

**1. 简介**

WebView是一个基于webkit引擎、展现web页面的控件。

Android的Webview在低版本和高版本采用了不同的webkit版本内核，4.4后直接使用了Chrome。

# 2. 作用

* 显示和渲染Web页面
* 直接使用html文件（网络上或本地assets中）作布局
* 可和JavaScript交互调用

WebView控件功能强大，除了具有一般View的属性和设置外，还可以对url请求、页面加载、渲染、页面交互进行强大的处理。

**3. 使用介绍**

一般来说Webview可单独使用，可联合其子类一起使用，所以接下来，我会介绍：

* Webview自身的常见方法；
* Webview的最常用的子类   
  （WebSettings类、WebViewClient类、WebChromeClient类)
* Android和Js的交互

## 3.1 Webview常用方法

#### 3.1.1 WebView的状态

//激活WebView为活跃状态，能正常执行网页的响应

webView.onResume() ；

//当页面被失去焦点被切换到后台不可见状态，需要执行onPause

//通过onPause动作通知内核暂停所有的动作，比如DOM的解析、plugin的执行、JavaScript执行。

webView.onPause()；

//当应用程序(存在webview)被切换到后台时，这个方法不仅仅针对当前的webview而是全局的全应用程序的webview

//它会暂停所有webview的layout，parsing，javascripttimer。降低CPU功耗。

webView.pauseTimers()

//恢复pauseTimers状态

webView.resumeTimers()；

//销毁Webview

//在关闭了Activity时，如果Webview的音乐或视频，还在播放。就必须销毁Webview

//但是注意：webview调用destory时,webview仍绑定在Activity上

//这是由于自定义webview构建时传入了该Activity的context对象

//因此需要先从父容器中移除webview,然后再销毁webview:

rootLayout.removeView(webView);

webView.destroy();

#### 3.1.2 关于前进 / 后退网页

//是否可以后退

Webview.canGoBack()

//后退网页

Webview.goBack()

//是否可以前进

Webview.canGoForward()

//前进网页

Webview.goForward()

//以当前的index为起始点前进或者后退到历史记录中指定的steps

//如果steps为负数则为后退，正数则为前进

Webview.goBackOrForward(intsteps)

* 1
* 2
* 3
* 4
* 5
* 6
* 7
* 8
* 9
* 10
* 11
* 12
* 13

**常见用法：Back键控制网页后退**

* 问题：在不做任何处理前提下 ，浏览网页时点击系统的“Back”键,整个 Browser 会调用 finish()而结束自身
* 目标：点击返回后，是网页回退而不是推出浏览器
* 解决方案：在当前Activity中处理并消费掉该 Back 事件

public boolean onKeyDown(int keyCode, KeyEvent event) {

if ((keyCode == KEYCODE\_BACK) && mWebView.canGoBack()) {

mWebView.goBack();

return true;

}

return super.onKeyDown(keyCode, event);

}

**我用到的：**

@Override  
public boolean onKeyDown(int keyCode, KeyEvent event) {  
 if ((keyCode == KEYCODE\_BACK) && webview.canGoBack()) {  
 if(webview.getUrl().equals("http://ng19839177.iask.in:37043/index.html")||webview.getUrl().equals("http://ng19839177.iask.in:37043/log.jsp#")){  
 System.exit(0); //如果是在首页返回就退出  
 }  
 webview.loadUrl("http://ng19839177.iask.in:37043/index.html");  
 return true;  
 }  
 return super.onKeyDown(keyCode, event);  
}

#### 3.1.3 清除缓存数据

//清除网页访问留下的缓存

//由于内核缓存是全局的因此这个方法不仅仅针对webview而是针对整个应用程序.

Webview.clearCache(true);

//清除当前webview访问的历史记录

//只会webview访问历史记录里的所有记录除了当前访问记录

Webview.clearHistory()；

//这个api仅仅清除自动完成填充的表单数据，并不会清除WebView存储到本地的数据

Webview.clearFormData()；