PHP V5.3 在 Unicode 方面有何新特性？

发布者：IBM 日期： 2010-01-28 00:00:00 浏览次数：46 (共有0条评论) 查看评论 | 我要评论

级别： 初级

方 正, 软件工程师, IBM

2010 年 1 月 28 日

由两篇文章组成的系列文章主要阐述如何在嵌入式 Linux 智能设备的应用程序中增加 Web 支持。第 1 部分介绍了如何设备上提供常规 Web 功能的支持。本文是第 2 部分，将重点介绍如何让在嵌入式设备上运行的 Web 程序能支持设备本身特有的功能。本文分别以四种应用场景为例，介绍如何通过修改浏览器内核代码来实现设备本地应用和 Web 结合的功能。

Web 与本地应用的关联

虽然在嵌入式 Linux 智能设备中采用 Web 支持已经解决了很多问题，但是还有一些和设备相关的特殊功能是 Web 支持不能提供的。比如广告机中的音视频播放功能，条码扫描机的模式识别功能，还有与某种外设的通信等。这些并不是 HTML 和浏览器的标准所包含的，而是需要本地应用的支持。既然我们希望使用 Web 和 B/S 等技术来实现我们的应用，那么这些本地应用功能也应该由 Web 来控制。比如说广告机的视频播放，实际的播放是由本地应用实现的，但是什么时候在什么位置播放什么视频应该由 Web 来决定。并且广告页面内容的编辑也应该在网页的 HTML 中体现，而不需要另外一套播放控制机制。

但是想要由 Web 来控制本地应用存在一个问题，这些本地应用的调用没有一种统一的机制。有的可能通过驱动，有的可能是通过 I2C、串口的通讯口，有的可能是第三方提供的库，还有的可能是与其他进程的通信。可以说，除了他们大多用 C/C++ 语言进行开发之外，几乎没有什么共同点。

那么现在我们要解决的问题就是，当 QWebView 渲染一个网页的时候，如何让我们在网页里编写的一些特定的 HTML 能和我们的 C/C++ 代码关联起来。幸运的是，Qt 封装的 WebKit 提供了多种方法使我们可以很好实现这个关联。接下来，我们会以几种应用场景为例来讨论 Web 和本地应用关联的几种实现方法。

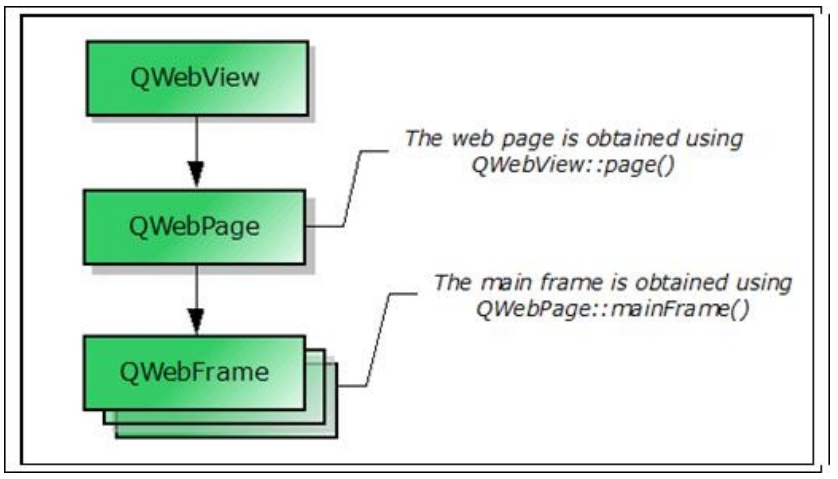
截取 request 的方法

首先我们介绍第一种应用场景：某嵌入式智能设备需要实现下面的功能，用户点击网页上“更新”的链接，设备就会下载指定的 Firmware 并且进行更新。

为了实现这个功能，客户端的浏览器需要在用户点击了某个特定的 Link 之后，启动系统的更新过程。包括获取最新 Firmware 的地址，进行下载，最后更新设备。Firmware 的更新过程和设备硬件相关，标准浏览器不能实现这个功能，因此我们必须“截获”用户的这个请求，然后使用本地代码来完成整个更新过程。

为了实现截获用户的这个 HTML request，我们先分析一下 QWebView 的结构。

图 1. QWebView 的结构图



QWebView 使用 QWebPage 来实现页面，QWebPage 使用 QWebFrame 来实现页面元素。当页面发出一个 Navigation 的 request 时，QWebPage 会来进行处理。这个时候有一个函数会被调用：

bool QWebPage::acceptNavigationRequest (QWebFrame *frame, const QWebNetworkRequest &request, QWebPage::NavigationType type)

这个函数会在发生 Navigation Request 的时候获取到触发事件的页面元素、request 内容和类型。如果函数如果返回 false，浏览器将忽略这个 request。

我们可以从 QWebPage 派生一个子类，重写 acceptNavigationRequest，在发现特定 request 内容的时候，做出自己的处理。假设目标地址是 http://xxxx.com/update/Firmware.bin，实现如下：

清单 **6. acceptNavigationRequest** 函数的定义和实现

```
class QMyWebPage : public QwebPage
{
protected:
bool acceptNavigationRequest ( QWebFrame *frame, const QWebNetworkRequest &request,
QWebPage::NavigationType type );

...
...

};

QMyWebPage::acceptNavigationRequest ( QWebFrame *frame,
const QWebNetworkRequest &request, QWebPage::NavigationType type )
{
   if( type == QWebPage::NavigationTypeFormSubmitted )
   {
       QString str = url = request.url().path();
       // 如果是特定的目标
       if( str == “http://xxxx.com/update/Firmware.bin”  )
       {
           // 从 link 中获取 Firmware 地址
           get Firmware addr from path
           // 下载 Firmware
           download Firmware
           // 更新设备 Firmware
           update Firmware
           // 返回 false 让浏览器不再处理这个 request
           return false;
       }
   }
   return QWebPage::acceptNavigationRequest ( frame, request , type );
}
```

上面实现部分中获取、下载和更新 Firmware 部分用说明性文字来表示，不是真实的实现代码，用户可以根据自身的需求改写这部分本地代码。

除了实现具体功能之外，我们还需要让 QMyWebPage 被 QWebView 使用。这是通过 QWebView 的 setPage 调用实现的，可以在构造 QWebView 实例的时候加入：

```
QWebView* Webview = new QWebView ( this );
QMyWebPage* page = new QMyWebPage ();Webview -> setPage ( page );  // 让 WebView 使用我们的 QwebPage
```

至此，我们实现了当页面发生了点击 http://xxxx.com/update/Firmware.bin 的时候，截取了这个 request，并让我们的本地代码能被适时的调用运行。

acceptNavigationRequest 还可以被用在另外一种场合，某些网站会根据设备的 mac 地址决定是否提供下载服务，让设备在请求下载链接的时候，要求其在头信息里提供 mac 地址。我们注意到 acceptNavigationRequest 的参数里有 QWebNetworkRequest 的变量，这个类实际上就包含了头信息，虽然在这个变量在这里是一个不可更改的引用，但是我们可以保留这个信息，复制一份，在头信息里加入 mac 信息，然后让 QWebView 主动进行一次下载请求，从而实现在头信息里添加自定义内容的功能。

在页面中执行自定义的 JavaScript 的方法

接着我们介绍另外一种应用场景：手持条码机对准货物的条码，按键扫描之后，该货物的信息立刻在条码机上显示出来。

这个功能是一个很典型的网页查询应用，我们可以假设条码是被手工输入到网页上的编辑框，然后 submit 一个请求，服务器返回该条码表示的货物信息。所以，如果在按键扫描之后，条码号能被填入网页上的编辑框并且触发一个 submit，这个功能就可以实现。

Qt 封装的 WebKit 可以在已加载的页面中插入执行用户自定的 JavaScript，这是通过 QWebFrame 的 evaluateJavaScript 接口来实现。

```
QVariant QWebFrame::evaluateJavaScript ( const QString& scriptSource );
```

下面我们通过几个例子来演示如何执行 JavaScript。

假设我们的页面中有一个编辑框，名称为“code”，它的旁边还有一个按钮名称为“query”。扫描机对准条形码之后，用户按下一个按键，触发了 Qt 程序窗体 form 中的一个消息响应函数，在消息响应函数中通过如下的语句可以设置编辑框中的内容：

清单 **7.** 设置编辑框内容的代码实现

```
QWebFrame *frame = form.WebView->page()->mainFrame();
QString code = getScanCode ();    // 调用扫描条形码的功能，需要自己实现
QString js = QString ("document.getElementById('code').value =\"%1\";" ).arg(code) );
frame->evaluateJavaScript ( js );
```

接下来可以用下面语句来实现触发 query 按钮：

清单 **8.** 触发 **query** 按钮的代码实现

```
QWebFrame *frame = form.WebView->page()->mainFrame();
QString js =
QString ( "document.getElementById('query').submit();" );
frame -> evaluateJavaScript ( js );
```

除了可以设置网页上编辑框内容外，我们还可以通过下面的语句获取编辑框中的内容：

清单 **9.** 获取编辑框内容的代码实现

```
QWebFrame *frame = form.WebView->page()->mainFrame();
QString s1 = frame->evaluateJavaScript ("document.getElementById ('code').name" );
```

这样就解决了条码机的货物查询功能所碰到的问题。我们可以让页面随时运行我们自定义的 JavaScript，这个功能将发挥非常大的作用。它实际上解决了在由设备进行主动触发的的应用模式下，本地代码和网页进行配合的问题。但是这个方法只能用于特定的网页，因为我们自己插入的 JavaScript 必须与网页上运行环境匹配。

自定义 JavaScript 扩展的方法

接着我们介绍第三种应用场景：我们先考虑这样一个问题，如果页面的 JavaScript 代码中需要得到本地应用的支持怎么办？比如一个机顶盒软件需要配置本地网络（这种应用原本都是编写本地应用程序实现的，但是既然我们讨论采用 Web 方法来代替原有开发模式，就需要考虑如何在 Web 上实现）。首先，页面需要获取当前网络设置方式，IP 地址、子网掩码、DNS 等。在网页上的编辑框、下拉框等控件内显示，用户做了一些配置之后，点击网页上的“确定”按钮，这些配置信息就生效了。

从页面的角度来说，这些都需要用 JavaScript 代码来实现，那么我们就需要让 JavaScript 代码能和本地代码关联起来。Qt 支持自定义的 JavaScript 扩展，也就是说用户可以自己在 Qt 中定义一个对象，编译到 WebKit 中。页面中的 JavaScript 脚本可以直接生成这个对象并且调用其方法。Qt 在目录 \src\3rdparty\WebKit\WebCore\bridge\ 中提供了一个 demo。测试代码在文件 testqtbindings.cpp 中。我们可以参考他的方法的编写自定义的类：

清单 **10.** 自定义类的实现代码

```
class MyObject : public QObject
{
Q_OBJECT
//   定义属性和函数的关联
   Q_PROPERTY ( QString ip READ ip WRITE setIp )
public:
   MyObject (){}

   QString ip ()
   {
       // 以字符串方式返回 IP 地址的实现
   };
   void setIp( QString )
   {
       // 设置 IP 地址的实现
   };
};

// 通过如下的代码来生成对象实例：
MyObject* myObject = new MyObject;
```

然后用下面的方法实现对象 myObject 和 JavaScript 中的对象 myInterface 的关联：

清单 **11. C++** 对象和 **JavaScript** 对象的关联代码

```
Global* global = new Global ();
RefPtr<Interpreter> interp = new Interpreter ( global );
ExecState* exec = interp->globalExec ();
// 实现 C++ 对象和 JavaScript 对象的关联
global->put ( exec, Identifier("myInterface" ), Instance::createRuntimeObject (
                                Instance::QtLanguage, (void*)myObject) );
```

将 MyObject 的定义在 QWebFrame.h 中声明，并且将清单 11 中的代码加入到 QWebFrame 的构造函数中（QWebFrame.cpp）。QWebFrame.h 和 QWebFrame.cpp 两个文件在目录 src\3rdparty\WebKit\WebKit\qt\Api 下。重新编译 WebKit 模块之后，在网页中就可以使用 myInterface 来调用对象 myObject 的方法了。调用的 JavaScript 代码如下：

需要获取 ip 的时候：

```
str = myInterface.ip;
```

需要设置 ip 的时候：

```
myInterface.Ip = str;
```

上面的代码只是 ip 地址获取和设置示例，其他类似掩码、dns 之类可以使用类似的方法。

任何嵌入式智能设备的用 C/C++ 实现的本地功能，都可以通过上述方法让 JavaScript 来进行调用。这种扩展能让浏览器来解释执行任何我们想要的功能，几乎让 Web 和本地代码的结合完全扫除了障碍。将

实现具体功能的本地代码封装成为库和模块，然后由 Web 来进行上层架构和耦合，这将大大降低嵌入式 Linux 智能设备的软件开发难度。特别是对于经常要更新功能的应用，以前需要刷新 Firmware，而现在只要更新远程服务器上的网页就可以了。

编写 WebKit plugin 的方法

除了自定义 JavaScript 对象之外，有时候我们还会用到自定义的网页元素。在 PC 上，最典型的网页元素就是 FLASH。FLASH 并不是 HTML 标准，但是可以用插件的方式让浏览器对这些特定的标签进行解释。

最后一种应用场景：以广告机为例，之前我们提到过广告机的视频播放功能。不同平台播放视频的方式都是不同的，而网页也没有像定义图片一样定义视频播放的标准，因此广告机作为区别于 PC 的嵌入式设备，其视频播放功能必须用本地代码来实现。当我们用网页方式来组织广告机的屏幕，视频部分就应该像图片、文字一样，成为网页中的一个部分，可以通过 HTML 的定义来控制。HTML 提供了标签“object”，来方便实现一些特别的对象。比如：

```
< object data="yahtzee.gif" type="image/gif" title="A Yahtzee animation"
                                width=200 height=100 >
```

如果我们的浏览器支持我们用类似方法在网页中插入一个自定义对象，那么这个问题就可以得到解决。

Qt 支持在 WebKit 中添加自定义的插件。在文件 FrameLoaderClientQt.cpp 中的函数 FrameLoaderClientQt::createPlugin 中，可以找到如下的代码片段：

清单 **12. FrameLoaderClientQt.cpp** 代码片段

```
if ( mimeType == "application/x-qt-plugin"
   || mimeType == "application/x-qt-styled-widget" )
{
       object = m_WebFrame->page()->createPlugin( classid, qurl, params, values );
```

通过上面的代码，可以看出如果 HTML 中一个 object 将自己的 type 设置为 application/x-qt-plugin 或者是 application/x-qt-styled-widget，Qt 则会识别并要求 QWebPage 来创建插件，其方式就是调用 QWebPage 的 createPlugin 函数，函数定义如下：

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

我们设计以下的 HTML 来标识我们的对象：

```
<object type="application/x-qt-plugin" classid="VideoPlayer" width=800
                    height=600 file="kfc.avi" ></object>
```

我们设定了在网页中插入一个 VideoPlaye 的对象，并且设定了宽高、要播放的文件等参数。因为我们设定了这个对象的 type 为 application/x-qt-plugin，所以当浏览器碰到这段 HTML 代码时，会调用到 QWebPage 的 createPlugin 功能。这个函数被要求返回一个窗体。而这个窗体会被当成一个标准网页对象，和编辑框、下拉框等一样被嵌入到 Web 页面中。

我们先从 QWebPage 中派生自己的对象，实现 createPlugin：

清单 **13. createPlugin** 函数的实现

```
class QMyWebPage : public QWebPage
{
protected:
QObject *createPlugin ( const QString &classid, const QUrl &url,
const QStringList &paramNames, const QStringList &paramValues );
...
...

};
QObject* QMyWebPage::createPlugin ( const QString &classid, const QUrl &url,
const QStringList &paramNames, const QStringList &paramValues )
{
   if ( classid == "VideoPlayer" )
```

```
    {
        // 在这里创建一个自定义的带视频播放功能的窗体，
        VideoWindow* window = new VideoWindow();
        // 配置参数如 width=800 等，会在参数 paramNames 和 paramValues 中传过来
        window->setGeometry( ........ ) ;
        window->setSourceFile( ...... ) ;

        return window;        // 返回创建的窗体
    }
    ...
}
```

与截取 request 的方法一样，我们要让自己 QMyWebPage 被使用：

```
QWebView* Webview = new QWebView ( this );
QMyWebPage* page = new QMyWebPage ();
Webview->setPage ( page );    // 让 WebView 使用我们的 QMyWebPage
```

注意加载页面之前要打开插件使能的选项，方法如下：

```
QWebSettings* setting = Webview->settings ();
setting->setAttribute ( QWebSettings::PluginsEnabled, true );
```

至此，我们创建了自己的网页元素：类型为 VideoPlayer 的 object。网页可以像使用标准网页元素一样，灵活的使用嵌入式平台自己特有的功能。当然，不一定非要把这个网页元素用 application/x-qt-plugin 或者是 application/x-qt-styled-widget 来定义，Qt 也支持 type 不是这两者或者以动态链接库的方式来使用插件，这样就可以支持类似 FLASH 之类非 Qt 自定义的 object，关于这方面更多信息可以参考 Qt 的文档。

展望

PC 平台的软件技术在这些年有了很大的发展，体现在设计思想、开发工具、开发规模、模块服用和集成等方面。与之相比，嵌入式 Linux 上的开发总有些不尽如意，存在着应用简单、开发环境简陋、复用性低等问题。当 SaaS、Web2.0 等概念兴起，PC 与嵌入式 Linux 的软件开发水平的差距进一步的增加。

PC 平台的软件开发在经历了长久的发展之后，现在有向瘦客户端的方向发展的趋势。云计算等概念兴起，大量应用的负荷转移到了服务器端，很多应用都降低了对 客户端平台性能的要求。嵌入式平台性能近几年也有了较大的发展，频率上 G 的 CPU 已经得到普遍应用，2D/3D 和视频的加速在嵌入式 SoC 中也有了支持。

一方面是对性能要求的降低，另一方面是嵌入式平台性能的增长，两者不断的接近；并且当嵌入式 Linux 平台得到了 Web 的充分支持之后，PC 平台在 Web Server 方面的积累就可以被嵌入式 Linux 平台继承。以点菜机和条码扫描机为例，编写一个同时能被数十个客户端查询的服务器程序并不容易，但是，一个比较流行的 Web 服务器却能轻松的支持数千个并发。充分利用 Web 平台和技术，能让嵌入式的开发迅速发展，直接和 PC 平台开发站到同一起跑线上。

消费类电子领域的快速增长是近年来嵌入式发展的亮点。在手机上，浏览器应用已经成为最热点的领域，移动消费领域的客户也成为了各大 IT 巨头的争夺目标。在 Web 技术的继续发展下，如果应用服务都是通过网络提供，就可以由浏览器来解决嵌入式和 PC 平台的差异问题。当应用的问题被解决之后，由于嵌入式设备的可定制性和针对性，在很多特别应用模式下，PC 和嵌入式之间的界限正在变得越来越模糊，嵌入式 Linux 平台很可能会取代现今的 PC 平台，比如专门用来无线上网，以便随身处理邮件等事务的商务便携式设备；小巧精致，专用于欣赏在线影片和音乐的娱乐设备。而这正是各 CPU 巨头，如 Arm，Intel 等正在推动的移动平台发展的方向。

小结

本文先描述了当前嵌入式 Linux 开发中存在的问题，然后通过若干实际应用场景，阐述了用 Web 技术来改进嵌入式 Linux 设备开发的方式。文章随后着重描述了如何解决在嵌入式 Linux 中实现 Web 和本地代码配合的问题，以解决用 Web 取代传统开发方式中存在的障碍。

嵌入式 Linux 近年快速发展，在 IT 应用 Web 化这个发展方向已趋于明朗的背景下，在嵌入式 Linux 设备中提供 Web 支持，必然是将来嵌入式 Linux 开发平台中的一个非常重要的方面。

>>更多交流，请到 ChinaUnix【PHP论坛】：http://bbs2.chinaunix.net/forumdisplay.php?fid=27

关键词： 特性 方面 我们 实现

0 顶一下　　0 踩一下

相关文章

PHP V5.3 在 Unicode 方面有何新特性？
让 JavaScript 拯救 HTML5 的离线存储
MySQL新特性之复制特性的测试
使用 Agavi 实现访问控制
细察 PHP V5.3.0 特性
使用 PHP 实现云计算，第 2 部分：通过 Zend Framework 使用 Amazon EC2
使用 PHP 实现云计算，第 1 部分：结合使用 Amazon S3 和 Zend Framework
PHP 的未来
初步了解 PHP V6 中的新特性
PHP V5.3 中的新特性，第 5 部分：从 PHP V5.2 升级到 PHP V5.3

网友评论

已有0位网友发表了看法

我也来说两句！

验证码： 【输入评论后显示验证码，均为大写字母，点击图片更新】

发表评论