# **[Android总结之WebView使用总结](http://www.cnblogs.com/whoislcj/p/5645025.html)**

#### **前言：**

     今天修改项目中一个有关WebView使用的bug，激起了我总结WebView的动机，今天抽空做个总结。

#### **使用场景：**

##### **1.）添加权限**

<uses-permission android:name="android.permission.INTERNET" />

##### **2.）布局文件**

<WebView

android:id="@+id/webView"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent" />

##### **3.）数据加载**

 加载本地资源

webView.loadUrl("file:///android\_asset/example.html");

 加载网络资源

webView.loadUrl("www.xxx.com/index.html");

 添加请求头信息

Map<String,String> map=new HashMap<String,String>();  
map.put("User-Agent","Android");

webView.loadUrl("www.xxx.com/index.html",map);

也可以加载html片段

String data = " Html 数据";

webView.loadData(data, "text/html", "utf-8");

实测会发现loadData会导致中文乱码，所以一般情况使用如下代码

String data = " Html 数据";

webView.loadDataWithBaseURL(null,data, "text/html", "utf-8",null);

##### **4.）支持JavaScript**

     比如项目总js触发一个native函数关闭Activity

    设置支持JavaScript

WebSettings webSettings = webView.getSettings();

webSettings.setJavaScriptEnabled(true);//设置支持javascript

webView.addJavascriptInterface(new JavaScriptInterface(), "xueleapp");

JavaScriptInterface 接口定义

[IMG_256](http://www.cnblogs.com/whoislcj/p/javascript:void(0);)

public class JavaScriptInterface {

@android.webkit.JavascriptInterface

public void doTrainFinish() {

finish();

}

}

[IMG_257](http://www.cnblogs.com/whoislcj/p/javascript:void(0);)

##### **5.）设置WebViewClient 主要辅助WebView处理各种通知、请求事件**

比如要实现WebView中链接在WebView内部跳转

[IMG_258](http://www.cnblogs.com/whoislcj/p/javascript:void(0);)

webView.setWebViewClient(new WebViewClient() {

public boolean shouldOverrideUrlLoading(WebView view, String url) {

view.loadUrl(url);

return true;

}

});

[IMG_259](http://www.cnblogs.com/whoislcj/p/javascript:void(0);)

除此之外WebViewClient更多的处理网页的地址的解析和渲染，例如

      onLoadResource//加载资源时响应  
　　onPageStart//在加载页面时响应  
　　onPageFinish//在加载页面结束时响应  
　　onReceiveError//在加载出错时响应  
　　onReceivedHttpAuthRequest//获取返回信息授权请求

##### **6.）设置WebChromeClient主要辅助WebView处理Javascript的对话框，网站图标，网站title，加载进度等**

   比如加载进度获取title

[IMG_260](http://www.cnblogs.com/whoislcj/p/javascript:void(0);)

webView.setWebChromeClient(new WebChromeClient() {

@Override

public void onProgressChanged(WebView view, int newProgress) {

if (newProgress == 100) {

//网页加载完成

} else {

//网页加载中 }

}

});

[IMG_261](http://www.cnblogs.com/whoislcj/p/javascript:void(0);)

除了上面检测进度之外还有

　　onCloseWindow//关闭WebView  
　　onCreateWindow() //触发创建一个新的窗口  
　　onJsAlert //触发弹出一个对话框  
　　onJsPrompt //触发弹出一个提示  
　　onJsConfirm//触发弹出确认提示  
　　onProgressChanged //加载进度  
　　onReceivedIcon //获取网页icon  
　　onReceivedTitle//获取网页title

##### **7.）设置网页栈返回**

    webview会默认把浏览过去的网页进行压栈存储，所以我们有时需要实现回退到上一目录

[IMG_262](http://www.cnblogs.com/whoislcj/p/javascript:void(0);)

@Override

public boolean onKeyDown(int keyCode, KeyEvent event) {

if (keyCode == KeyEvent.KEYCODE\_BACK) {

if (webView.canGoBack()) {

webView.goBack();//返回上一浏览页面

return true;

} else {

finish();//关闭Activity }

}

return super.onKeyDown(keyCode, event);

}

[IMG_263](http://www.cnblogs.com/whoislcj/p/javascript:void(0);)

##### **8.）WebView 缓存控制**

* LOAD\_CACHE\_ONLY: 不使用网络，只读取本地缓存数据
* LOAD\_DEFAULT: 根据cache-control决定是否从网络上取数据。
* LOAD\_CACHE\_NORMAL: API level 17中已经废弃, 从API level 11开始作用同LOAD\_DEFAULT模式
* LOAD\_NO\_CACHE: 不使用缓存，只从网络获取数据.
* LOAD\_CACHE\_ELSE\_NETWORK，只要本地有，无论是否过期，或者no-cache，都使用缓存中的数据。

WebSettings webSettings = webView.getSettings();

webSettings.setCacheMode(WebSettings.LOAD\_CACHE\_ELSE\_NETWORK);

##### **9.）WebView屏幕自适应**

WebSettings webSettings = webView.getSettings();

webSettings.setUseWideViewPort(true);

webSettings.setLoadWithOverviewMode(true);

##### **10.）其他不常用设置**

[IMG_264](http://www.cnblogs.com/whoislcj/p/javascript:void(0);)

WebSettings webSettings = webView.getSettings();

webSettings.setSupportZoom(true); //支持缩放

webSettings.setLayoutAlgorithm(WebSettings.LayoutAlgorithm.SINGLE\_COLUMN); //支持内容重新布局

webSettings.supportMultipleWindows(); //多窗口

webSettings.setAllowFileAccess(true); //设置可以访问文件

webSettings.setNeedInitialFocus(true); //当webview调用requestFocus时为webview设置节点

webSettings.setBuiltInZoomControls(true); //设置支持缩放

webSettings.setJavaScriptCanOpenWindowsAutomatically(true); //支持通过JS打开新窗口

webSettings.setLoadsImagesAutomatically(true); //支持自动加载图片

[IMG_265](http://www.cnblogs.com/whoislcj/p/javascript:void(0);)

##### **11.)知识扩展WebViewJSBridge**

虽然谷歌也提供了js与native函数互相调用的方式，但是通过addjavascriptInterface这种方式在Android 4.2以下版本存在一定的安全隐患，在Android 4.2以上也需要加@JavascriptInterface注解，否则无法调用。基于上面的原因建议学习一下

WebViewJSBridge这个比较不错的开源框架，地址：https://github.com/firewolf-ljw/WebViewJSBridge

##### **12.）硬件加速**

 开启硬件加速强制使用GPU渲染，确实给app流畅度带来不小的提升，但是在使用过程中遇见webview闪烁，也有导致加载webView黑屏或者白屏

解决办法：关闭硬件加速

webView.setLayerType(View.LAYER\_TYPE\_SOFTWARE, null);

 这是把webview 中的硬件加速关闭。设置LAYER\_TYPE\_SOFTWARE后会把当前view转为bitmap保存。这样就不能开多个webview，否则会报out of memory。

需要在在webview中加入如下代码

protected void onMeasure(int widthMeasureSpec, int heightMeasureSpec) {

invalidate();

super.onMeasure(widthMeasureSpec, heightMeasureSpec);

}