# android WebView全面总结

****使用WebView并不需要开通网络权限****

网上有文章说webview需要开通internet权限，否则会出Web page not available错误，这是不对的，出现Web page not available并不是因为使用了webview，而是webview访问了网络，如果webview只是加载本地html（比如assets目录中的文件），或者只是加载带有html文本的字符串，即使没有internet权限，也不会报错。

****如何调用webview****

xml中

1. <WebView
2. android:id="@+id/blog\_detail\_webview"
3. android:layout\_width="fill\_parent"
4. android:layout\_height="wrap\_content"
5. android:background="#FFFFFF"/>

activity中

1. mWebView = (WebView)findViewById(R.id.blog\_detail\_webview);
2. mWebView.getSettings().setJavaScriptEnabled(false);
3. mWebView.getSettings().setSupportZoom(false);
4. mWebView.getSettings().setBuiltInZoomControls(false);
5. mWebView.getSettings().setLayoutAlgorithm(LayoutAlgorithm.SINGLE\_COLUMN);
6. mWebView.getSettings().setDefaultFontSize(18);

****基本设置****

上面的java代码部分相信大家都懂，可以看到WebView 和其他控件不同的地方在于其属性设置是调用mWebView.getSettings()来完成的，不知道谷歌这样设计的用意，其中：

****mWebView.getSettings().setJavaScriptEnabled(false);****

表示不支持js，如果想让java和js交互或者本身希望js完成一定的功能请把false改为true。

****mWebView.getSettings().setSupportZoom(false);****

设置是否支持缩放，我这里为false，默认为true。

****mWebView.getSettings().setBuiltInZoomControls(false);****

设置是否显示缩放工具，默认为false。

****mWebView.getSettings().setLayoutAlgorithm(LayoutAlgorithm.SINGLE\_COLUMN);****

一般很少会用到这个，用WebView组件显示普通网页时一般会出现横向滚动条，这样会导致页面查看起来非常不方便。LayoutAlgorithm是一个枚举，用来控制html的布局，总共有三种类型：  
NORMAL：正常显示，没有渲染变化。  
SINGLE\_COLUMN：把所有内容放到WebView组件等宽的一列中。  
NARROW\_COLUMNS：可能的话，使所有列的宽度不超过屏幕宽度。

****mWebView.getSettings().setDefaultFontSize(18);****

设置默认的字体大小，默认为16，有效值区间在1-72之间。

****加载内容****

（1）加载assets目录下的本地网页

一般我们都是把html文件放在assets目录下， WebView调用assets目录下的本地网页和图片等资源非常方便，使用形如

1. mWebView.loadUrl("file:///android\_asset/html/test1.html");

的调用方法即可。

（2）加载远程网页

1. mWebView.loadUrl("http://www.google.com");

（3）使用 LoadData 或者 loadDataWithBaseURL方法加载内容

有时候我们的webview可能只是html片段，而不是一个完整的网页，事实上绝大多数时候都是如此，完整的网页无需做成应用，而直接在浏览器访问。

这种情况我们使用 LoadData 或者 loadDataWithBaseURL方法，后者用的最多：

void loadDataWithBaseURL (String baseUrl, String data, String mimeType, String encoding, String historyUrl)

loadDataWithBaseURL()比loadData()多两个参数，可以指定HTML代码片段中相关资源的相对根路径，也可以指定历史Url，其余三个参数相同。

这里主要注意参数baseUrl，baseUrl指定了你的data参数中数据是以什么地址为基准的，因为data中的数据可能会有超链接或者是image元素，而很多网站的地址都是用的相对路径，如果没有baseUrl，webview将访问不到这些资源。

举个例子：

1. String body ="示例：这里有个img标签，地址是相对路径<img src='/uploads/allimg/130923/1FP02V7-0.png' />";
2. mWebView.loadDataWithBaseURL("http://www.jcodecraeer.com", body, "text/html", "utf-8",null);

如果baseUrl没有指定为http://www.jcodecraeer.com，那么这张图片将显示不出来。

上面的例子其实演示了loadDataWithBaseURL的用法，我们直接加载一个字符串里面的html内容，而有些时候这些内容是从assets目录下的本地网页文件中读取，下面我们将html/test1.html中的内容通过LoadData来加载：

1. String data = "";
2. try {
3. // 读取assets目录下的文件需要用到AssetManager对象的Open方法打开文件
4. InputStream is = getAssets().open("html/test2.html");
5. // loadData()方法需要的是一个字符串数据所以我们需要把文件转成字符串
6. ByteArrayBuffer baf = new ByteArrayBuffer(500);
7. int count = 0;
8. while ((count = is.read()) != -1) {
9. baf.append(count);
10. }
11. data = EncodingUtils.getString(baf.toByteArray(), "utf-8");
12. } catch (IOException e) {
13. e.printStackTrace();
14. }
15. // 下面两种方法都可以加载成功
16. mWebView.loadData(data, "text/html", "utf-8");
17. // wv.loadDataWithBaseURL("", data, "text/html", "utf-8", "");

这种通过读取文件再用loadData加载其实和mWebView.loadUrl("file:///android\_asset/html/test1.html")是一致的，只不过loadData方式因为没有指定地址的基准url，html/test1.html文件中一些资源文件或者链接地址会失效。

loadDataWithBaseURL和loadData两个方法加载的HTML代码片段的不同点在于，loadData()中的html data中不能包含'#', '%', '\', '?'四中特殊字符，在平时测试时，你的数据时，你的数据里含有这些字符，但不会出问题，当出问题时，你可以替换下。

      %，会报找不到页面错误，页面全是乱码。乱码样式见符件。

      #，会让你的goBack失效，但canGoBAck是可以使用的。于是就会产生返回按钮生效，但不能返回的情况。

****WebView内容的处理****

****android 中webView控件 padding不起作用****  
在一个布局文件中有一个WebView，想使用padding属性让左右向内留出一些空白，但是padding属性不起左右，内容照样贴边显示，反而移动了右边滚动条的位置。android的bug，用一个外围的layout包含webview，可以有所改进，但不能完全解决。其实正确的做法是在webView的加载的css中增加padding,没必要为了padding而更改xml布局文件。

****重写shouldOverrideUrlLoading时指定url****

指定只有url里包含eoe.cn的时候才在webview里打开，否则还是启动浏览器打开.

1. @Override
2. public boolean shouldOverrideUrlLoading(WebView view, String url) {
3. LogUtil.i(this, "url=" url);
4. if ( url.contains("eoe.cn") == true){
5. view.loadUrl(url);
6. return true;
7. }else{
8. Intent in = new Intent (Intent.ACTION\_VIEW , Uri.parse(url));
9. startActivity(in);
10. return true;
11. }
12. }

android:scrollbarStyle控制滚动条位置

WebView有一个设置滚动条位置的属性：android:scrollbarStyle 可以是insideOverlay可以是outsideOverlay，两个的区别是SCROLLBARS\_INSIDE\_OVERLAY的样式是滚动条在整个page里，类似css中的padding，看代码下的这个图吧，很清晰.

1. mWebView.setScrollBarStyle(View.SCROLLBARS\_OUTSIDE\_OVERLAY);

mWebView.setScrollBarStyle(View.SCROLLBARS\_INSIDE\_OVERLAY);