QMultiMap+QSharedPointer

Введение

Библиотека Qt предоставляет множество удобных средств для разработки приложений, например api для взаимодействия с мобильными платформами или набор классов для GUI программ. В данной статье речь пойдет об одной не совсем очевидной особенности совместного использования таких классов как QMultiMap и QSharedPointer. В качестве среды используется Ubuntu 10.04 с установленным Qt 4.6 .

Проблема

Впервые с данной проблемой я столкнулся, когда попробовал скомпилировать код, написанный для Qt 4.7, с использованием Qt 4.6. Часть незначительных ошибок удалось решить, исключив использование функций, появившихся в новой версии Qt. Однако одна ошибка казалась очень загадочной.

```
 In file included from /usr/include/qt4/QtCore/qsharedpointer.h:52,
 from /usr/include/qt4/QtCore/QSharedPointer:1,
 /usr/include/qt4/QtCore/qsharedpointer_impl.h: In constructor
'QtSharedPointer::ExternalRefCount<T>::ExternalRefCount(const
QtSharedPointer::ExternalRefCount<X>&) [with X = Channel, T = DataMark]':
 /usr/include/qt4/QtCore/qsharedpointer_impl.h:461: instantiated from
'QSharedPointer<T>::QSharedPointer(const QSharedPointer<X>&) [with X =
Channel, T = DataMark]'
 /usr/include/qt4/QtCore/qmap.h:782: instantiated from 'QList<T> QMap<Key,
T>::uniqueKeys() const [with Key = QSharedPointer<Channel>, T =
QSharedPointer<DataMark>]'
 src/RSSFeedJSON.cpp:129: instantiated from here
 /usr/include/qt4/QtCore/qsharedpointer_impl.h:378: error: no matching
function for call to 'QtSharedPointer::Basic<DataMark>::Basic(Channel*
const&)'
 /usr/include/qt4/QtCore/qsharedpointer_impl.h:150: note: candidates are:
QtSharedPointer::Basic<T>::Basic(Qt::Initialization) [with T = DataMark]
 /usr/include/qt4/QtCore/qsharedpointer_impl.h:149: note:
QtSharedPointer::Basic<T>::Basic(T*) [with T = DataMark]
 /usr/include/qt4/QtCore/qsharedpointer_impl.h:123: note:
QtSharedPointer::Basic<DataMark>::Basic(const
QtSharedPointer::Basic<DataMark>&)
```

Больше всех настораживало сообщение

```
 /usr/include/qt4/QtCore/qmap.h:782: instantiated from 'QList<T> QMap<Key,
T>::uniqueKeys() const [with Key = QSharedPointer<Channel>, T =
QSharedPointer<DataMark>]'
```

так как сигнатура функции uniqueKeys должна выглядеть иначе.

Дальнейшие изыскания позволили выяснить, что проблема заключалась в использовании QMultiMap, у которого в качестве типа для ключевых значений был использован тип QSharedPointer<...>.

Рассмотрим небольшой пример, который наглядно демонстрирует специфику совместного использования QMultiMap и QSharedPointer в Qt 4.6.

```
#include <QSharedPointer>
#include <QMultiMap>
#include <QList>
#include <cstdio>

int main(int argc,char** argv){
        QMultiMap<QSharedPointer<int>, QString> map;
        int* a,*b;
        a=new int;
        b=new int;
        QSharedPointer<int> aa(a);
        QSharedPointer<int> bb(b);
        map.insert(aa,"q");
        map.insert(bb,"qq");
        map.insert(bb,"qqq");
        QList<QSharedPointer<int> > unq=map.uniqueKeys();
        return 0;
}
```

Если мы попытаемся скомпилировать данный пример, то получим следующие ошибки:

```
In file included from /usr/include/qt4/QtCore/QMultiMap:1,
                 from main.cpp:2:
/usr/include/qt4/QtCore/qmap.h: In member function 'QList<T> QMap<Key,
T>::uniqueKeys() const [with Key = QSharedPointer<int>, T = QString]':
main.cpp:23:   instantiated from here
/usr/include/qt4/QtCore/qmap.h:782: error: no match for 'operator<' in 'aKey
< i.QMap<Key, T>::const_iterator::key [with Key = QSharedPointer<int>, T =
QString]()'
/usr/include/qt4/QtCore/qchar.h:385: note: candidates are: bool
operator<(QChar, QChar)
/usr/include/qt4/QtCore/qbytearray.h:520: note:                 bool
operator<(const QByteArray&, const QByteArray&)
/usr/include/qt4/QtCore/qbytearray.h:522: note:                 bool
operator<(const QByteArray&, const char*)
/usr/include/qt4/QtCore/qbytearray.h:524: note:                 bool
operator<(const char*, const QByteArray&)
/usr/include/qt4/QtCore/qstring.h:927: note:                bool
operator<(const char*, const QString&)
/usr/include/qt4/QtCore/qstring.h:940: note:                bool
operator<(const char*, const QLatin1String&)
/usr/include/qt4/QtCore/qstring.h:953: note:                bool
```

```
operator<(const QLatin1String&, const QLatin1String&)
/usr/include/qt4/QtCore/qstring.h:1181: note:                 bool
operator<(const QStringRef&, const QStringRef&)
/usr/include/qt4/QtCore/qmap.h: In function 'bool qMapLessThanKey(const
Key&, const Key&) [with Key = QSharedPointer<int>]':
/usr/include/qt4/QtCore/qmap.h:760:   instantiated from 'QMapData::Node*
QMap<Key, T>::mutableFindNode(QMapData::Node**, const Key&) const [with Key
= QSharedPointer<int>, T = QString]'
/usr/include/qt4/QtCore/qmap.h:576:   instantiated from 'QMap<Key,
T>::iterator QMap<Key, T>::insertMulti(const Key&, const T&) [with Key =
QSharedPointer<int>, T = QString]'
/usr/include/qt4/QtCore/qmap.h:953:   instantiated from 'typename QMap<Key,
T>::iterator QMultiMap<Key, T>::insert(const Key&, const T&) [with Key =
QSharedPointer<int>, T = QString]'
main.cpp:20:   instantiated from here
/usr/include/qt4/QtCore/qmap.h:107: error: no match for 'operator<' in 'key1
< key2'
/usr/include/qt4/QtCore/qchar.h:385: note: candidates are: bool
operator<(QChar, QChar)
/usr/include/qt4/QtCore/qbytearray.h:520: note:                 bool
operator<(const QByteArray&, const QByteArray&)
/usr/include/qt4/QtCore/qbytearray.h:522: note:                 bool
operator<(const QByteArray&, const char*)
/usr/include/qt4/QtCore/qbytearray.h:524: note:                 bool
operator<(const char*, const QByteArray&)
/usr/include/qt4/QtCore/qstring.h:927: note:                bool
operator<(const char*, const QString&)
/usr/include/qt4/QtCore/qstring.h:940: note:                bool
operator<(const char*, const QLatin1String&)
/usr/include/qt4/QtCore/qstring.h:953: note:                bool
operator<(const QLatin1String&, const QLatin1String&)
/usr/include/qt4/QtCore/qstring.h:1181: note:                 bool
operator<(const QStringRef&, const QStringRef&)
make: *** [main.o] Ошибка 1
```

Обратим внимание на упоминаемую одной из первых функцию

```
‘QList<T> QMap<Key, T>::uniqueKeys() const [with Key = QSharedPointer<int>,
T = int]’
```

из-за которой и происходит сбой компиляции. Вот так выглядит ее прототип в документации Qt uniqueKeys :

```
QList<Key> QMap::uniqueKeys () const
```

Очевидно, что возвращаемые значения в обоих случаях не совпадают и нам удалось повторить ошибку на более простом примере.
Обратимся к описанию данной функции. Исходя из документации она возвращает список всех ключей в QMultiMap в порядке возрастания. Логично предположить, что основная проблема заключается в сортировке полученного списка ключей, ввиду отсутствия оператора сравнения определенного для типа QSharedPointer<T>. Определим свой оператор сравнения следующим

образом:

```
template <class T>
bool operator <(const  QSharedPointer<T>& a, const  QSharedPointer<T>& b){
        return a.data()<b.data();
}
```

Если включить код данного оператора в код примера, то компиляция пройдет нормально и мы получим работающее приложение.

Проблема, похожая на описанную выше, возникает при замене QMultiMap на QMultiHash в примере. Для нормальной компиляции в таком случае необходимо определить собственную функцию qHash со следующим прототипом:

```
template <class T>
uint qHash(const QSharedPointer<T>& a);
```

Заключение

Очевидно, что перегрузка оператора “<” для типа QSharedPointer<T> также следует и из сообщений компилятора. А это показывает, что написание простого примера ошибки в большом приложении позволяет весьма тривиальным образом найти ее причины. Описанная проблема устранена в новой версии Qt 4.7, которая входит в состав Qt SDK 1.1. Однако предложенное здесь решение остается актуальным, так как на момент написания статьи в OVI находится Qt 4.6.