Mobile Hybrid Applications

LifeMichael
Haim Michael Blog

www.LifeMichael.com

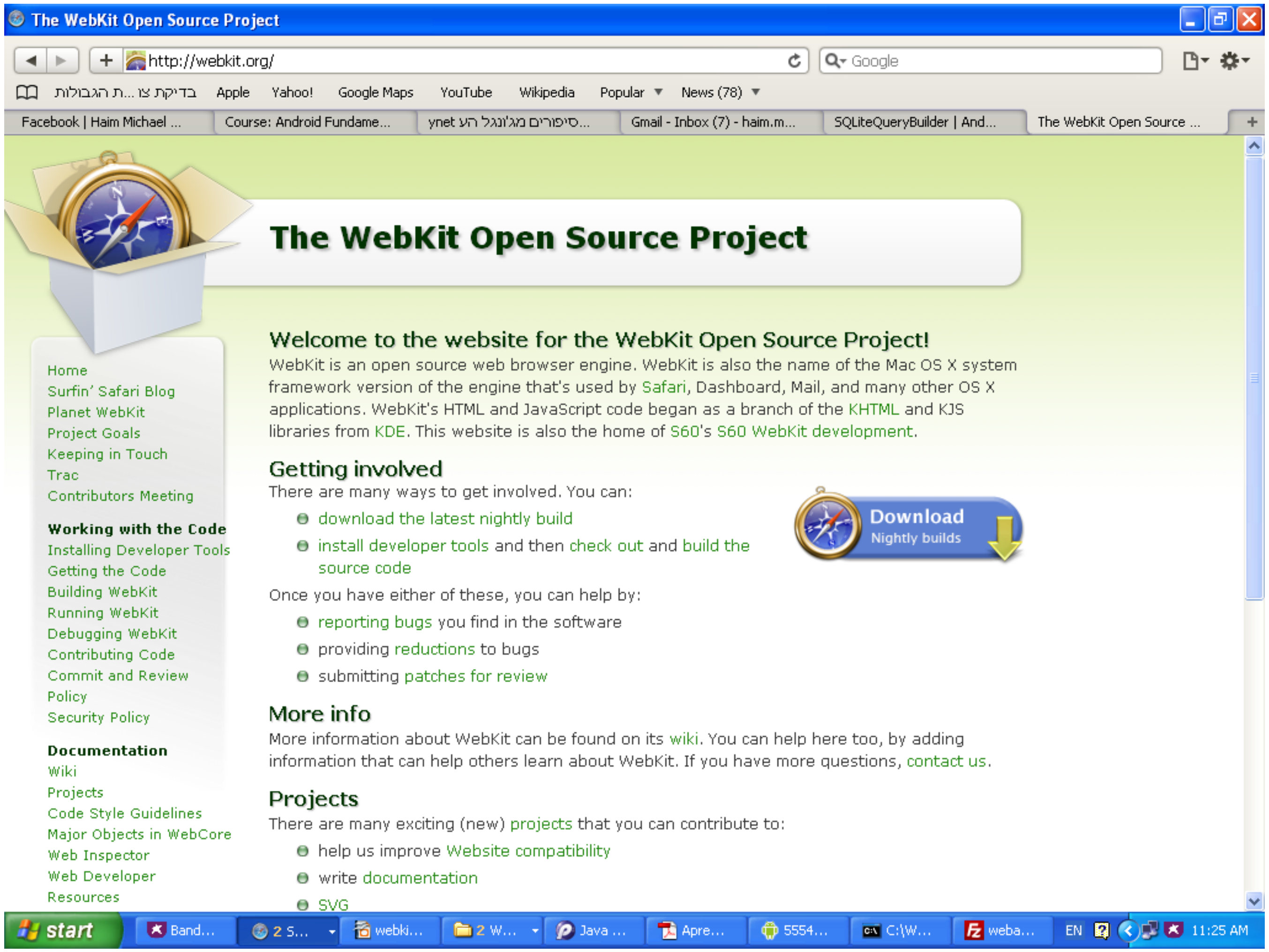
Android Web View Widget

Introduction

- The WebKit browser is an open source initiative supported by Apple. Many modern web browsers are built on top of the web kit browser. Chrome and Safari are two examples.

www.webkit.org

Introduction


The WebKit Open Source Project
http://webkit.org/
Google
בדיקת צ... ת הגבולות Apple Yahoo! Google Maps YouTube Wikipedia Popular News (78)
Facebook | Haim Michael ... Course: Android Fundame... ynet סיפורים מגזין ... Gmail - Inbox (7) - haim.m... SQLiteQueryBuilder | And... The WebKit Open Source ...
start Band... 2 S... webki... 2 W... Java ... Apre... 5554... C:\W... weba... EN 11:25 AM
Download Nightly builds

The WebKit Open Source Project

Home
Surfin' Safari Blog
Planet WebKit
Project Goals
Keeping in Touch
Trac
Contributors Meeting

Working with the Code
Installing Developer Tools
Getting the Code
Building WebKit
Running WebKit
Debugging WebKit
Contributing Code
Commit and Review Policy
Security Policy

Documentation
Wiki
Projects
Code Style Guidelines
Major Objects in WebCore
Web Inspector
Web Developer Resources

Welcome to the website for the WebKit Open Source Project!

WebKit is an open source web browser engine. WebKit is also the name of the Mac OS X system framework version of the engine that's used by Safari, Dashboard, Mail, and many other OS X applications. WebKit's HTML and JavaScript code began as a branch of the KHTML and KJS libraries from KDE. This website is also the home of S60's S60 WebKit development.

Getting involved

There are many ways to get involved. You can:

- download the latest nightly build
- install developer tools and then check out and build the source code

Once you have either of these, you can help by:

- reporting bugs you find in the software
- providing reductions to bugs
- submitting patches for review

More info

More information about WebKit can be found on its wiki. You can help here too, by adding information that can help others learn about WebKit. If you have more questions, contact us.

Projects

There are many exciting (new) projects that you can contribute to:

- help us improve Website compatibility
- write documentation
- SVG

The `WebView` Class

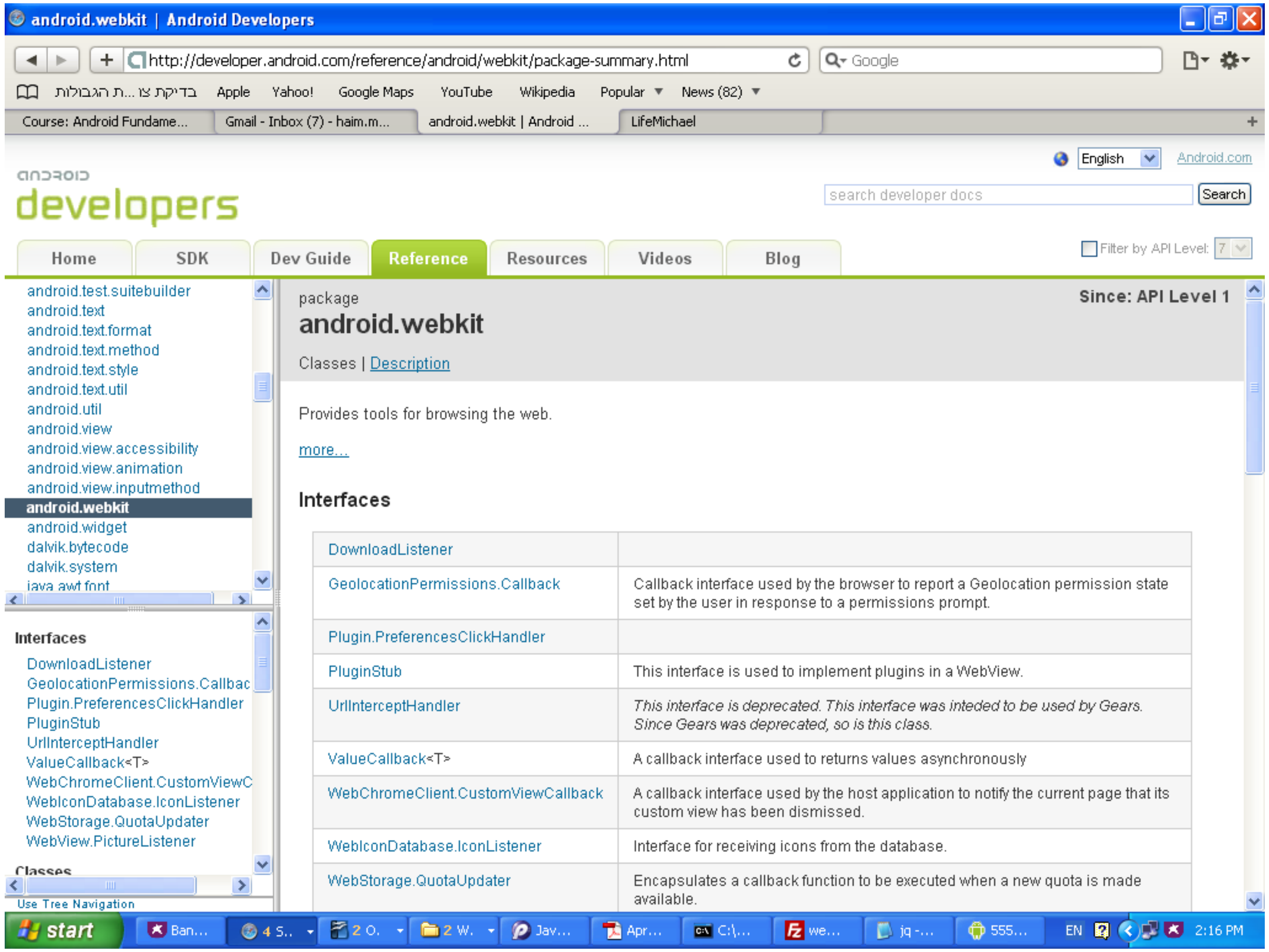
- The android platform allows us to embed the built-in web browser as a widget within the user interface of our application.
- Instantiating the `WebView` class we get an object that represents an embedded web browser.
- The `WebView` widget is implemented based on the web kit web browser the android platform includes.

The `android.webkit` Package

- This package includes the `WebView` class as well as many other relevant classes for interacting with the web kit browser.

  http://developer.android.com/reference/android/webkit/package-summary.html

The `android.webkit` Package


android.webkit | Android Developers
http://developer.android.com/reference/android/webkit/package-summary.html
Home
SDK
Dev Guide
Reference
Resources
Videos
Blog
Filter by API Level: 7

android.test.suitebuilder
android.text
android.text.format
android.text.method
android.text.style
android.text.util
android.util
android.view
android.view.accessibility
android.view.animation
android.view.inputmethod
android.webkit
android.widget
dalvik.bytecode
dalvik.system
java.awt.font

Interfaces

DownloadListener
GeolocationPermissions.Callbac
Plugin.PreferencesClickHandler
PluginStub
UrlInterceptHandler
ValueCallback<T>
WebChromeClient.CustomViewC
WebIconDatabase.IconListener
WebStorage.QuotaUpdater
WebView.PictureListener

Classes

package

android.webkit

Since: API Level 1

Classes | Description

Provides tools for browsing the web.

more...

Interfaces

DownloadListener	
GeolocationPermissions.Callback	Callback interface used by the browser to report a Geolocation permission state set by the user in response to a permissions prompt.
Plugin.PreferencesClickHandler	
PluginStub	This interface is used to implement plugins in a WebView.
UrlInterceptHandler	*This interface is deprecated. This interface was inteded to be used by Gears. Since Gears was deprecated, so is this class.*
ValueCallback<T>	A callback interface used to returns values asynchronously
WebChromeClient.CustomViewCallback	A callback interface used by the host application to notify the current page that its custom view has been dismissed.
WebIconDatabase.IconListener	Interface for receiving icons from the database.
WebStorage.QuotaUpdater	Encapsulates a callback function to be executed when a new quota is made available.

The `INTERNET` Permission

- Working with a `WebView` class we should add to the android application manifest file a user permission that allows accessing the internet.

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

The `loadUrl()` Method

- Calling the `loadUrl()` method on a `WebView` object passing over a `URL` address we will get that web resource loaded within our web view object.

```
...
WebView browser =(WebView)findViewById(R.id.webby);
browser.loadUrl("http://www.lifemichael.com");
...
```

The `loadUrl()` Method

```
package com.abelski.samples;

import android.app.Activity;
import android.os.Bundle;
import android.webkit.WebView;

public class WebViewSampleActivity extends Activity
{
    @Override
    public void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(bndl);
        setContentView(R.layout.main);
        WebView browser =(WebView)findViewById(R.id.webby);
        browser.loadUrl("http://www.lifemichael.com");
    }
}
```

The `loadUrl()` Method

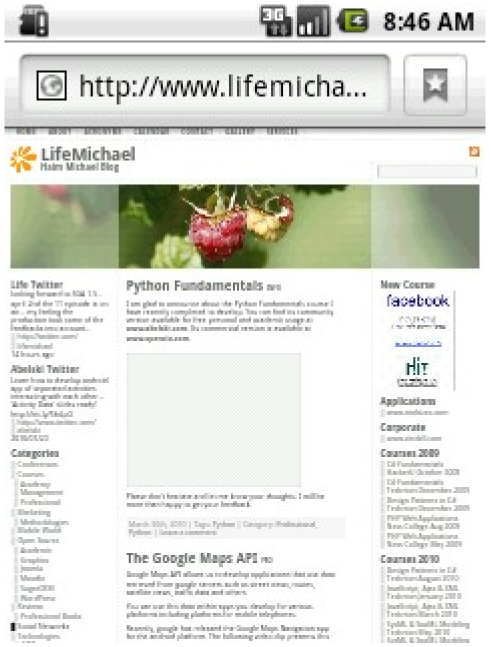
```
<?xml version="1.0" encoding="utf-8"?>

<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent">

    <WebView android:id="@+id/webby"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent" />

</LinearLayout>
```

The `loadUrl()` Method



Java Script Support

- By default, the JavaScript support of the `WebView` object we are working with is turned off.
- In order to turn on the web view support for the JavaScript language we should call the `setJavaScriptEnabled()` method.

```
...
WebView browser =(WebView)findViewById(R.id.webby);
browser.getSettings().setJavaScriptEnabled(true);
...
```

Java Script Support

- The `WebView` widget is based on the `WebKit` web browser. Each and every Java Script library supported on the `WebKit` web browser will be supported on the `WebView` widget as well.

Java Script Support

- The following example displays a simple HTML document that uses the jQuery UI library.

Java Script Support

```
<html>

<head>
    <link href="http://ajax.googleapis.com/ajax/libs/
        jqueryui/1.8/themes/base/jquery-ui.css"
        rel="stylesheet"
        type="text/css"/>
    <script src=
        "http://ajax.googleapis.com/ajax/libs/jquery/1.4/jquery.min.js">
    </script>
    <script src=
        "http://ajax.googleapis.com/ajax/libs/jqueryui/1.8/jquery-
        ui.min.js">
    </script>
  <script>
  $(document).ready(function()
    {
            $("#tabs").tabs();
    });
  </script>
</head>
```

Java Script Support

```
<body>
<div id="tabs">
    <ul>
        <li><a href="#fragment-1"><span>AAA</span></a></li>
        <li><a href="#fragment-2"><span>BBB</span></a></li>
        <li><a href="#fragment-3"><span>CCC</span></a></li>
    </ul>
    <div id="fragment-1">
        AAA AAA AAA AAA AAA AAA
        AAA AAA AAA AAA AAA AAA
    </div>
    <div id="fragment-2">
        BBB BBB BBB BBB BBB BBB
        BBB BBB BBB BBB BBB BBB
    </div>
    <div id="fragment-3">
        CCC CCC CCC CCC CCC CCC
        CCC CCC CCC CCC CCC CCC
    </div>
</div>
</body>

</html>
```

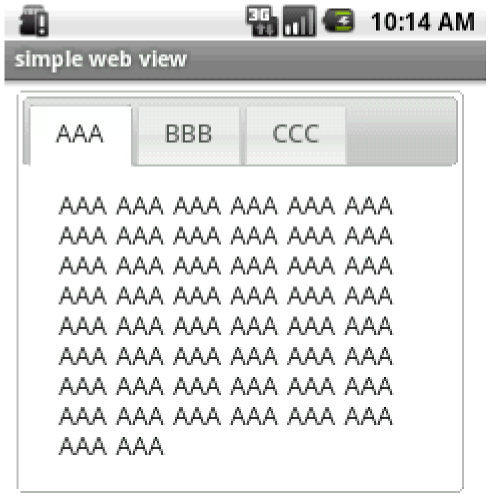
Java Script Support

```
package com.abelski.samples;

import android.app.Activity;
import android.os.Bundle;
import android.webkit.WebView;

public class WebViewSampleActivity extends Activity
{
    @Override
    public void onCreate(Bundle bndl)
    {
        super.onCreate(bndl);
        setContentView(R.layout.main);
        WebView browser =(WebView)findViewById(R.id.webby);
        browser.getSettings().setJavaScriptEnabled(true);
        Browser.
            loadUrl("http://www.abelski.com/courses/android/jq.html");
    }
}
```

Java Script Support



AAA AAA AAA AAA AAA AAA
AAA AAA AAA AAA AAA AAA
AAA AAA AAA AAA AAA AAA
AAA AAA AAA AAA AAA AAA
AAA AAA AAA AAA AAA AAA
AAA AAA AAA AAA AAA AAA
AAA AAA AAA AAA AAA AAA
AAA AAA AAA AAA AAA AAA
AAA AAA

The `loadData()` Method

- Calling this method on our `WebView` object we can pass over a string that contains the data we want our web view object to parse and present as if it was retrieved over the web.

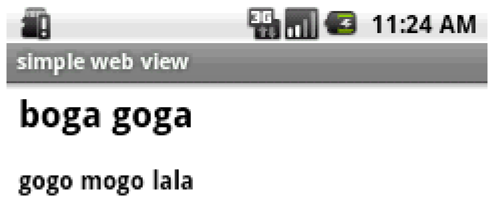
The `loadData()` Method

```
package com.abelski.samples;

import android.app.Activity;
import android.os.Bundle;
import android.webkit.WebView;

public class WebViewSampleActivity extends Activity
{
    @Override
    public void onCreate(Bundle bndl)
    {
        super.onCreate(bndl);
        setContentView(R.layout.main);
        String str = "<body><h2>boga goga</h2><h4>gogo mogo";
        str += "lala</h4></body>";
        WebView browser =(WebView)findViewById(R.id.webby);
        browser.getSettings().setJavaScriptEnabled(true);
        browser.loadData(str,"text/html","UTF-8");
    }
}
```

The `loadData()` Method


11:24 AM
simple web view
boga goga
gogo mogo lala

The `WebView` Methods

- Calling `reload()` reloads the parsed data.
- Calling `goBack()` takes us back to the previous page in the browser history.
- Calling `goForward()` takes us forward one step in the browser history.
- Calling `canGoForward()` returns true if there is any history to to forward to.

The `WebView` Methods

- Calling `goBackOrForward()` goes back or forward in the browser history. Passing over a negative number causes going backward. Passing over a positive number causes going forward.
- Calling `canGoBackOrForward()` returns true if it is possible to go forward or backward the specified number of steps.

The `WebView` Methods

- Calling `clearHistory()` clears the browser history.
- Calling `clearCashe()` clears the browser cash memory.

The `WebViewClient` Class

- Each `WebView` object can be connected with a `WebViewClient` object.
- Calling the `setWebViewClient()` method on our `WebView` object passing over a reference for `WebViewClient` object we can put the two connected with each other. The supplied callback object will be notified of a wide range of activities.

The `WebViewClient` Class

- It is common to define a new class that extends `WebViewClient` and overrides the methods we are interested at.
- Overriding the `shouldOverrideUrlLoading()` method we can indirectly have our web view client handling various events that take place within the scope of the `WebView` object.

The `WebViewClient` Class

```
package com.abelski;

import java.util.*;
import android.os.*;
import android.app.*;
import android.webkit.*;

public class WebActivity extends Activity
{
    @Override
    public void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        String str = "";
        str += "<br><a href=\"clock\">system time</a>";
        str += "<br><a href=\"sdk\">sdk version</a>";
        str += "<br><a href=\"developer\">developer name</a>";
        WebView browser = (WebView) findViewById(R.id.webby);
        browser.getSettings().setJavaScriptEnabled(true);
        browser.setWebViewClient(new URLIntercepter());
        browser.loadData(str, "text/html", "UTF-8");
    }
```

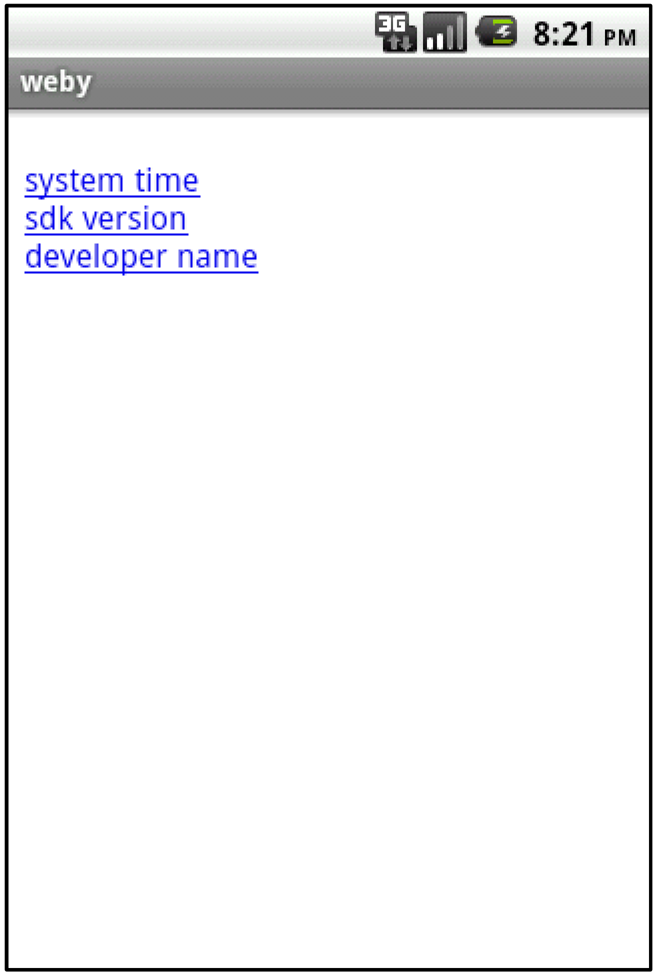
The `WebViewClient` Class

```
public class URLIntercepter extends WebViewClient
{
    @Override
    public boolean shouldOverrideUrlLoading(WebView view, String url)
    {
        if (url.contains("clock"))
        {
            String html = "<h2>" + new Date().toString() + "</h2>";
            view.loadData(html, "text/html", "UTF-8");
            return true;
        }
        else if(url.contains("sdk"))
        {
            String html = "<h2>The SDK version is " +
                Build.VERSION.SDK_INT + "</h2>";
            view.loadData(html, "text/html", "UTF-8");
            return true;
        }
```

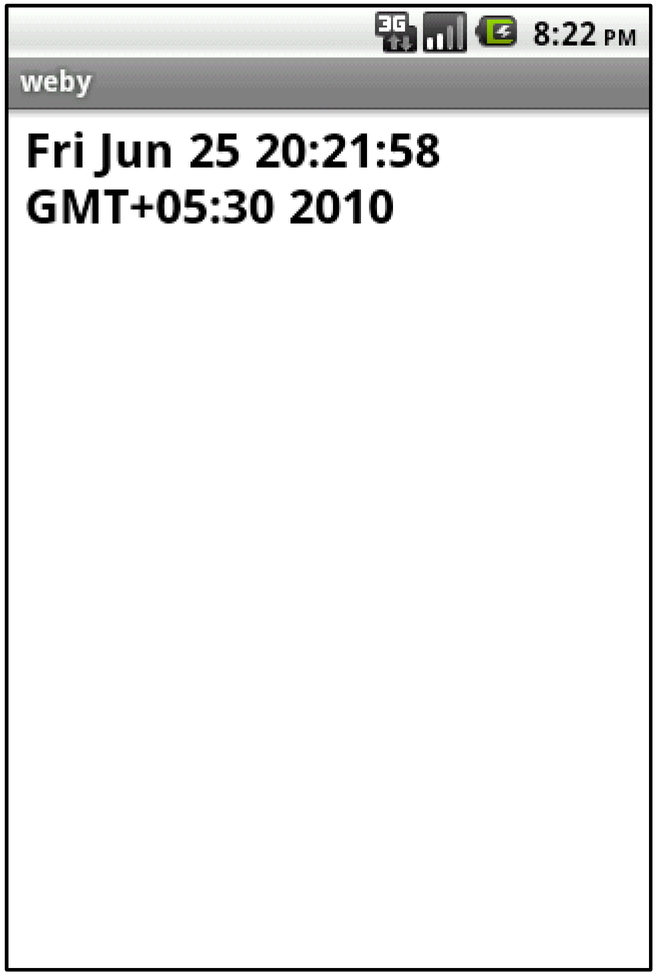
The `WebViewClient` Class

```
            else if(url.contains("developer"))
            {
                String html = "<h2>Developer name is Haim Michael</h2>";
                view.loadData(html, "text/html", "UTF-8");
                return true;
            }
            else
            {
                return false;
            }
        }
    }
}
```

The `WebViewClient` Class

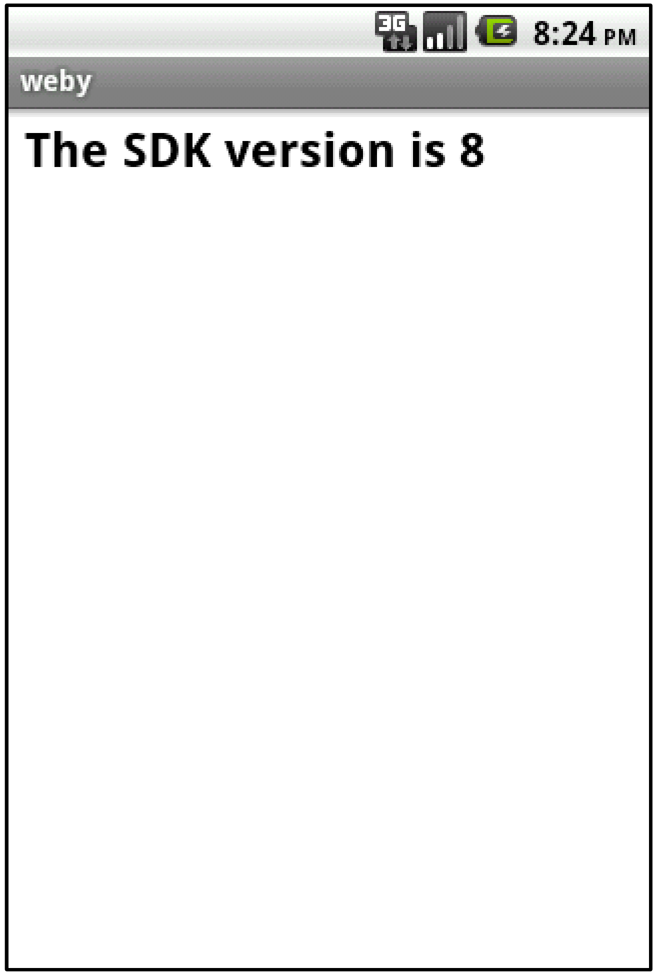


system time
sdk version
developer name

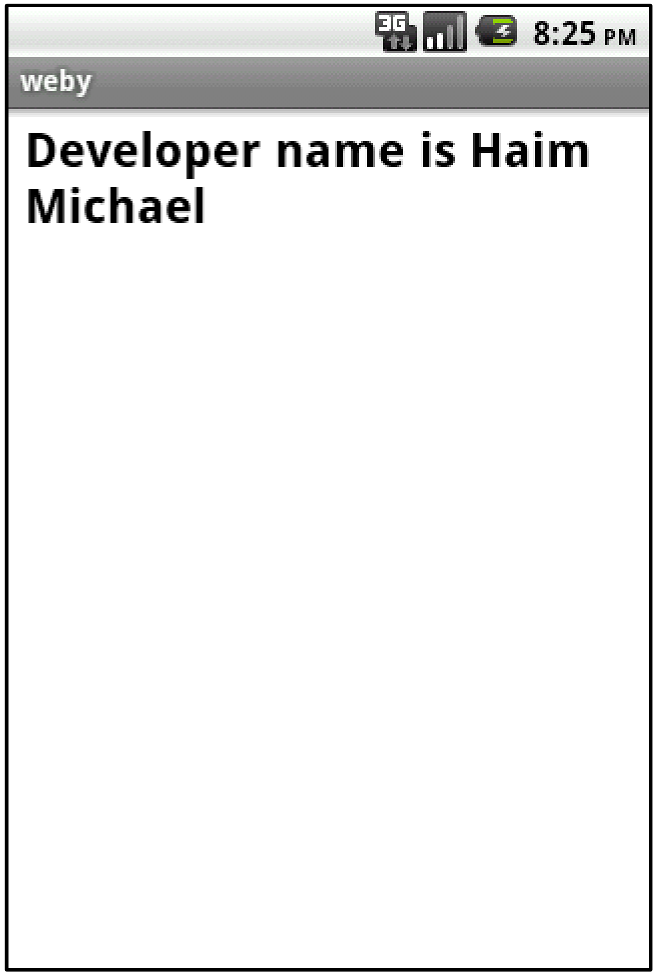


Fri Jun 25 20:21:58
GMT+05:30 2010

The `WebViewClient` Class


8:24 PM
weby

The SDK version is 8


8:25 PM
weby

Developer name is Haim Michael

The `addJavascriptInterface()` Function

- Calling this method we can bind an object to the JavaScript execution code allowing code in JavaScript to call methods on that object.

```
public void addJavascriptInterface(
          Object obj, String interfaceName)
```

The name to use within the Java Script code

Tha Java class instance we want to expose

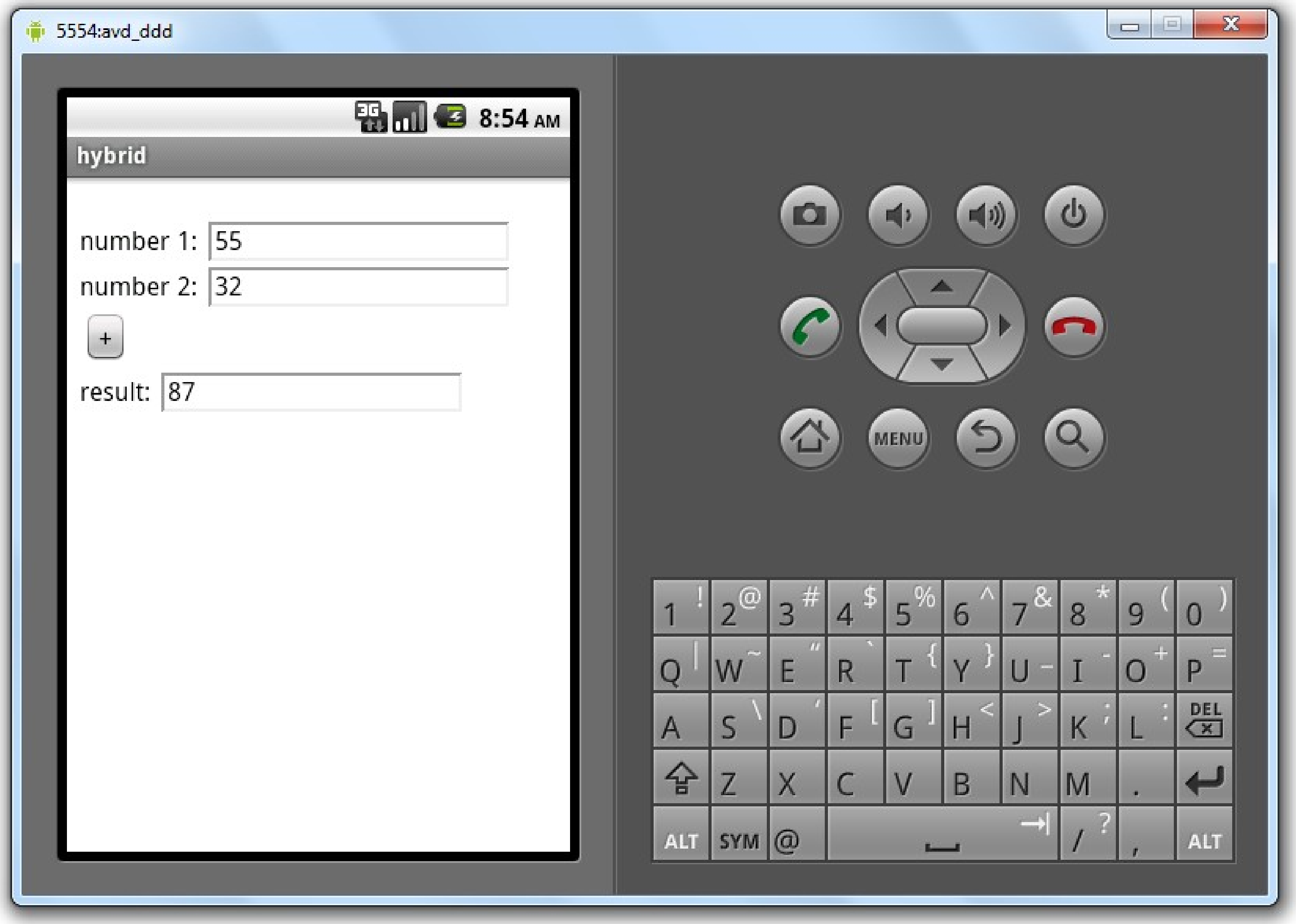
The `addJavascriptInterface()` Function

```
public class HybridActivity extends Activity
{
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState)
    {
        CalculateObject calcObject = new CalculateObject();
        super.onCreate(savedInstanceState);
        WebView webView = new WebView(this);
        webView.loadUrl("http://www.abelski.com/courses/android/simple.html");
        webView.getSettings().setJavaScriptEnabled(true);
        webView.addJavascriptInterface(calcObject, "ob");
        setContentView(webView);
    }
    class CalculateObject
    {
        public int calculateSum(int numA, int numB)
        {
            return numA + numB;
        }
    }
}
```

The `addJavascriptInterface()` Function

```
<html>
    <head>
        <script>
        function calc()
        {
            var a = parseInt(document.myform.num_a.value,10);
            var b = parseInt(document.myform.num_b.value,10);
            var sum = window.ob.calculateSum(a,b);
            document.myform.result.value = sum;
        }
        </script>
    </head>
    <body>
        <form name="myform">
            <br/>number 1: <input type="text" name="num_a"/>
            <br/>number 2: <input type="text" name="num_b"/>
            <br/><input type="button" onclick="calc()" value="+"/>
            <br/>result: <input type="text" name="result"/>
        </form>
    </body>
</html>
```

The `addJavascriptInterface()` Function



number 1: 55
number 2: 32
result: 87