<http://blog.csdn.net/allenjiao/article/details/44963131>

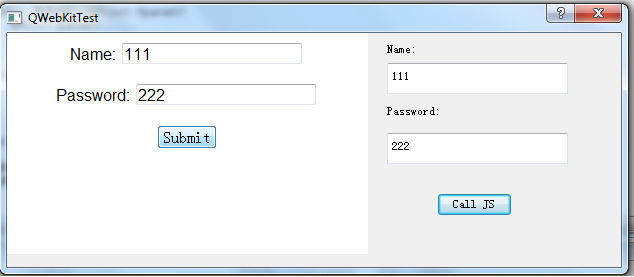
# QT QWebKit使用心得(Qwebkit与Html之间通信)

转载 2015年04月09日 18:03:01

* 9757
* [编辑](http://write.blog.csdn.net/postedit/44963131)
* 删除

最近这段时间，正在用QWebkit开发一个Application store的项目。因为需要用到第三方的数据，但后台无法直接去获取，所以需要做一个底层的模块，去获取第三方数据，然后给网页调用，因此我们选取了以WebKit为引擎的QWebkit来开发我们的项目。

在开发中，获得了一些心得体会，写下来供大家参考。为了方便说明，我特意做了一个例子。

.

                                                                           （图1）

此例子中，左边显示网页，右边是控件部分。

用户在网页的Name和Password输入框输入内容后，点击"Submit"按钮，则右边的Name和Password框则会出现网页中的内容，这就是网页调用Qt模块。

而用户在右边输入分别输入Name和Password，点击"Call JS"后，输入的信息也会显示在左边的Name和Password栏。这就是Qt调用网页。

由此我们可以看出QWebKit，即可以Qt调用网页，也可以网页调用Qt。这样我们就可以做用Qt做一些网页做不了的事情。

接下来，将阐述如何实现Qt和网页互相调用。

首先，在“图1”所示的主界面中生成WebKit控件，可以自己new或者在Design中拖入。在本例中，采取的是拖入此控件。此控件对应的类是QWebView。它可以Load和显示页面。通过此控件我们来调用网页的方法和接受网页的调用。

本主界面的类是

.h的代码：

**[cpp]** [view plaincopyprint?](http://blog.csdn.net/esonpo/article/details/9243027)

1. #ifndef QWEBKITTEST\_H
2. #define QWEBKITTEST\_H
4. #include <QDialog>
5. #include <QString>
6. #include <QWidget>
7. #include <QWebView>
8. #include <QWebFrame>
9. #include <QTextEdit>
10. #include "mywebkit.h"
12. **namespace** Ui {
13. **class** QWebKitTest;
14. }
15. **class** QWebKitTest : **public** QDialog
16. {
17. Q\_OBJECT
19. **public**:
20. **explicit** QWebKitTest(QWidget \*parent = 0);
21. ~QWebKitTest();
23. **void** setValueFromWeb(**const** QString &strName,**const** QString &strPwd);
25. **protected** slots:
26. **void** onBtnCallJSClicked();
27. **void** populateJavaScriptWindowObject();
29. **private** slots:
30. **void** on\_pushButton\_clicked();
32. **private**:
33. Ui::QWebKitTest \*ui;
34. //MyWebKit\* m\_webObj;
35. **class** MyWebKit\* m\_webObj;
36. };
38. #endif // QWEBKITTEST\_H

具体cpp代码为：

**[cpp]** [view plaincopyprint?](http://blog.csdn.net/esonpo/article/details/9243027)

1. #include "qwebkittest.h"
2. #include "ui\_qwebkittest.h"
4. QWebKitTest::QWebKitTest(QWidget \*parent) :
5. QDialog(parent)
6. {
7. m\_webObj = **new** MyWebKit(**this**);
8. ui->setupUi(**this**);
9. ui->webView->setUrl((QUrl("qrc:/test.html")));
10. connect(ui->webView->page()->mainFrame(), SIGNAL(javaScriptWindowObjectCleared()), **this**, SLOT(populateJavaScriptWindowObject()));
12. }
14. QWebKitTest::~QWebKitTest()
15. {
16. **delete** ui;
17. }
19. **void** QWebKitTest::setValueFromWeb(**const** QString &strName, **const** QString &strPwd)
20. {
21. ui->textEdit->setText(strName);
22. ui->textEdit\_2->setText(strPwd);
23. }
25. **void** QWebKitTest::onBtnCallJSClicked()
26. {
27. QString strVal = QString("callfromqt(\"%1\",\"%2\");").arg(ui->textEdit->toPlainText()).arg(ui->textEdit\_2->toPlainText() );
28. ui->webView->page()->mainFrame()->evaluateJavaScript(strVal);
29. }
31. **void** QWebKitTest::populateJavaScriptWindowObject()
32. {
33. qDebug()<<"populateJavaScriptWindowObject";
34. ui->webView->page()->mainFrame()->addToJavaScriptWindowObject(QString("mywebkit"),m\_webObj);
35. }
37. **void** QWebKitTest::on\_pushButton\_clicked()
38. {
39. onBtnCallJSClicked();
40. }

Qt使用的是向网页注册一个QObject对象，通过这个对象，网页可以通过这个调用这个对象的方法来实现调用底层的逻辑。

以下是本例中的注册对象的声明，

**[h]** [view plaincopy](http://blog.csdn.net/superfpe/article/details/6641360)

1. **class** MyWebKit : **public** QObject
2. {
3. Q\_OBJECT
5. **public**:
6. MyWebKit(QObject \*parent=0);
7. ~MyWebKit();
9. **public** slots:
11. **void** onCall(QString strName,QString strPwd);
13. **private**:
15. };

cpp

#include "MyWebKit.h"

MyWebKit::MyWebKit(QObject \*parent):

QObject(parent)

{

;

}

MyWebKit::~MyWebKit()

{

;

}

void MyWebKit::OnDataCallback(QString strBussid,QString strType,QString strIndex,QString strName,QString strValue)

{

//CCrEnvironmengDlg \*pMain = (CCrEnvironmengDlg\*)parent();

//pMain->setValueFromWeb(strBussid,strType);

}

html部分代码

**[html]** [view plaincopy](http://blog.csdn.net/superfpe/article/details/6641360)

1. **<html>**
2. **<head>**
3. **<title>**Login page**</title>**
5. **<script** language="javascript" **>**
6. function calltoqt()
7. {
8. var nameArray;
9. nameArray = document.getElementsByName("username");
10. var pwdArray;
11. pwdArray = document.getElementsByName("userpwd");
12. window.mywebkit.onCall(nameArray[0].value,pwdArray[0].value);
14. }
16. function callfromqt(name,pwd)
17. {
18. var nameArray;
19. nameArray = document.getElementsByName("username");
20. var pwdArray;
21. pwdArray = document.getElementsByName("userpwd");
22. nameArray[0].value = name;
23. pwdArray[0].value = pwd;
24. }
26. **</script>**
28. **</head>**
29. **<body>**
30. **<p** align="center"**>**Name:
31. **<input** type="text" name="username" **/>**
32. **</p>**
34. **<p** align="center"**>**Password:
35. **<input** type="text" name="userpwd"  **/>**
36. **</p>**
37. **<p** align="center" **>**
38. **<input** type="button" value="Submit" onclick="calltoqt()" **/>**
39. **</p>**
40. **</body>**
41. **</html>**

当然，evaluateJavaScript的参数不一定只是javascript方法，同样一段javascript代码也是可以的。比如以上功能也可以这样调用

**[cpp]** [view plaincopy](http://blog.csdn.net/superfpe/article/details/6641360)

1. ui.webView->page()->mainFrame()->evaluateJavaScript(" var nameArray; \
2. nameArray = document.getElementsByName(\"username\"); \
3. var pwdArray; \
4. pwdArray = document.getElementsByName(\"userpwd\"); \
5. nameArray[0].value = \"andrei\"; \
6. pwdArray[0].value = \"123456\";" );