WebView

[Загружаем локальные страницы и картинки](http://developer.alexanderklimov.ru/android/views/webview.php#local)  
[Загружаем данные при помощи loadData() и loadDataWithBaseURL()](http://developer.alexanderklimov.ru/android/views/webview.php#loaddata)  
[Проблемы с кодировкой](http://developer.alexanderklimov.ru/android/views/webview.php#encoding)  
[Методы](http://developer.alexanderklimov.ru/android/views/webview.php#methods)  
[Используем зум для просмотра](http://developer.alexanderklimov.ru/android/views/webview.php#zoom)  
[Прозрачность](http://developer.alexanderklimov.ru/android/views/webview.php#transparent)  
[Дополнительное чтение](http://developer.alexanderklimov.ru/android/views/webview.php#seealso)

**WebView** — это компонент, который позволяет встраивать веб-страницы в приложения, своеобразный мини-браузер.

Надеюсь, вы уже познакомились с [базовым примером по созданию собственного браузера](http://developer.alexanderklimov.ru/android/mybrowser.php). Рассмотрим дополнительные возможности элемента **WebView**, который расположен в разделе**Widgets** на панели инструментов.

Загружаем локальные страницы и картинки

Если вы хотите загружать в **WebView** страницы не из интернета, а со своего приложения, то разместите нужные файлы в папке [assets](http://developer.alexanderklimov.ru/android/theory/assets.php), например, **assets/mypage.html**. Доступ к файлу вы можете получить через конструкцию **file://android\_asset**:

myBrowser = (WebView)findViewById(R.id.mybrowser);

myBrowser.loadUrl("file:///android\_asset/mypage.html");

Аналогично поступаем с картинками, которые встречаются в html-файле

<img src="file:///android\_asset/myimage.gif">

Также можно загрузить файл из папки **res/raw**:

webView.loadUrl("file:///android\_res/raw/cat.html");

Если картинка находится на внешнем накопителе, то попробуйте вариант:

WebView webView = (WebView) findViewById(R.id.webView);

String imageName = "cutecat.png";

String catUrl = "file://"

+ Environment.getExternalStorageDirectory().getAbsolutePath()

.toString() + "/" + imageName;

webView.loadUrl(catUrl);

Загружаем данные при помощи loadData() и loadDataWithBaseURL()

Данные можно загрузить с помощью метода **loadData()**:

String htmlText = "<html><body>Percent test: 100% </body></html>";

WebView webView = (WebView) findViewById(R.id.webView);

webView.loadData(htmlText, "text/html", "en\_US");

Если текст простой, то этот способ подойдёт. Но в данном примере встречается символ процента, который относится к спецсимволам и часть текста может оказаться недоступной. Если в тексте встречаются подобные символы, то лучше использовать метод **loadDataWithBaseURL()**:

webView.loadDataWithBaseURL(null, htmlText, "text/html", "en\_US", null);

Если вам приходится использовать **loadData()**, то спецсимволы можно заменить при помощи метода **replace()**:

String webData = stringBuffer.toString(); // поступающие данные

webData = webData.replace("#", "%23");

webData = webData.replace("%", "%25");

webData = webData.replace("\\", "%27");

webData = webData.replace("?", "%3f");

webView.loadData(webData, "text/html", "UTF-8");

Проблемы с кодировкой

У меня есть программа в Google Play, использующая **WebView**. К моему удивлению, некоторые пользователи жаловались, что текст нечитаем, так как они видят только кракозябры. Особенно много жалоб было от пользователей с планшетами. Оказалось, что проблема довольна распространённая и обсуждается на форумах. Танцы с бубнами (установка явной кодировки UTF-8) не помогают. Нашёл один ответ, который у некоторых заработал, на всякий случай я его здесь оставлю.

// перед загрузкой данных (load...)

WebSettings settings = mWebView.getSettings();

settings.setDefaultTextEncodingName("utf-8");

Но я рекомендую просто использовать метод **loadDataWithBaseURL()**. Работает стабильно.

Методы

У **WebView** есть множество методов, которые позволяют добиваться полной функциональности как у обычного браузера - обновить страницу, перейти на предыдущую страницу и т.д. Часть методов представлена ниже:

* reload()
* goForward()
* goBack()

Используем зум для просмотра

Не забывайте, что **WebView** можно использовать не только для просмотра html-страниц, но и для просмотра изображений. Поэтому данный компонент вполне можно использовать как вьювер картинок, к тому же вы можете включить встроенный механизм масштабирования:

mWebView = (WebView) findViewById(R.id.webView1);

// устанавливаем Zoom control

mWebView.getSettings().setBuiltInZoomControls(true);

// загружаем картинку (не забудьте установить разрешение на интернет)

mWebView.loadUrl("http://netsources.narod.ru/friday/alkocat.jpg");

this.setTitle("WebView");



Прозрачность

Устанавливать прозрачность лучше программно. Встречал жалобы, что через XML это свойство не работает.

webView.setBackgroundColor(0x00000000);

WebView в Lollipop

В Android 5.0 компонент доступен в Google Play ([Android System WebView](https://play.google.com/store/apps/details?id=com.google.android.webview)) и его можно обновлять на устройстве.

Компонент теперь основывается на движке Chromium и поддерживает следующие новинки.

* WebRTC
* WebAudio
* WebGL

Можно ознакомиться с некоторыми примерами - [GoogleChrome/chromium-webview-samples](https://github.com/GoogleChrome/chromium-webview-samples). Там есть примеры с WebRTC, полноэкранным режимом, касаниями экрана, выбора файла, работой с JavaScript-сценариями.

Советы

Фон

Если вы заметили, что экран мерцает во время загрузки **WebView**, то поменяйте фон. Мерцание происходит из-за смены фона приложения (темы), на белый фон по умолчанию для**WebView**, а потом на фон, который прописан на странице.

mWebView.setBackgroundColor(Color.parseColor("#3498db"));

mWebView.setBackgroundColor(getResources().getColor(R.color.my\_color\_name));

// и т.п.

Касания экрана

Так как поддерживаются касания экрана, то старайтесь использовать на веб-странице визуальные эффекты нажатия кнопок и других элементов при помощи псевдокласса **:active**, например, так:

.btn {

display: inline-block;

position: relative;

background-color: #f39c12;

padding: 14px;

border-radius: 5px;

border-bottom-style: solid;

border-width: 4px;

border-color: #DA8300;

}

.btn:active {

background-color: #E68F05;

border-color: #CD7600;

border-width: 2px;

top: 2px;

}

Дополнительное чтение

[Migrating to WebView in Android 4.4](http://developer.android.com/intl/ru/guide/webapps/migrating.html)

[WebView - создай свой браузер](http://developer.alexanderklimov.ru/android/mybrowser.php)

[Продвинутый WebView](http://developer.alexanderklimov.ru/android/webview-advanced.php)