Mobile solutions

Kocsis Gergely

2019.10.22.

AsyncTask

One of the most hated things a user can face is a UI that freezes after pushing a button that starts a resource expensive task. And meanwhile nothing else can be done.A tipical such case is downloading a file from the internet.

To solve this the Android SDK uses the AsyncTask abstract class with which we can run tasks in the background while we can use the UI for other things.

AsyncTask

One of the most hated things a user can face is a UI that freezes after pushing a button that starts a resource expensive task. And meanwhile nothing else can be done.A tipical such case is downloading a file from the internet.

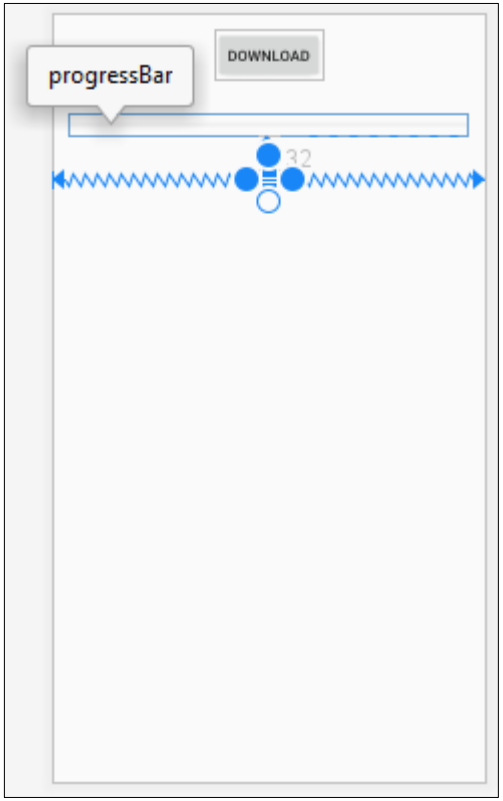
To solve this the Android SDK uses the AsyncTask abstract class with which we can run tasks in the background while we can use the UI for other things.

Create an Adnroid app in which after pushing a button the progress of downloading a file is shown in a progressBar. As a first step instead of really downloading something just make the background task sleep for 1s.

AsyncTask

Create the app and name it Celebration. Design the below layout for it.

After pushing the button the “download” starts. The progress can be seen also in the TextView.



AsyncTask

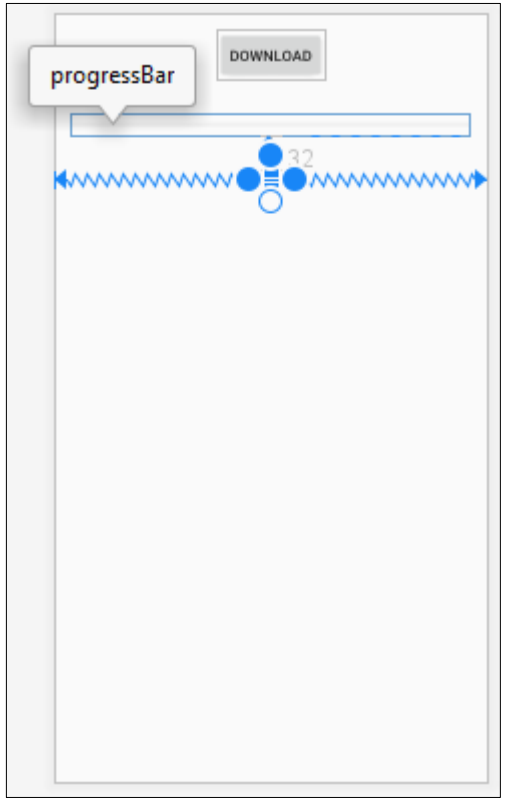
Create the app and name it Celebration. Design the below layout for it.

After pushing the button the "download" starts. The progress can be seen also in the TextView.

Pushing the button calls the startDownload method

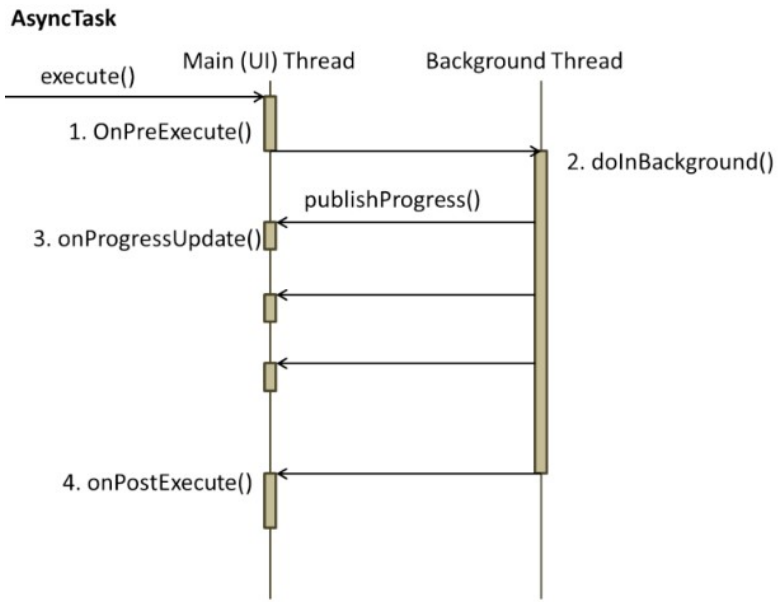
```
public void startDownload(View view) {
}
```

Add respective variables for all three Components at the programmatic side and Initialize them in the onCreate method.



AsyncTask

Two pictures showing the work of AsyncTask.



Source: https://corochann.com/asynctask-usage-summary-341.html

```
public class AsyncTaskTestActivity extends Activity {

    @Override
    public void onCreate(Bundle savedInstanceState) {
        ...

        new MyTask().execute("my string paramater");
    }


    private class MyTask extends AsyncTask<String, Integer, String> {

        @Override
        protected void onPreExecute() {

        }

        @Override
        protected String doInBackground(String... params) {

            String myString = params[0];

            int i = 0;
            publishProgress(i);

            return "some string";
        }

        @Override
        protected void onProgressUpdate(Integer... values) {

        }

        @Override
        protected void onPostExecute(String result) {
            super.onPostExecute(result);

        }
    }
}
```

Source: https://stackoverflow.com/questions/9671546/asynctask-android-example

AsyncTask

Add three WeakReference variables with which we refer the three respective components of the MainActivity. Initialize these in the constructor.

```
private WeakReference<TextView> weakTextView;
private WeakReference<ProgressBar> weakProgressBar;
private WeakReference<Button> weakButton;

DownloadAsyncTask(TextView tv, ProgressBar pb, Button b) {
    weakTextView = new WeakReference<>(tv);
    weakProgressBar = new WeakReference<>(pb);
    weakButton = new WeakReference<>(b);
}
```

AsyncTask

Add three WeakReference variables with which we refer the three respective components of the MainActivity. Initialize these in the constructor.

```
private WeakReference<TextView> weakTextView;
private WeakReference<ProgressBar> weakProgressBar;
private WeakReference<Button> weakButton;

DownloadAsyncTask(TextView tv, ProgressBar pb, Button b) {
    weakTextView = new WeakReference<>(tv);
    weakProgressBar = new WeakReference<>(pb);
    weakButton = new WeakReference<>(b);
}
```

When starting the download inactivate the DOWNLOAD button:

```
@Override
protected void onPreExecute() {
    super.onPreExecute();
    weakButton.get().setEnabled(false);
}
```

AsyncTask

Sleep thor one second in the doInBackgroud method. Set feedback to the UI in every 10ms.

```
@Override
protected String doInBackground(Void... voids) {
    try {
        for (int i = 0; i < 100; i++) {
            Thread.sleep(10);
            publishProgress(i);
        }
    } catch (InterruptedException e) {
        e.printStackTrace();
    }
    return "Download complete";
}
```

AsyncTask

Show the progress in the onProgressUpdate method both on the ProgressBaron and on the TextView.

Whenn finishing the AsyncTask print out the the download is completed and make the Down load button enabled again.

```
@Override
protected void onProgressUpdate(Integer... values) {
    super.onProgressUpdate(values);
    weakTextView.get().setText("" + values[0] + "%");
    weakProgressBar.get().setProgress(values[0]);
}

@Override
protected void onPostExecute(String result) {
    weakTextView.get().setText(result);
    weakButton.get().setEnabled(true);
}
```

AsyncTask

Start the new AsyncTask when pushing the button.

```
//MainActivity
public void startDownload(View view) {
    new DownloadAsyncTask(progressTextView, progressBar,
downloadButton).execute();
}
```

We are finished. Run the application!

AsyncTask

Upgrade the application so that it really downloads an mp3 file from the internet. After that let the user play the song.

Add a Play button to the UI that is set at the moment to be inactive. When pushing the button the playSong starts. Add a respective field to the class and initialize it in the consructor.

```
private Button playButton;
...
playButton = findViewById(R.id.playButton);
...
public void playSong(View view) {
}
```

AsyncTask

Alakítsuk át a doInBackgroud törzsét az alábbiak szerint:

```
int count;
try {
    URL url = new URL("https://irh.inf.unideb.hu/~kocsisg/song.mp3");
    URLConnection connection = url.openConnection();
    connection.connect();
    int lenghtOfFile = connection.getContentLength();
    // input stream to read file - with 8k buffer
    InputStream input = new BufferedInputStream(url.openStream(),10*1024);
    // Output stream to write file
    OutputStream output = new FileOutputStream("/data/user/0/
                                hu.unideb.inf.mobil.celebration/files/song.mp3");
    byte data[] = new byte[1024];
    long total = 0;
    while ((count = input.read(data)) != -1) {
        total += count;
        publishProgress((int) ((total * 100) / lenghtOfFile));
        output.write(data, 0, count);
    }
    output.flush();
    output.close();
    input.close();
} catch (Exception e) {
    Log.e("Error: ", e.getMessage());
}
return "Download complete";
```

Package name

AsyncTask

We also need to set up lesz az AndroidManifest.

```
<!-- Permission: Allow application to connect to Internet -->
<uses-permission android:name="android.permission.INTERNET" />
<!-- Permission: Allow application to write to SDCard -->
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" /
>

<application
    android:usesCleartextTraffic="true"
```

Add the following line to the onPostExecute.. To do this we need weak reference just as in the befor cases.…

```
weakPlayButton.get().setEnabled(true);
```

AsyncTask

Play music:

```
private MediaPlayer mPlayer;

public void playSong(View view) {
    mPlayer  = new MediaPlayer();
    try {
        mPlayer.setDataSource(
           "/data/user/0/hu.unideb.inf.mobil.celebration/files/song.mp3");
        mPlayer.prepare();
        // Start playing the Music file
        mPlayer.start();

    } catch (Exception e) {
        Log.e("IOEX", e.getMessage());
    }
}
```