【Qt】Web与本地应用的混合开发

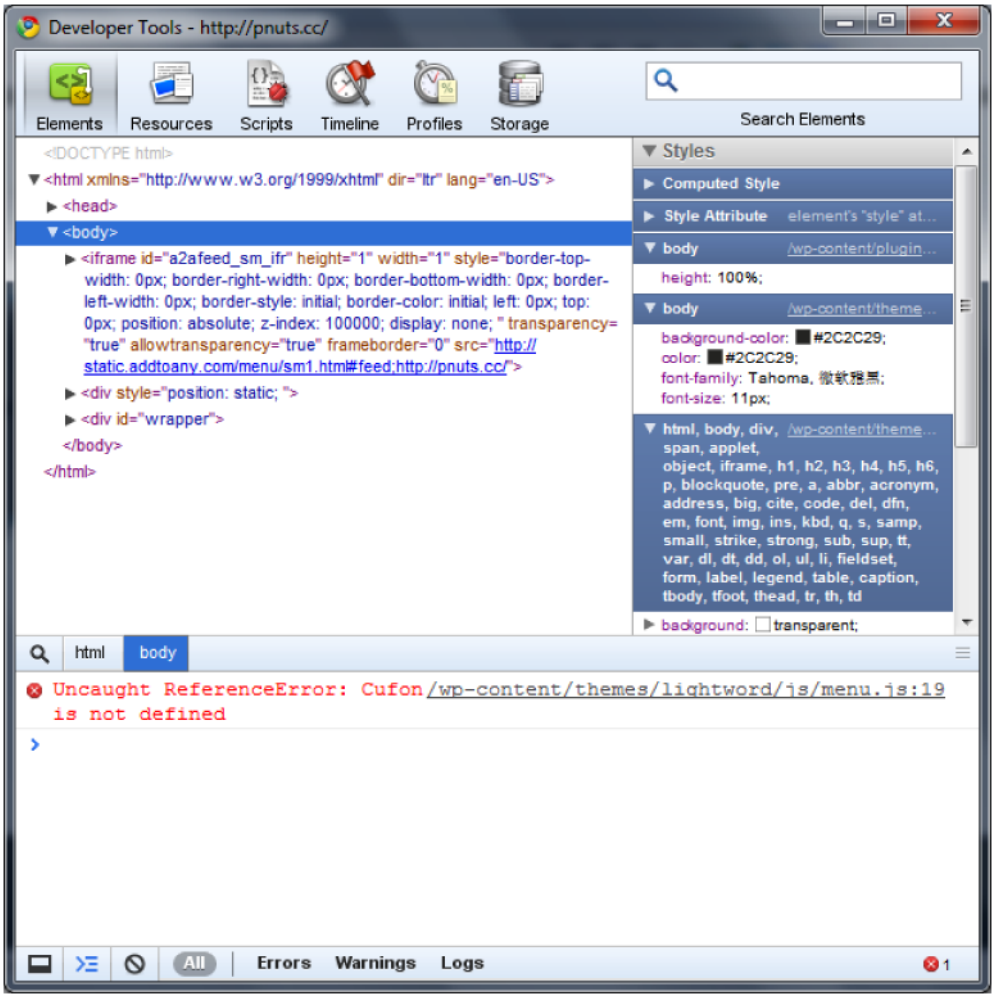
--Excerpt--

本文介绍了在Qt中如何开发Web本地混合应用，以及如何让js与c++双向调用。附带几个简单实例。

接触Qt也就两个星期多吧，所以文章中难免有幼稚和错误的地方，请各位不吝赐教。

个人认为标记语言描述的界面是界面开发的一个发展趋势。WPF、Java FX，当然也少不了Html。

基于Html的界面在开发效率，可移植性上都十分有优势，所以也被很多程序采用，只是我们平时没注意到而已。比如：



Chrome的网页调试工具本身就是一个网页，不信你可以右键点击，选查看源代码。

其它的还有像ServU和VMware Server等，懒的截图了。

下面进入正题，[illegible] QT [illegible] Web [illegible]。

因为QT中集成了Webkit，拿来做这种还是一件相当happy的事（什么？为什么要用Webkit？你想自己写一个浏览器引擎不成？）

Pnuts CC

❤ Miku ❤

Tag Cloud

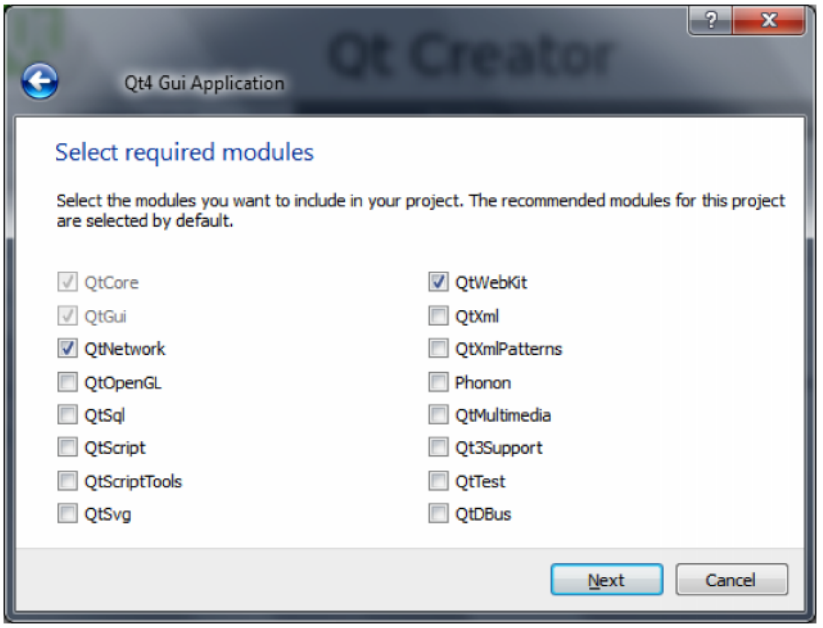
LuckyStar Flash Ubuntu Blog JavaScript Light Novel ACG Wordpress LilyBBS Miku Life Funny Algorithm Movie Project Linux Feeling

Categories

Select Category

Recent Posts

- 写在FF9十周年纪念
- Poooooooocky
- Happy Birthday To Me
- MO – Manga Online
- Php My Manga



新建一个Qt Gui项目，记得选上QtWebKit和QNetwork（这个后面要用到）。

写如下一段代码，一个浏览器就做成了：

```
#include <QtGui/QApplication>
#include <QWebView>
#include <QMainWindow>

int main(int argc, char *argv[])
{
    QApplication a(argc, argv);
    QMainWindow window;
    QWebView view(&window);
    view.setGeometry(0, 0, 600, 400);
    view.setUrl(QUrl("http://pnuts.cc"));
    window.show();
    return a.exec();
}
```

如图:

QWebView有两种方法可以用来设定要显示的内容，一个是setUrl方法，一个是setContent方法。这个很简单，试一下就会，不多说了。

不过对这两种方法会有两种不同的开发方式：

setContent 方式，需要手动将网页代码生成出来放到QWebView中，网页中的元素（如：图片,样式,脚本）就只能采用"file:///"的方式了。

setUrl 方式 更用技术含量一点，可能需要自己做一个简单的Http服务器，（不考虑效率优化的话也没那么难），然后监听本地端口，掉用QWebView的setUrl(QUrl(http://127.0.0.1:XXXX))就可以了。

我用了一种更省事的方法：直接把要显示的html放到apache上去了。这样连php也可以用了，哈哈哈哈~（咦，砖头？）

显示出来并不难，最主要的是如何同界面双向交互，又不是asp，总不能没点按钮就刷一次页面吧。

最容量想到的是传统Web开发的中常用的Ajax，不过就有两个缺点：

一个是要监听本地端口，第二个更致命，Ajax不是双向的，Server向Client发消息就不行了。

下面说一种更好的方法，实现js与C++的双向调用。

js调用c++方法

首先要将一个C++的对象“送”到js中，js拿到这个对象以后就可以像其它对象一样，自由的调用它的方法了。

这一步有两种实现方式：

1. 在网页中插入控件

JS端: 在网页中插入下面一段代码：

```
<object type="application/x-qt-plugin" id="qt"></object>
```

之后就可以通过document.getElementById('qt');获取这个对象，并调用方法了。

C++端:

首先要自定义一个类继承自QWebPage，在构造函数中加入如下一句话开起plugin的支持。

```
settings()->setAttribute(QWebSettings::PluginsEnabled, true);
```

然后重载QWebPage中的如下方法（protected的）

```
virtual QObject *createPlugin(const QString &classid, const QUrl &url, const QStringList &paramNames, const
QStringList &paramValues);
```

方法的返回值就是要传递给js的对象。

注意：返回值要是一个QWidget *类型的，所以可能还要自己自己写一个Widget，写好后，返回的QWidget中的所有public slots在js中都是可见的方法。

例子如下，有点长，不过很简单：

MyWidget.h

```
#ifndef MYWIDGET_H
#define MYWIDGET_H

#include<QWidget>
#include<QWebPage>
#include<QWebFrame>

class MyWidget :public QWidget {
    Q_OBJECT
private:
    QWebPage *page;
public:
    MyWidget(QWebPage *page) : page(page) { }
public slots:
    void func(QString arg) {
        this->page->mainFrame()->evaluateJavaScript("document.body.innerHTML += '" + arg + "';");
    }
};

#endif // MYWIDGET_H
```

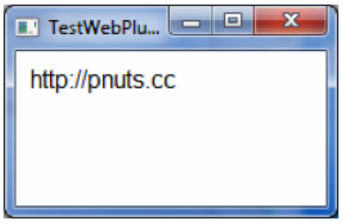
MyPage.h

```
#ifndef MYPAGE_H
#define MYPAGE_H

#include<QWebPage>
#include<QWebFrame>
#include"MyWidget.h"

class MyPage : public QWebPage {
    Q_OBJECT
public:
    MyPage(QObject *parent = 0) : QWebPage(parent) {
        settings()->setAttribute(QWebSettings::PluginsEnabled, true);
    }
protected:
    QObject *createPlugin(const QString &classid, const QUrl &url, const QStringList &paramNames, const
QStringList &paramValues){
        return new MyWidget(this);
    }
};

#endif // MYPAGE_H
```

main.cpp

```
#include <QtGui/QApplication>
#include <QMainWindow>
#include <QWebView>
#include <QWebPage>
#include <QWebFrame>
#include "MyPage.h"

int main(int argc, char *argv[])
{
    QApplication a(argc, argv);
    QMainWindow window;
    QWebView view(&window);
    MyPage page;
    view.setPage(&page);
    view.setGeometry(0, 0, 600, 400);
    // Object
    QString content("<object type='application/x-qt-plugin' height='1' width='1' id='qt'></object>");
    // JS Function f() : Invoke the 'func' function
    content += "<script>document.getElementById('qt').func('http://pnuts.cc');</script>";
    view.setContent(content.toAscii());
    window.show();
    return a.exec();
}
```

运行结果如下：



2. 用QWebFrame的addToJavaScriptWindowObject 方法

相比上一个方法，个人推荐这种方法。因为上一个在Linux下遇到过很诡异的问题。

例子是最好的说明方式，于是再给出一下例子：

要点：addToJavaScriptWindowObject 的第一个参数是对应的js中的变量，第二个参数是QObject 或者是QWidget

MyObject.h

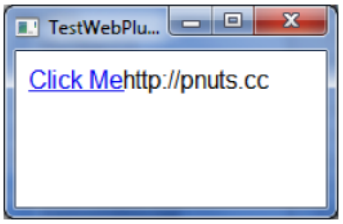
```
#ifndef MYOBJECT_H
#define MYOBJECT_H

#include<QObject>
#include<QWebPage>
#include<QWebFrame>

// !! ATTENTION !! : The object do NOT need to inherit from QWidget anymore.
class MyObject :public QObject {
    Q_OBJECT
private:
    QWebPage *page;
public:
    MyObject(QWebPage *page) : page(page) { }
public slots:
    void func(QString arg) {
        this->page->mainFrame()->evaluateJavaScript("document.body.innerHTML += '" + arg + "';");
    }
};

#endif // MYOBJECT_H
```

main.cpp

```
#include <QtGui/QApplication>
#include <QMainWindow>
#include <QWebView>
#include <QWebPage>
#include <QWebFrame>
#include "MyObject.h"

int main(int argc, char *argv[])
{
    QApplication a(argc, argv);
    QMainWindow window;
    QWebView view(&window);
    QWebPage page;
    view.setPage(&page);
    view.setGeometry(0, 0, 600, 400);
    MyObject obj(&page);
    page.mainFrame()->addToJavaScriptWindowObject("qt", &obj);
    QString content("<script>function f() { qt.func('http://pnuts.cc'); }</script>");
    content += "<a href='javascript:f()'>Click Me</a>";
    view.setContent(content.toAscii());
    window.show();

    return a.exec();
```

```
24 }
```

至于为什么要用个链接点一下再显示而不是直接执行，自己去想，想不清再来问。



C++调用js代码

依旧是两种方法。

1. evaluateJavaScript

这个超简单了，上面的例子中就用到了。不多说了。

2. connect

这个更符合Qt的风格一点。首先用上一部分介绍的两种方法中的任意一种将一个C++对象"送"到js中去。然后调用这个对象的connect方法，将自己的signals和js方法进行bind。

再放个例子吧，不过例子中竟然用evaluateJavaScript来bind。。呵呵。。。

当QWebView加载好后，绑定MyObject的Miku Signal到js本地方法f，再触发Miku Signal。

MyObject.h

```
01 #ifndef MYOBJECT_H
02 #define MYOBJECT_H
03
04 #include<QObject>
05 #include<QWebPage>
06 #include<QWebFrame>
07
08 class MyObject :public QObject {
09     Q_OBJECT
10 private:
11     QWebPage *page;
12 public:
13     MyObject(QWebPage *page) : page(page) { }
14 signals:
15     void Miku();
16 public slots:
17     void viewLoad() {
18         this->page->mainFrame()->evaluateJavaScript("qt.Miku.connect(f);");
19         this->Miku();
20     }
21 };
22
23 #endif // MYOBJECT_H
```

main.cpp

```
01 #include <QtGui/QApplication>
02 #include <QMainWindow>
03 #include <QWebView>
04 #include <QWebPage>
05 #include <QWebFrame>
06 #include "MyObject.h"
07
08 int main(int argc, char *argv[])
09 {
10     QApplication a(argc, argv);
11     QMainWindow window;
12     QWebView view(&window);
13     QWebPage page;
14     view.setPage(&page);
15     view.setGeometry(0, 0, 600, 400);
16     MyObject obj(&page);
17     QObject::connect(&view, SIGNAL(loadFinished(bool)), &obj, SLOT(viewLoad()));
18     page.mainFrame()->addToJavaScriptWindowObject("qt", &obj);
19     QString content("<SCRIPT>function f() { document.body.innerHTML += 'http://pnuts.cc'; }</SCRIPT>");
20     view.setContent(content.toAscii());
21     window.show();
22     return a.exec();
23 }
```

[illegible]Qt[illegible]

最后放几个自己Linux作业（本来也是因为linux作业才接触的Qt）的截图吧：

[illegible]来了


Talker
Talker
用户名：
密　码：
记住密码
自动登陆
注 册
登 陆

（登陆窗口）


Talker
cider
私の恋を悲劇のジュリエットにしないで 14:30
14:31 ここから连れ出して…
そんな気分よ 14:31
14:31 パパとママにおやすみなさい
せいぜい いい梦をみなさい 14:31
せいぜい いい梦をみなさい
发送

（聊天窗口1）

（聊天窗口2）

（隐藏版的HX主题和鲜花主题）

- EF -

P.S.

1. MikuAppend的第五首试听曲放出来了：chocolat(D.B. used sweet) by chiquewa，个人认为5首中这首最好听了，唱的好有味道。不过也成熟了好多.....5555.....
2. 强烈鄙视爆初音吧的SB蓝猫教的SB们，见过赛脸的，没见过给脸不要脸的。（真是的，竟然逼我骂人。上次骂人还是高一吧。如果不是Miku也懒得搭理你们。）

3. 听说《文学少女》要动漫化，急忙找来轻小说补习中。
4. Tiwtter单方面封了我米国服务器的ip，交涉无果中。
5. 忽然发现aoi sola的名字很不错，以前都没注意，aoi发音和“青色的”发音相同，sola是天空，合起来就是“蓝色的天空”。
6. [illegible]~

Tagged as: Tutorial Leave a comment

Comments (7) Trackbacks (0) (subscribe to comments on this post)

LIGHT
April 13th, 2010 - 22:13

文学少女，轻小说，必看。
改编的剧场版，可以不看。

(REPLY)

Pnuts
April 14th, 2010 - 12:57

貌似已经有漫画了。

(REPLY)

LIGHT
April 14th, 2010 - 23:36

漫画也可以不看..

(REPLY)

Shiaron
April 14th, 2010 - 00:44

啊　居然没有沙发了　你博客怎么这么卡了。。。

(REPLY)

Pnuts
April 14th, 2010 - 12:56

可能是这篇里要加高亮的代码太多了。

(REPLY)

nic
May 2nd, 2010 - 19:19

博主多大了，漫画型IT人才？

(REPLY)

Pnuts
May 2nd, 2010 - 22:24

有个更专业一点的名词叫“技术宅”。

(REPLY)

Leave a comment

Name (required)

Mail (will not be published) (required)

Website

Submit

Notify me of follow-up comments via e-mail

« 雨天，推荐两首Miku的歌　　All About LilyStuido »

Go to top ↑