



Asignatura: Application Development for Mobile Devices.

Tarea 24: La Clase WebView.

CONCEPTOS

La clase `WebView` es un `View` que muestra páginas web. Esta clase es la base sobre la que se puede ejecutar un navegador web propio, o simplemente mostrar contenidos online dentro de la `Activity`. Utiliza el motor de renderizado `WebKit` para mostrar páginas web e incluye métodos para navegar hacia adelante y hacia atrás a través de una secuencia de enlaces, acercar y alejar la apariencia, realizar búsquedas textuales y mucho más.

Observar que para que la actividad tenga acceso a las páginas web de Internet y cargarlas en un `WebView`, se debe agregar el permiso `INTERNET` al archivo `AndroidManifest.xml`:

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

Esta etiqueta debe ser hijo de un elemento `<manifest>`.

La jerarquía de clases es la siguiente:

```
public class WebView extends AbsoluteLayout implements
    ViewTreeObserver.OnGlobalFocusChangeListener,
    ViewGroup.OnHierarchyChangeListener

java.lang.Object -> android.view.View -> android.view.ViewGroup -> android.widget.AbsoluteLayout ->
    android.webkit.WebView
```

Por definición, un `WebView` no proporciona algún widget similar a un navegador, no permite JavaScript y se ignoran errores los errores de las páginas web. Si solamente se desea mostrar algo de código HTML, como parte de la interfaz de usuario, esto no implica algún problema; el usuario no tendrá que interactuar con la página web y solamente para su lectura, y la página web no tiene que interactuar con el usuario. Si se desea realmente un navegador web completo, entonces se debe invocar la aplicación del navegador con un `Intent` URL que lo muestre en un `WebView`. Por ejemplo:

```
Uri uri = Uri.parse("http://www.example.com");
Intent intent = new Intent(Intent.ACTION_VIEW, uri);
startActivity(intent);
```

Para utilizar un `WebView` en la `Activity`, se incluye la etiqueta `<WebView>` en la plantilla diseño, o configurar una ventana completa `Activity` como un `WebView` durante el `onCreate()`:

```
WebView webview = new WebView(this);
setContentView(webview);
```

Luego se carga la página web deseada:

```
// Simplest usage: note that an exception will NOT be thrown
// if there is an error loading this page (see below).
webview.loadUrl("http://slashdot.org/");

// OR, you can also load from an HTML string:
String summary = "<html><body>You scored <b>192</b> points.</body></html>";
webview.loadData(summary, "text/html", null);
// ... although note that there are restrictions on what this HTML can do.
// See the JavaDocs for loadData() and loadDataWithBaseURL() for more info.
```

Un `WebView` tiene varios puntos de personalización donde se puede cambiar el comportamiento.

Estos son:

- Crear y configurar la subclase `WebChromeClient`. Esta clase se invoca cuando algo afecta la interfaz de usuario, por tanto, el progreso se actualiza y las alertas de JavaScript se envían aquí.



- Crear y configurar la subclase `WebViewClient`. Se invoca cuando algo afecta a los contenidos, por ejemplo, los errores o los envíos de formularios. También se puede interceptar la carga URL aquí (por medio de `shouldOverrideUrlLoading()`).
- Modificar los `WebSettings`, como la habilitación de JavaScript con `setJavaScriptEnabled()`.
- Insertar objetos Java en el `WebView` utilizando el método `addJavascriptInterface` (`Object`, `String`). Este método permite insertar objetos Java en el contexto JavaScript de la página, de modo que puedan ser accedidos por JavaScript en la página.

Enseguida se muestra un ejemplo más elaborado, que muestra el control de errores, la configuración y notificación del progreso:

```
// Muestra el progreso en la barra del título de la actividad, similar a un navegador.
getWindow().requestFeature(Window.FEATURE_PROGRESS);
webview.getSettings().setJavaScriptEnabled(true);
final Activity activity = this;
webview.setWebChromeClient(new WebChromeClient() {
    public void onProgressChanged(WebView view, int progress) {
        // Activities and WebViews measure progress with different scales.
        // The progress meter will automatically disappear when we reach 100%
        activity.setProgress(progress * 100);
    }
});
webview.setWebViewClient(new WebViewClient() {
    public void onReceivedError(WebView view,
        int errorCode, String description, String failingUrl){Toast.makeText(activity,
        "Oh no! " + description, Toast.LENGTH_SHORT).show();
    }
});
webview.loadUrl("http://developer.android.com/");
```

DESARROLLO

EJEMPLO

Paso 1. Crear un nuevo proyecto `MyApplication`. En la carpeta `java/com.example.mipaquete`, abrir y modificar el archivo `MainActivity.java`.

```
import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.view.View.OnClickListener;
import android.webkit.*;
import android.widget.*;
public class MainActivity extends Activity implements OnClickListener{
    Button jbn1, jbn2, jbn3, jbn4;
    WebSettings ws;
    WebView wv;
    EditText jet;
    String s="https://www.google.com/";
    public void onCreate(Bundle b){
        super.onCreate(b); setContentView(R.layout.activity_main);
        jbn1 = (Button) findViewById(R.id.xbn1);
        jbn1.setOnClickListener(this);
        jbn2 = (Button) findViewById(R.id.xbn2);
        jbn2.setOnClickListener(this);
        jbn3 = (Button) findViewById(R.id.xbn3);
        jbn3.setOnClickListener(this);
        jbn4 = (Button) findViewById(R.id.xbn4);
```



```

        jbn4.setOnClickListener(this);
        jet = (EditText) findViewById(R.id.xet);
        wv = (WebView) findViewById(R.id.xwv);
        wv.setWebViewClient(new Cliente());
        ws = wv.getSettings();
        ws.setBuiltInZoomControls(true);
        ws.setJavaScriptEnabled(true);
        ws.setUseWideViewPort(true);
    }
    class Cliente extends WebViewClient{
        public boolean shouldOverrideUrlLoading(WebView view, String url){
            return false;
        }
        public void onPageFinished(WebView view, String url){
            jet.setText(url);
        }
    }
    public void onClick(View v){ int id = v.getId(); switch(id){
        case R.id.xbn1: wv.goBack(); break;
        case R.id.xbn2: wv.loadUrl(s); break;
        case R.id.xbn3: wv.goForward(); break;
        case R.id.xbn4: wv.loadUrl(jet.getText() + ""); break;
    }
    }
}

```

Paso 2. En la carpeta `res/values`, abrir y modificar el archivo `activity_main.xml` con el siguiente código:

```

<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal" >
        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textColor="#000000"
            android:text="@string/opcion" />
        <Button
            android:id="@+id/xbn1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@string/back" />
        <Button
            android:id="@+id/xbn2"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@string/home" />
        <Button
            android:id="@+id/xbn3"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@string/forward" />
    </LinearLayout>

```



```
<Button
    android:id="@+id/xbn4"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Go"/>
</LinearLayout>
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:textColor="#000000"
    android:text="@string/url" />
<EditText
    android:id="@+id/xet"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:hint="Escribir una URL"
    android:textColor="#000000"/>
<WebView
    android:id="@+id/xwv"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:focusable="true" />
</LinearLayout>
```

Paso 3. En la carpeta `res/values`, abrir el archivo `strings.xml` para modificarlo con el siguiente código:

```
<resources>
    <string name="app_name">MisPaginas</string>
    <string name="action_settings">Settings</string>
    <string name="back">Back</string>
    <string name="home">Home</string>
    <string name="forward">Forward</string>
    <string name="internet">URL: </string>
    <string name="url">URL</string>
    <string name="opcion">Opciones</string>
</resources>
```

Paso 4. Abrir el archivo `AndroidManifest.xml` para insertar la etiqueta del permiso de uso de Internet, la cual se indica con el texto en negritas:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest
    xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.escom.miweb">
    <application
        : >
        <activity
            android:name=".MainActivity"
            :
        </activity>
    </application>
    <uses-permission android:name="android.permission.INTERNET" />
</manifest>
```

Paso 5. Por último, ejecutar la aplicación. Al inicio, en la figura 1 se muestra solamente el contenido vacío del `WebView`. Si se selecciona el botón `Home`, se carga la página predeterminada, ver la figura 2, en este caso `http://www.google.com`. Después se pueden cargar otras páginas, ver la figura 3 y posteriormente navegar entre ellas, ver la figura 4, según se desee:



Figura 1. Inicio.



Figura 2. Página inicial.



Figura 3. Selección de otra URL.



Figura 4. Se muestra otro sitio web.

NOTA: Capturar las imágenes de la ejecución del ejercicio en un documento y guardarlo con la sintaxis AlumnoTarea24Grupo.pdf. Enviar el archivo al sitio indicado por el profesor.