第33篇 (网络) WebKit初识

代码地址：https://github.com/Lornatang/QtStartQuicklyTutorial/tree/main/Network09

一、简单 @应用

下面我们来实现一个可以打开特定网页的程序。新建空的Qt项目，在`pro`项目文件中添加一行代码：`QT += webkit`，然后向项目中添加一个`main.cpp`文件，并在其中添加如下代码：

```
#include <QWebView>#include <QApplication>int main(int argc, char* argv[])
{
    QApplication a(argc, argv);
    QWebView view;
    view.load(QUrl("http://www.qter.org"));
    view.show();
    return a.exec();
}
```

要使用WebKit，就要先添加`webkit`模块。`QWebView`是`QtWebKit`模块主要的窗体部件，它可以在各种应用程序中用来显示Internet上的网页内容。`QWebView`作为一个窗口部件，可以嵌入到窗体或者图形视图部件中。

`QWebView`用来显示Web页面，每个`QWebView`实例都包含一个`QWebPage`对象。`QWebPage`提供了对一个页面的文档结构的访问，描述了如框架（frame）、访问历史记录和可编辑内容的撤销/重做栈等特色。每一个`QWebPage`都包含一个`QWebFrame`对象作为它的主框架。在HTML中的每一个单独的框架都可以使用`QWebFrame`来表示，这个类包含了到JavaScript窗口对象的桥梁，而且可以进行绘制。在QWebPage的主框架中可以包含很多的子框架。

HTML文档中单独的元素可以通过DOM JavaScript接口进行访问，在`QtWebKit`中与这个接口等价的接口由`QWebElement`来表示。`QWebElement`对象可以使用`QWebFram`e的`findAllElement()`和`findFirstElement()`函数来获取。一般的网页浏览器的特色设置都可以

通过`QWebSettings`类来配置，可以通过默认设置为所有的`QWebPage`实例提供默认值。单独的属性可以使用页面指定的设置对象进行重写。

二、扩展应用

下面再来看一个可以随意更改网址并且可以显示网站logo的例子。新建Qt Gui应用，项目名称为`webview`，类名和基类保持`MainWindow`和`QMainWindow`不变。完成后向`webview.pro`文件中添加`QT += webkit`一行代码，并按下`Ctrl + S`保存该文件。

1．下面到`mainwindow.h`文件中，先添加头文件：

```
#include <QWebView>#include <QLineEdit>
```

然后添加槽的声明：

```
protected slots:
    void changeLocation();     // 改变路径
    void setProgress(int);     // 更新进度
    void adjustTitle();        // 更新标题显示
void finishLoading(bool);  // 加载完成后进行处理
       再添加对象和变量定义：
QWebView *view;
QLineEdit *locationEdit;
int progress;
```

2．下面到`mainwindow.cpp`文件中，在构造函数中添加如下代码：

```
progress = 0;
view = new QWebView(this);
setCentralWidget(view);
resize(800, 600);

// 关联信号和槽
connect(view, SIGNAL(loadProgress(int)), this, SLOT(setProgress(int)));
connect(view, SIGNAL(titleChanged(QString)), this, SLOT(adjustTitle()));
connect(view, SIGNAL(loadFinished(bool)), this, SLOT(finishLoading(bool)));
locationEdit = new QLineEdit(this);
connect(locationEdit, SIGNAL(returnPressed()), this, SLOT(changeLocation()));

// 向工具栏添加动作和部件
ui->mainToolBar->addAction(view->pageAction(QWebPage::Back));
ui->mainToolBar->addAction(view->pageAction(QWebPage::Forward));
ui->mainToolBar->addAction(view->pageAction(QWebPage::Reload));
```

```
ui->mainToolBar->addAction(view->pageAction(QWebPage::Stop));
ui->mainToolBar->addWidget(locationEdit);

// 设置并加载初始网页地址
locationEdit->setText("http://www.baidu.com");
view->load(QUrl("http://www.baidu.com"));
```

当`QWebView`开始加载时，会发射`loadStarted()`信号；而每当一个网页元素（例如一张图片或一个脚本等）加载完成时，都会发射`loadProgress()`信号；最后，当加载全部完成后，会发射`loadFinished()`信号，如果加载成功，该函数的参数为`true`，否则为`false`。可以使用`title()`来获取HTML文档的标题，如果标题发生了改变，将会发射`titleChanged()`信号。

3．下面添加那几个槽的定义：

```
void MainWindow::changeLocation()
{
    QUrl url = QUrl(locationEdit->text());
    view->load(url);
    view->setFocus();
}
void MainWindow::setProgress(int p)
{
    progress = p;
    adjustTitle();
}
void MainWindow::adjustTitle()
{
    if ( progress <= 0 || progress >= 100) {
       setWindowTitle(view->title());
    } else {
setWindowTitle(QString("%1 (%2%)").arg(view->title()).arg(progress));
    }
}
void MainWindow::finishLoading(bool finished)
{
    if (finished) {
       progress = 100;
       setWindowTitle(view->title());
    } else {
       setWindowTitle("web page loading error!");
    }
}
```

下面运行程序，效果如下图所示：


Qter开源社区
http://www.qter.org
Qter
开源社区
开源 · 共享 · 自由
首页
论坛
问答
书籍
Qter开源社区全新改版，开放测试
期待你的加入！