget->addItem(strInfo);


  strInfo.clear();
  strInfo.append("  Keyboard: ");
  if((inputMethods & QSystemDeviceInfo::Keyboard))
  {
   strInfo.append("Yes");

  }
  else
  {
   strInfo.append("No");
  }

  listWidget->addItem(strInfo);

  strInfo.clear();
  strInfo.append("  Single touch: ");
  if((inputMethods & QSystemDeviceInfo::SingleTouch))
  {
   strInfo.append("Yes");

  }
  else
  {
   strInfo.append("No");
  }

  listWidget->addItem(strInfo);

  strInfo.clear();
  strInfo.append("  Multi touch: ");
  if((inputMethods & QSystemDeviceInfo::MultiTouch))
  {
   strInfo.append("Yes");

  }
  else
  {
   strInfo.append("No");
  }

  listWidget->addItem(strInfo);

  strInfo.clear();
  strInfo.append("  Mouse: ");
  if((inputMethods & QSystemDeviceInfo::Mouse))
  {
   strInfo.append("Yes");

  }
  else
  {
   strInfo.append("No");
  }

  listWidget->addItem(strInfo);
 }
```

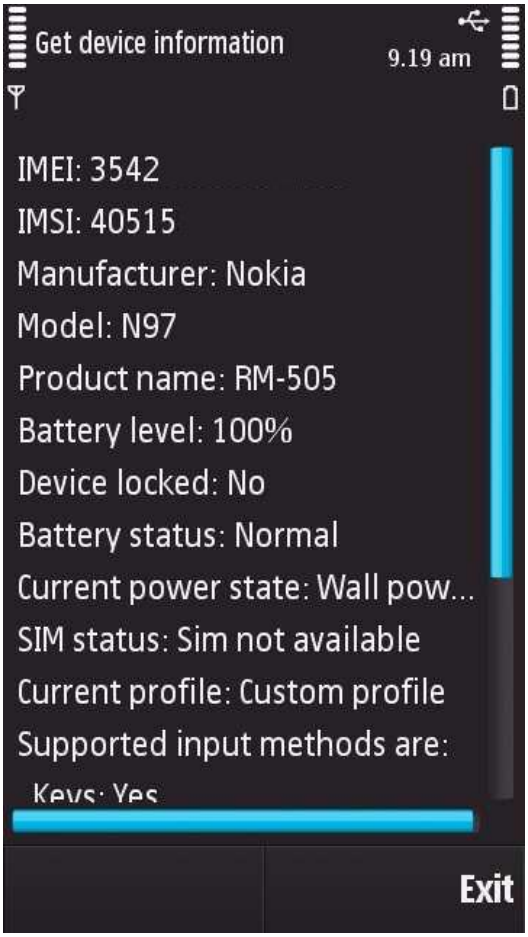
Post-conditions

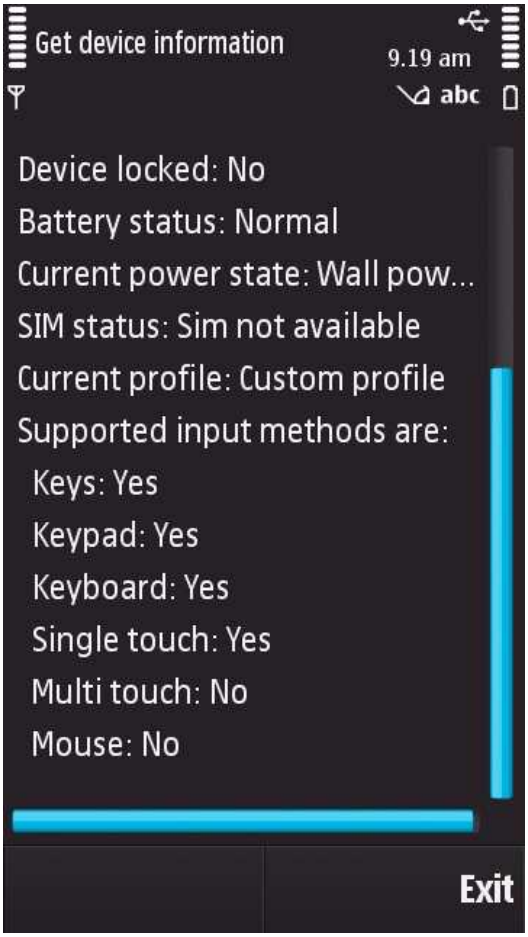
The code snippet is expected to read device information from device and show it on screen.

Device information 1

Device information 2

(C) Copyright Nokia 2013. All rights reserved.





Download Code Example

- Working Code Example is available to download from here.

Related articles

- Get Display information using Qt
- Get Network information using Qt
- Get Storage information using Qt
- Get System information using Qt

(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.