QXmlToQObjectCreator Class Reference

[QtXmlPatterns module]

The QXmlToQObjectCreator class builds a QObject tree that corresponds to XML it receives. More...

```
#include <QXmlToQObjectCreator>
```

Inherits QAbstractXmlReceiver.

Note: All the functions in this class are reentrant.

- List of all members, including inherited members

Public Functions

- **QXmlToQObjectCreator** (const QXmlNamePool & *namePool*)
- QObject * **createdObject** () const
- QXmlNamePool **namePool** () const

- 12 public functions inherited from QAbstractXmlReceiver

Protected Functions

- virtual QObject * **createObject** (const QXmlName & *name*, QObject * *parentObject*) const
- virtual QObject * **createObject** (const QString & *name*, QObject * *parentObject*) const

---

Detailed Description

The QXmlToQObjectCreator class builds a QObject tree that corresponds to XML it receives.

QXmlToQObjectCreator is a QAbstractXmlReceiver sub-class which builds a

tree of QObject instances, receivable using createdObject(), for the XML that this class received.

For instance, if QXmlQuery output the following XML:

```
<product name="Qt">
    <pronouncedCute/>
</product>
```

and is subsequently sent to this class using QXmlQuery::evaluateTo(QAbstractXmlReceiver *), this class would conceptually receive:

```
startElement("product");
attribute("name", "Qt");
startElement("pronouncedCute");
endElement(); // For pronouncedCute
endElement(); // For product
```

After QXmlQuery::evaluateTo() returns, createdObject() would return a QObject which:

- Has the object name `product`
- Has one dynamic property named `name`, whose value is "Qt"
- Has one child object whose name is `pronouncedCute`

Example: QObject tree from XML document

Here's how to create a QObject tree from an XML document:

```
    QXmlQuery query;
    query.bindVariable("myFileName", QVariant(fileName));
    query.setQuery("doc($myFileName)");

    QXmlToQObjectCreator creator(query.namePool());
    query.evaluateTo(&creator);

    QObject *const object = creator.createdObject();
```

Extending and Customizing

Each time a QObject is created, QXmlToQObjectCreator calls the protected function createObject(), which is expected to create a QObject. It's possible to reimplement this function to return a QObject subclass.

Similarly, all the functions in the QAbstractXmlReceiver can be overriden to customize behavior. For instance, overriding characters() could change the

current behavior of ignoring text nodes, to do something which is meaningful for the application.

Mapping XML to QObjects

Since the XML data model is more elaborated than the one of QObjects, the question of how they map together naturally arise.

- Namespaces and prefixes in element and attributes are ignored and lost, since they cannot be presented using QObject::objectName(). In other words, QXmlToQObjectCreator doesn't work for XML that uses namespaces.
- Nothing is created for processing instructions
- Nothing is created for comments
- Nothing is created for document nodes
- Nothing is created for text nodes
- Nothing is created for atomic values

Member Function Documentation

QXmlToQObjectCreator::QXmlToQObjectCreator (const QXmlNamePool & *namePool*)

Creates a QXmlToQObjectCreator that uses *namePool* for converting QXmlName to QString, when for instance attribute() and startElement() is called.

The same name pool must be used between the caller and this class, otherwise behavior is undefined. Typically the name pool returned by QXmlQuery::namePool() is passed.

See also QXmlNamePool.

QObject * QXmlToQObjectCreator::createObject (const QXmlName & *name*, QObject * *parentObject*) const

`[virtual protected]`

Calls createObject(const QString &) and returns its return value. Passed is the local name in *name*. That is, the namespace is lost.

The caller guarantees that *name* is a non-null QXmlName. *parentObject* may be `null`.

QObject * QXmlToQObjectCreator::createObject (const QString & *name*, QObject * *parentObject*) const [virtual protected]

This is an overloaded member function, provided for convenience.

Creates a QObject, sets its object name to name, and returns it.

The caller guarantees that *name* is a valid `NCName`. *parentObject* may be `null`.

QObject * QXmlToQObjectCreator::createdObject () const

Returns the QObject that QXmlToQObjectCreator created.

QXmlToQObjectCreator does not own the created object.

QXmlNamePool QXmlToQObjectCreator::namePool () const

Returns the QXmlNamePool that this class is using. The name pool is set by passing it in the constructor.
