Embedding the WebKit Browser

Other GUI toolkits let you use HTML for presenting information, from limited HTML renderers (e.g., Java/Swing and wxWidgets) to embedding Internet Explorer into .NET applications. Android is much the same, in that you can embed the built-in web browser as a widget in your own activities, for displaying HTML or full-fledged browsing. The Android browser is based on WebKit, the same engine that powers Apple’s Safari web browser.

The Android browser is sufficiently complex that it gets its own Java package (android.webkit). Using the WebView widget itself can be simple or powerful, based on your requirements, as you’ll learn in this chapter.

A Browser, Writ Small

For simple stuff, WebView is not significantly different than any other widget in Android. You pop it into a layout, tell it which URL to navigate to via Java code, and you’re finished.

For example (WebKit/Browser1), here is a simple layout with a WebView:

```xml
<?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/webkit"
            android:layout_width="fill_parent"
            android:layout_height="fill_parent"/>
</LinearLayout>
```

As with any other widget, you need to tell it how it should fill up the space in the layout. In this case, it fills all remaining space.
The Java code is equally simple:

```java
package com.commonsware.android.browser1;

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

public class BrowserDemo1 extends Activity {
    WebView browser;

    @Override
    public void onCreate(Bundle icicle) {
        super.onCreate(icicle);
        setContentView(R.layout.main);
        browser=(WebView)findViewById(R.id.webkit);

        browser.loadUrl("http://commonsware.com");
    }
}
```

The only unusual bit with this edition of `onCreate()` is that we invoke `loadUrl()` on the `WebView` widget, to tell it to load a web page (in this case, the home page of some random firm).

However, we also need to make one change to `AndroidManifest.xml`, requesting permission to access the Internet:

```xml
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.commonsware.android.browser1">
    <uses-permission android:name="android.permission.INTERNET" />
    <application android:icon="@drawable/cw">
        <activity android:name=".BrowserDemo1" android:label="BrowserDemo1">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

If we fail to add this permission, the browser will refuse to load pages. Permissions are covered in greater detail in Chapter 28.

The resulting activity looks like a web browser, but with hidden scrollbars, as shown in Figure 13–1.