



ID	CS001436	Creation date	June 18, 2009
Platform	Qt	Tested on devices	5800 XpressMusic
Category	Qt for Symbian	Subcategory	

Keywords (APIs, classes, methods, functions): Qt, XML, SAX, QXmlStreamReader

Overview

This example shows you how to parse well-formed XML with `QXmlStreamReader`. Qt documentation states that it is a faster and more convenient replacement for Qt's own SAX parser (`QXmlSimpleReader`) and it's also faster, as well as in some cases more convenient, than a DOM tree (`QDomDocument`).

Preconditions

- Install latest Qt for S60 see [Qt for S60 - Installation packages](#)
- Check this link for installation guide: [How to install the package.](#)
- Go through this article: [Getting started with Qt for S60](#)

QXmlStreamReader Example

XML file

```
<?xml version="1.0" encoding="UTF-8" ?>
<persons>
  <person id="1">
    <firstname>John</firstname>
    <surname>Doe</surname>
    <email>john.doe@example.com</email>
    <website>http://en.wikipedia.org/wiki/John_Doe</website>
  </person>
  <person id="2">
    <firstname>Jane</firstname>
    <surname>Doe</surname>
    <email>jane.doe@example.com</email>
    <website>http://en.wikipedia.org/wiki/John_Doe</website>
  </person>
  <person id="3">
    <firstname>Matti</firstname>
```

CS001436_-_Using_QXmlStreamReader_to_parse_XML

```
<surname>Meikäläinen</surname>
<email>matti.meikalainen@example.com</email>
<website>http://fi.wikipedia.org/wiki/Matti_Meikäläinen</website>
</person>
</persons>
```

Header file

```
#include <QtXml/QXmlStreamReader>

class QXSRExample : public QMainWindow {
    Q_OBJECT

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

private slots:
    void parseXML();

private:
    QPointer<QVBoxLayout> _layout;

    void setupUI();

    QMap<QString, QString> parsePerson(QXmlStreamReader& xml);
    void addElementDataToMap(QXmlStreamReader& xml,
                             QMap<QString, QString>& map) const;

    void addPersonsToUI(QList< QMap<QString,QString> >& persons);
};
```

Source file

```
void QXSRExample::parseXML() {
    /* We'll parse the example.xml */
    QFile* file = new QFile(":/example.xml");
    /* If we can't open it, let's show an error message. */
    if (!file->open(QIODevice::ReadOnly | QIODevice::Text)) {
        QMessageBox::critical(this,
                               "QXSRExample::parseXML",
                               "Couldn't open example.xml",
                               QMessageBox::Ok);
        return;
    }
    /* QXmlStreamReader takes any QIODevice. */
    QXmlStreamReader xml(file);
    QList< QMap<QString,QString> > persons;
    /* We'll parse the XML until we reach end of it.*/
    while(!xml.atEnd() &&
           !xml.hasError()) {
        /* Read next element.*/
        QXmlStreamReader::TokenType token = xml.readNext();
        /* If token is just StartDocument, we'll go to next.*/
        if(token == QXmlStreamReader::StartDocument) {
            continue;
        }
        /* If token is StartElement, we'll see if we can read it.*/
```

CS001436_-_Using_QXmlStreamReader_to_parse_XML

```
    if(token == QXmlStreamReader::StartElement) {
        /* If it's named persons, we'll go to the next.*/
        if(xml.name() == "persons") {
            continue;
        }
        /* If it's named person, we'll dig the information from there.*/
        if(xml.name() == "person") {
            persons.append(this->parsePerson(xml));
        }
    }
}
/* Error handling. */
if(xml.hasError()) {
    QMessageBox::critical(this,
                          "QXSRExample::parseXML",
                          xml.errorString(),
                          QMessageBox::Ok);
}
/* Removes any device() or data from the reader
 * and resets its internal state to the initial state. */
xml.clear();
this->addPersonsToUI(persons);
}

 QMap<QString, QString> QXSRExample::parsePerson(QXmlStreamReader& xml) {
    QMap<QString, QString> person;
    /* Let's check that we're really getting a person. */
    if(xml.tokenType() != QXmlStreamReader::StartElement &&
        xml.name() == "person") {
        return person;
    }
    /* Let's get the attributes for person */
    QXmlStreamAttributes attributes = xml.attributes();
    /* Let's check that person has id attribute. */
    if(attributes.hasAttribute("id")) {
        /* We'll add it to the map. */
        person["id"] = attributes.value("id").toString();
    }
    /* Next element... */
    xml.readNext();
    /*
     * We're going to loop over the things because the order might change.
     * We'll continue the loop until we hit an EndElement named person.
     */
    while(!(xml.tokenType() == QXmlStreamReader::EndElement &&
            xml.name() == "person")) {
        if(xml.tokenType() == QXmlStreamReader::StartElement) {
            /* We've found first name. */
            if(xml.name() == "firstname") {
                this->addElementDataToMap(xml, person);
            }
            /* We've found surname. */
            if(xml.name() == "surname") {
                this->addElementDataToMap(xml, person);
            }
            /* We've found email. */
            if(xml.name() == "email") {
                this->addElementDataToMap(xml, person);
            }
            /* We've found website. */
            if(xml.name() == "website") {
                this->addElementDataToMap(xml, person);
            }
        }
    }
}
```

```

    }
    }
    /* ...and next... */
    xml.readNext();
}
return person;
}

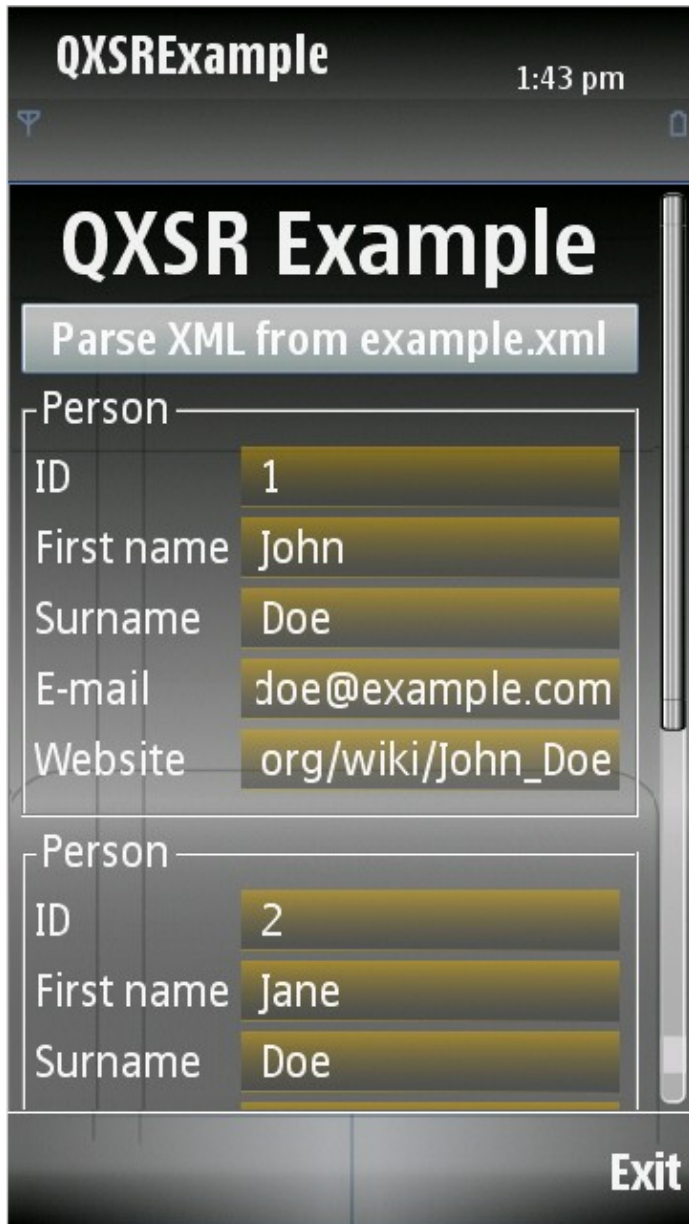
void QXSRExample::addElementDataToMap(QXmlStreamReader& xml,
                                       QMap<QString, QString>& map) const {
    /* We need a start element, like <foo> */
    if(xml.tokenType() != QXmlStreamReader::StartElement) {
        return;
    }
    /* Let's read the name... */
    QString elementName = xml.name().toString();
    /* ...go to the next. */
    xml.readNext();
    /*
     * This elements needs to contain Characters so we know it's
     * actually data, if it's not we'll leave.
     */
    if(xml.tokenType() != QXmlStreamReader::Characters) {
        return;
    }
    /* Now we can add it to the map.*/
    map.insert(elementName, xml.text().toString());
}

void QXSRExample::addPersonsToUI(QList< QMap<QString,QString> >& persons) {
    while(!persons.isEmpty()) {
        QGroupBox* personGB = new QGroupBox("Person");
        QFormLayout* layout = new QFormLayout;
        QMap<QString,QString> person = persons.takeFirst();
        layout->addRow("ID", new QLineEdit(person["id"]));
        layout->addRow("First name", new QLineEdit(person["firstname"]));
        layout->addRow("Surname", new QLineEdit(person["surname"]));
        layout->addRow("E-mail", new QLineEdit(person["email"]));
        layout->addRow("Website", new QLineEdit(person["website"]));
        personGB->setLayout(layout);
        this->_layout->addWidget(personGB);
    }
}

```

Postconditions

You are able to parse the XML and display it in the UI.



Supplementary material

- You can test `QXSRExample` with a test application. The application is available for download at [Media:QXSRExample.zip](#).

See also

- [QXmlStreamReader documentation](#)