PDM Avanzado Comunicacion Descarga de arquivos

Sumario

- 1 Introdución
- 2 Identificar o tipo de rede
- 3 Descargar o arquivo
- 4 Caso Práctico
    - 4.1 Preparación
    - 4.2 Creamos a Activity

Introdución

Nota: Se utilizades un dispositivo real para facer esta práctica teredes que ter conexión a Internet.

Se utilizades o emulador o computador onde estea correndo debe ter conexión a Internet.

O proceso que temos que seguir para descargar un arquivo é:

- Identificar o tipo de conexión para saber se estamos conectados a Internet. Neste punto podemos indicar o usuario que o uso da aplicación pode levar un custo se está conectado pola rede móbil.
- Descargar o arquivo.

Para facer o anterior necesitaremos engadir ó arquivo **AndroidManifiest.xml** os seguintes permisos:

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

Xa que imos acceder a Internet e imos descargar o arquivo na tarxeta SD Externa.

Identificar o tipo de rede

Para obter información sobre a rede necesitamos engadir o seguinte permiso o arquivo de **AndroidManifiest.xml**:

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

Cando estamos conectados cun dispositivo móbil a unha rede temos tres posibilidades:

- Móbil
- Ethernet (cable de rede)
- WIFI

Para obter o tipo de rede ó que estamos conectados deberemos usar un obxecto da clase ConnectivityManager.

Para obtelo, deberemos chamar ó método getSystemService da seguinte forma:

```
ConnectivityManager connMgr = (ConnectivityManager)contexto.getSystemService(Context.CONNECTIVITY_SERVICE);
```

Unha vez feito isto podemos obter información acerca da rede na que estamos conectados mediante un obxecto da clase NetworkInfo:

```
NetworkInfo networkInfo=null;
networkInfo = connMgr.getActiveNetworkInfo();
```

NetworkInfo informa se estamos conectados e o tipo de rede ó que estamos conectados:

```
if (networkInfo != null && networkInfo.isConnected()) {
switch(networkInfo.getType()){
case ConnectivityManager.TYPE_MOBILE:
break;
case ConnectivityManager.TYPE_ETHERNET:
// ATENCION API LEVEL 13 PARA ESTA CONSTANTE
break;
case ConnectivityManager.TYPE_WIFI:
// NON ESTEAS MOITO TEMPO CO WIFI POSTO
// MAIS INFORMACION EN http://www.avaate.org/
break;
}
}
else {
// NON TEMOS REDE
}
```

Nota: Se necesitamos manexar a conexión WIFI podemos facer uso da clase WifiManager: http://developer.android.com/reference/android/net/wifi/WifiManager.html e engadir o permiso android.permission.ACCESS_WIFI_STATE no AndroidManifiest.xml.

Descargar o arquivo

Agora imos resolver o problema de descargar un arquivo dende Internet.

Teremos que utilizar un InputStream para lela e un OutputStream para escribila a disco.

Nota: O manexo de arquivos xa os vimos neste punto: http://manuais.iessanclemente.net/index.php/PDM_Avanzado_Datos_Persistentes_Arquivos

O cartafol onde imos gardala será o predeterminado polo S.O. para gardar imaxes.

O proceso será o seguinte:

- Determinamos a dirección URL para descargar o arquivo:

```
private String IMAXE_DESCARGAR="http://www.aspedrinas.com/imagenes/santiago/santiago1.jpg";
URL url=null;
try {
url = new URL(IMAXE_DESCARGAR);
} catch (MalformedURLException e1) {
// TODO Auto-generated catch block
e1.printStackTrace();
return;
}
```

Se queremos obter só o nome do arquivo da URL podemos poñer este código:

```
String nomeArquivo = Uri.parse(IMAXE_DESCARGAR).getLastPathSegment();
```

- Agora necesitamos ler dende Internet o arquivo a descargar.

A idea é moi sinxela. Temos que facer unha conexión có Servidor. Ó establecer dita conexión podemos especificar unha serie de parámetros coma son:

- Tempo máximo de lectura (en milisegundos).
- Tempo máximo para facer a conexión.

- Método de transmisión (GET ou POST, entre outros).

Todo isto se fai cun obxecto da clase HttpURLConnection:

```
HttpURLConnection conn=null;

conn = (HttpURLConnection) url.openConnection();
        conn.setReadTimeout(10000);             /* milliseconds */
        conn.setConnectTimeout(15000);  /* milliseconds */
        conn.setRequestMethod("POST");
        conn.setDoInput(true);/* Indicamos que a conexión vai recibir datos */

        conn.connect();
```

Ó facer o intento de conexión o servidor web pode darnos unha resposta de que todo é correcto ou un erro (por exemplo, recursos non atopado, que non deixe descargar,...) Comprobaremos por tanto que non tivemos ningún erro:

```
int response = conn.getResponseCode();
        if (response != HttpURLConnection.HTTP_OK){
                        // TRATAREMOS O ERRO
return;
        }
```

Nota: Aínda que dependemos do servidor, podemos obter o tamaño do que queremos descargar da seguinte forma:

```
int fileLength = conn.getContentLength();// Non funciona sempre
```

Isto nos pode servir para modificar unha barra de progreso a indicar canto lle queda por descargar....

En caso de que o servidor non resposnda filelength terá de valor -1.

- Definimos o OutputStream (para gardar o arquivo lido) e un InputStream para ler o contido de Internet:

```
OutputStream os;
        InputStream in;
        ......
os = new FileOutputStream(Environment.getExternalStoragePublicDirectory(Environment.DIRECTORY_PICTURES+File.separator+nomeArquivo));
        in = conn.getInputStream();
```

Nota: Necesitaremos usar try...catch

- Agora só queda ler o contido e escribilo ó mesmo tempo como fixemos na unidade de arquivos:

```
byte data[] = new byte[1024];// Buffer a utilizar
            int count;
            while ((count = in.read(data)) != -1) {
                os.write(data, 0, count);
            }
            os.flush();
            os.close();
            in.close();
            conn.disconnect();
```

- **O PUNTO MÁIS IMPORTATE**: Todo o anterior ten que facerse nun fío separado do principal.

Como se vimos na Unidade de Threads e AsynTask o podemos facer utilizando un Thread ou un AsynTask.

Aclaración: Estamos a amosar como descargar un arquivo calquera de Internet e unha vez baixado facer algunha operación sobre el.

Se queremos descargar unha imaxe podemos cargar directamente un Bitmap dende o InputStream desta forma:

```
Bitmap bitmap = BitmapFactory.decodeStream(in);
```

Caso Práctico

O obxectivo desta práctica é comprobar como podemos descargar un arquivo dende Internet (neste exemplo é unha imaxe).

A dirección URL está posta na propia aplicación do exemplo. O alumno pode cambiala por unha súa pero tendo en conta que hai sitios web que van responder cun erro de conexión (403, forbidden) e outros teñen unha visualización diferente para móbiles que para PC, polo que unha dirección probada dende un PC pode funcionar pero que cando se pon na aplicación móbil pode dar outro erro (307, redirect).

Neste exemplo utilizamos un Thread sen paso de mensaxes.

Isto non será a forma adecuada de implementalo, xa que o lóxico sería premer o botón de Descargar Imaxe e 'informar' a activity cando rematou de descargala.

Unha forma elegante de facelo é utilizar un diálogo de progreso como vimos nas unidades anteriores.



Preparación

Engadimos no AndroidManifiest.xml os seguintes permisos:

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

- Lembrar que no caso de utilizar un emulador será necesario que teña a tarxeta externa SD.

Creamos a Activity

- Nome do proxecto: **UD5_01_Comunicacion**
- Nome da activity: **UD5_01_Comunicacion.java**

Código do layout xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="${relativePackage}.${activityClass}" >

    <ImageView
        android:id="@+id/ud5_imgvwImaxeDescargada"
        android:layout_width="300dp"
        android:layout_height="300dp"
        android:layout_alignParentBottom="true"
        android:layout_centerHorizontal="true"
        android:contentDescription="IMAXE DESCARGADA"
        android:src="@drawable/ic_launcher" />

    <Button
        android:id="@+id/ud5_btnDescargarImaxe"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="34dp"
        android:text="DESCARGAR IMAXE" />

    <Button
        android:id="@+id/ud5_btnVisualizarImaxe"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/ud5_btnDescargarImaxe"
        android:layout_centerHorizontal="true"
        android:text="VISUALIZAR IMAXE" />

</RelativeLayout>
```

Código da clase UD5_01_Comunicacion

Obxectivo: Descargar unha imaxe de Internet.

```
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
```

```
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;

import android.app.Activity;
import android.content.Context;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.net.ConnectivityManager;
import android.net.NetworkInfo;
import android.net.Uri;
import android.os.Bundle;
import android.os.Environment;
import android.util.Log;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.Toast;

public class UD5_01_Comunicacion extends Activity {

public static enum TIPOREDE{MOBIL,ETHERNET,WIFI,SENREDE};
private TIPOREDE conexion;

private final String IMAXE_DESCARGAR="http://www.aspedrinas.com/imagenes/santiago/santiago1.jpg";
private File rutaArquivo;
private Thread thread;

private TIPOREDE comprobarRede(){
NetworkInfo networkInfo=null;

ConnectivityManager connMgr = (ConnectivityManager)getSystemService(Context.CONNECTIVITY_SERVICE);
networkInfo = connMgr.getActiveNetworkInfo();

if (networkInfo != null && networkInfo.isConnected()) {
switch(networkInfo.getType()){
case ConnectivityManager.TYPE_MOBILE:
return TIPOREDE.MOBIL;
case ConnectivityManager.TYPE_ETHERNET:
// ATENCION API LEVEL 13 PARA ESTA CONSTANTE
return TIPOREDE.ETHERNET;
case ConnectivityManager.TYPE_WIFI:
// NON ESTEAS MOITO TEMPO CO WIFI POSTO
// MAIS INFORMACION EN http://www.avaate.org/
return TIPOREDE.WIFI;
}
}
return TIPOREDE.SENREDE;
}


private void descargarArquivo() {
URL url=null;
try {
url = new URL(IMAXE_DESCARGAR);
} catch (MalformedURLException e1) {
// TODO Auto-generated catch block
e1.printStackTrace();
return;
}

    HttpURLConnection conn=null;
String nomeArquivo = Uri.parse(IMAXE_DESCARGAR).getLastPathSegment();
rutaArquivo = Environment.getExternalStoragePublicDirectory(Environment.DIRECTORY_PICTURES+File.separator+nomeArquivo);
try {

conn = (HttpURLConnection) url.openConnection();
        conn.setReadTimeout(10000);             /* milliseconds */
        conn.setConnectTimeout(15000);  /* milliseconds */
        conn.setRequestMethod("POST");
        conn.setDoInput(true);/* Indicamos que a conexión vai recibir datos */
```

```
        conn.connect();

        int response = conn.getResponseCode();
        if (response != HttpURLConnection.HTTP_OK){
return;
        }
        OutputStream os = new FileOutputStream(rutaArquivo);
        InputStream in = conn.getInputStream();
        byte data[] = new byte[1024];// Buffer a utilizar
            int count;
            while ((count = in.read(data)) != -1) {
                os.write(data, 0, count);
            }
            os.flush();
            os.close();
            in.close();
        conn.disconnect();
Log.i("COMUNICACION","ACABO");
}
catch (FileNotFoundException e) {
// TODO Auto-generated catch block
Log.e("COMUNICACION",e.getMessage());
} catch (IOException e) {
// TODO Auto-generated catch block
e.printStackTrace();
Log.e("COMUNICACION",e.getMessage());
}

}

private void xestionarEventos(){


Button btnDescargarImaxe=(Button) findViewById(R.id.ud5_btnDescargarImaxe);
btnDescargarImaxe.setOnClickListener(new OnClickListener() {

@Override
public void onClick(View v) {
// TODO Auto-generated method stub
        thread = new Thread(){

             @Override
             public void run(){
             descargarArquivo();
             }
        };
        thread.start();


}
});
Button btnVisualizarImaxe=(Button) findViewById(R.id.ud5_btnVisualizarImaxe);
btnVisualizarImaxe.setOnClickListener(new OnClickListener() {

@Override
public void onClick(View v) {
// TODO Auto-generated method stub
if ((thread!=null) && (!thread.isAlive())){
ImageView imgviewImaxe = (ImageView)findViewById(R.id.ud5_imgvwImaxeDescargada);
Bitmap bmpImaxe = BitmapFactory.decodeFile(rutaArquivo.getAbsolutePath());
imgviewImaxe.setImageBitmap(bmpImaxe);
}

}
});

}

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_ud5_01__comunicacion);
```

```
            conexion = comprobarRede();
            if (conexion==TIPOREDE.SENREDE){
                 Toast.makeText(this, "NON SE PODE FACER ESTA PRACTICA SEN CONEXION A INTERNET", Toast.LENGTH_LONG).show();
                 finish();
            }

            xestionarEventos();
        }
    }
```

- Liñas 29,30,36-56: Como vimos na explicación inicial, comprobamos o tipo de conexión do noso dispositivo móbil. O método vai devolver un valor dun tipo ENUM (liña 29).
- Liñas 59-108: Descargamos o arquivo como vimos na explicación inicial.
- Liñas 122-124: Xestionamos o evento de Click sobre o botón 'Descargar Imaxe'. Creamos un fío novo de execución e chamamos ó método 'descargarArquivo'.
- Liñas 137-141: Comprobamos que o fío rematou de executarse e nese caso cargamos a imaxe descargada.
- Liñas 153-157: En caso de non estar conectado a ningunha rede finalizamos a activity.

-- Ángel D. Fernández González e Carlos Carrión Álvarez -- (2014).