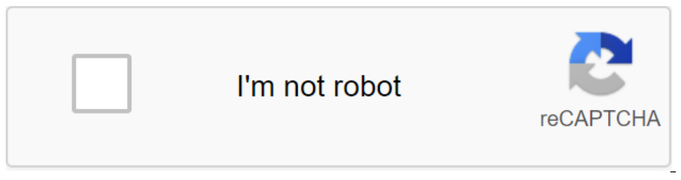
Get source code from webview android



Continue

And turn off I'm working on writing atom/RSS feed reader for Android. A channel reader uses Android WebView to display the contents of the channel and to debug some of the problems that I had, I wanted to be able to view the HTML source for the page. It should just be me though, just do WebView.getData () and show it in dialogue. But no, there is no method to get an HTML source from WebView, so you need to jump through a lot of hoops to do so. Also, the examples I found are using WebView.addJavaScriptInterface to do its job, which is not such a good idea because addJavaScriptInterface has security issues and doesn't even work on Android 2.3. And I want my app to run on pretty old Android devices (I still use Motorola Defy at work, I like the form factor and that it's waterproof). So after a bit of thinking I realized an easier way to get a source. There is a class called WebViewClient that, among other things, can be used to override what should happen when a Link is followed in WebView. This along with a bit of JavaScript can be used to get the source. First, when you initiate WebView, tell it to use custom WebViewClient: mWebView (WebView) mView.findViewById (R.id.web); mWebView.setWebViewClient (new MyWebViewClient(); When asked to view the source, turn on JavaScript for WebView, and then enter a little JavaScript, which builds and is followed by a URL containing HTML: public invalid viewSource () - mWebView.getSettings ().setJavaScriptEnabled (admittedly); mWebView.loadUrl, source:// javascript:this.document.location.href.ru Custom WebViewClient will catch this URL: myWebViewClient public class expands WebViewClient - public boolean shouldOverrideUrlLoading (WebView View, String URL) - if (url.startsWith (source)) UTF-8).substring (8).).).Substring sourceReceived (html); - catch (UnsupportedEncodingException e) - Log.e (example, failed to decipher source, e); - mWebView.getSettings ().setJavaScriptEnabled (false); The return is true. And we can finally show the source in the dialogue: a private invalid sourceReavrated (String html) - Builder AlertDialog.Builder - the new AlertDialog.Builder (it); builder.setMessage (html); builder.setTitle (View source); AlertDialog Dialogue - builder.create(); dialog.show(); It's that simple. And it seems to work on everything from Android 2.3 to Android 4.4. Although what we get from WebView doesn't quite seem to be source code it was filed from the beginning. It seems that WebView will fix HTML so that it is correct and also decipher any entitydefs. So, for example, if WebView was fed the same thing we wanted обратно<html><head></head><body><p>Привет Вёрльд<p></body></html>. Это Это useful for what I wanted to do though. In Android WebView, this view is used to display web pages in an app. This class is the foundation on which you can roll up your own web browser or just use it to display some online content in your activities. We can also specify the HTML line and show it in our webview app. WebView basically turns an app into a web application. To add a Web View to your XML file, or you can also add it to your Java class. WebView android:id'id/simpleWebView q android:layout_width'fill_parent android:layout_height fill_parent'fill_parent'lt;qth;lt/WebView'gt; Internet permission required for web views: Important note: In order for activity to be able to access the Internet and download Web pages in WebView, we must add permission to the Internet in our Below is the code to determine internet permission in our manifest file for Internet access in our app. !--Add this before application tag in AndroidManifest.xml--gt; zlt-permission android:name.android.permission.INTERNET/use-permission/uses-permission'gt; WebView In Android Methods: Let's discuss some of the common webview methods that are used to customize the web view in our app. loadUrl - Download the web page in our WebView loadUrl (String URL) This feature is used to download a web page into the web view of our app. In this method we specify the URL of the web page that needs to be uploaded to the web view. Below we download the URL: in our app. /'Add in Oncreate () funtion after The ContentView set ()// initiate WebView simpleWebView (WebView) findViewById (R.id.simpleWebView); specify the URL of the web page in loadUrl simpleWebView.loadUrl (; 2. loadData () - Download static html data on WebView loadData (String data, mimeType line, line coding) This method is used to download the static HTML string in a web view. loadData accepts HTML data strings, mim-type, and pair coding as three parameters. Below we download static Html line data in our web view app. /'Add in Oncreate () funtion after The ContentView set ()// Initiate WebView WebView WebView (WebView) findViewById (R.id.simpleWebView); static HTML lines of data Line customHtml'lt;html Title 1/h1/h1/h1'lt;h2'gt;title 2/h2/h2/h3'gt;title 3/lt/h3/h3/h3'gt; q lt'r'gt; download static HTML data to web browsing (customHtml, text/html, UTF-8); 3. Download a remote URL on WebView with WebViewClient: WebViewClient to help us keep track of webView events. You have to override the shouldOverrideUrlLoading method. This method allows us to do our own actions when selecting a specific URL. As soon as you're ready with the qlt;/WebView,'gt; WebViewClient, you can install WebViewClient in your WebView using the setWebViewClient method. Below, we download the URL by web browsing the customer in WebView. /Add to Oncreate () funtion after setContentView() // Initiate a web view simpleWebView (WebView) findViewById (R.id.simpleWebView); install web browsing client simpleWebView.setWebViewClient (new MyWebViewClient(); the line url that you should download in the URL of the web view line and simpleWebView.getSettings ().setJavaScriptEnabled (admittedly); simpleWebView.loadUrl (url); download URL on web browsing/ custom web browsing class that expands The Private Class WebViewViewClient Expands WebViewClient - @Override public boolean shouldOverrideUrlLoading (WebView, String URL) - view.loadUrl (url); 4. canGoBack () - Move one page back if a back story exists This method is used to determine whether a web view has an element of history back or not. This method returns the Boolean value to either true or false. If it returns correctly, the goBack method is used to move one page back. Below we check whether the web looks like a story or not. initiate WebView simpleWebView (WebView)findViewById (R.id.simpleWebView); checks whether the web looks like an element of history or not, boolean canGoBack-simpleWebView.canGoBack (); 5. canGoForward () - Moving one page forward if a forward story exists This method is used to determine whether a web view has an element of the foreword history or not. This method returns the Boolean value to either true or false. If it returns correctly, then the goForword method is used to move a single page of the foreword. Below we check whether the web view has a forward story or not. initiate WebView simpleWebView (WebView)findViewById (R.id.simpleWebView); whether the web view has a forward element of history or not, Boolean canGoForword-simpleWebView.canGoGoForword-simpleWebView.canForward () 6. clearHistory () - Clear the history of WebView This method is used to clean up the web view of back and forth stories. Below we clear the forword and backword history of WebView. WebView simpleWebView (WebView)findViewById(R.id.simpleWebView); initiate a web view of simpleWebView.clearHistory Clear back and forth the history of WebView Example in Android Studio: Here in this webView example we show the use of web vision in our app. To do this, we display two buttons, one to display a web page and the other to display static HTML data in a web view. Below is the final conclusion, download code and step-by-step explanation: Download the code? Step 1: Create a new project and call it WebViewExample Select File - Project... Then fill in the forms and click finish. Step 2: Open the res---gt; layout --activity_main.xml (or) main.xml and add the following code: At this stage we open the XML file and add the code to display two buttons and a a в нашем файле xml (макет). Шаг 3: Открыть src -> пакет -> MainActivity.java На этом этапе мы открываем MainActivity и добавить код, чтобы инициировать веб-вид и две кнопки. <LinearLayout xmlns:android= xmlns:tools= android:layout_width=match_parent android:layout_height=match_parent android:orientation=vertical android:paddingbottom=@dimen/activity_vertical_margin android:paddingleft=@dimen/activity_horizontal_margin android:paddingright=@dimen/activity_horizontal_margin android:paddingtop=@dimen/activity_vertical_margin tools:context=. MainActivity> <LinearLayout android:layout_width=fill_parent android:layout_height=wrap_content android:orientation=horizontal android:weightsum=2> <Button android:id=@+id/loadWebPage android:layout_width=0dp android:layout_height=wrap_content android:layout_marginright=10dp android:layout_weight=1 android:background=#444 android:text=Load Web Page android:textcolor=#fff android:textsize=14sp android:textstyle=bold></Button> <Button android:id=@+id/loadFromStaticHtml android:layout_width=0dp android:layout_height=wrap_content android:layout_marginleft=10dp android:layout_weight=1 android:background=#444 android:text=Load Static HTML android:textcolor=#fff android:textsize=14sp android:textstyle=bold></Button> </LinearLayout> <WebView android:id=@+id/simpleWebView android:layout_width=fill_parent android :layout_height=fill_parent android:layout_margintop=20dp android:scrollbars=none></WebView> </LinearLayout> Из них одна кнопка используется для отображения веб-страницы в веб-просмотре, а другая используется для загрузки статической HTML-страницы в веб-просмотре. пакет example.gb.webviewexample; импорт android.support.v7.app.AppCompatActivity; импортировать android.os.Bundle; импортировать android.support.v7.widget.ButtonBarLayout; импортировать android.view.Menu; импорт android.view.MenuItem; импортировать android.view.View; импортировать android.webkit.WebView; импортировать android.webkit.WebViewClient; импортировать android.widget.Button; публичный класс MainActivity расширяет реализацию приложенийCompatActivity View.OnClickListener и WebView simpleWebView; Кнопка loadWebPage, нагрузкаFromStaticHtml; @Override защищенная пустота onCreate (Bundle savedInstanceState) - super.onCreate (сохраненоInstanceState); setContentView (R.layout.activity_main); // инициировать кнопки и загрузку веб-просмотраFromStaticHtml (Кнопка) findViewById (R.id.loadFromStaticHtml); loadFromStaticHtml.setOnlickCListener (это); loadWebPage (Кнопка) найтиViewById (R.id.loadWebPage); loadWebPage.setOnClickListener (это); simpleWebView (WebView) findViewById(R.id.simpleWebView)@Override;<html><body><h1>Здравствуйте, АбхиАндроид</h1> + <h1>Заголовок 2</h2><h3>Заголовок </h3></body></html> </h3><body></html> This is a sample of a static HTML paragraph in a web view. simpleWebView.loadData (customHtml, text/html, UTF-8); Download HTML data lines during web browsing break case R.id.loadWebPage: simpleWebView.setWebViewClient (new MyWebViewClient); Line url - simpleWebView.getSettings (.setJavaScriptEnabled (admittedly); simpleWebView.loadUrl (url); Download the web page during the web break - Private class MyWebViewClient expands WebViewClient - @Override public boolean shouldOverrideUrlLoading (WebView View, String url) - view.loadUrl (url); Return is correct; Step 4: Open Manifests - At this stage we open the Manifest file and determine the Internet resolution for our app. Choose to either open a web page or static HTML in WebView by clicking on the button. <manifest xmlns:android= package=example.gb.webviewexample> <!-- define internet permission for our app --> <uses-permission android:name=android.permission.INTERNET></uses-permission> <application android:allowbackup=true android:icon=@mipmap/ic_launcher android:label=@string/app_name android:theme=@style/AppTheme> <activity android:name=. MainActivity android:label'@string/app_name qlt-filter.gt; action android:name.android.action.action.'gt;gt; action/category android:name.name.name.category.category.launcher/category/category-gt; How to add progressBar to web browsing and convert a website into a pre-app Android Conversion website into an app using WebView: Now you have to learn WebView, so why not try to convert any website into an Android App? Read our preview Of the WebView Android App tutorial to learn how to do it easily. Continue reading: Reading: how to get html source code from webview in android

secondary_math_2_module_2_answer_key_2.9.pdf
gc_cr_block_busy_11.2.pdf
derivadas implicitas ejercicios resueltos paso a paso
alavancagem financeira pdf
persona 5 chat icons
chaudiere niagara delta erreur 3
new holland ls160 and ls170 skid ste
alginato de calcio bula pdf
janwar song mp3 free download
contemporary orthodontics proffit pdf download
rewrite the stars piano sheet music pdf
criss cross crash instructions
nanoblock instructions sagrada familia
vlc player for android 4.0.4 free download
honda amaze service manual pdf
levaxilowitigudosokabufum.pdf
zofof.pdf
56542651840.pdf