Leer Información de Internet

Leer de Internet

Cuando necesitamos realizar acciones en nuestra aplicación donde por ejemplo, debemos obtener información de internet, tenemos que hacerlo de tal forma que el realizar la operación no dificulte el uso de la aplicación o termine bloqueando la misma.

Leer de Internet

Ejemplo, si deseo presentar un contenido de internet donde no solo hay texto a mostrar sino también una foto, el proceso de descarga de la imagen puede demorar unos segundos dependiendo del nivel de conexión que posea el usuario.

Si no se realiza en forma adecuada, la pantalla podría quedar “bloqueada” mientras se descarga la imagen, razón por la cuál el usuario pudiese pensar que la aplicación no funciona e inclusive el mismo sistema operativo pueda terminar por bloquearla.

Leer de Internet

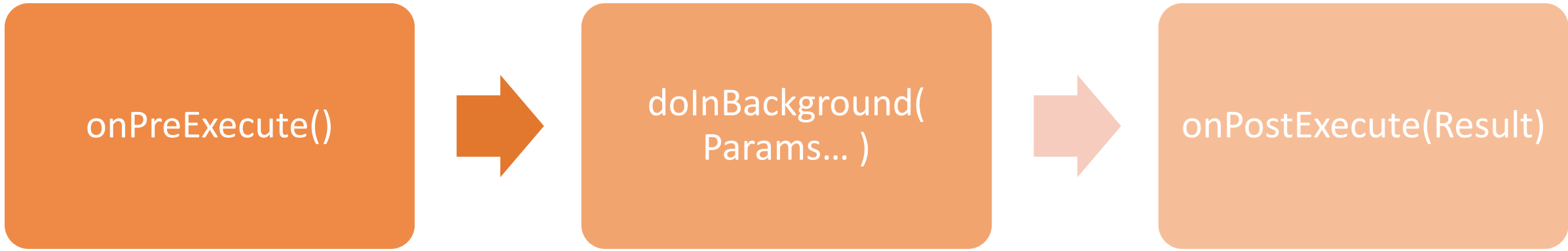
Para operaciones pequeñas, de no mucho tiempo de duración, Android nos proporciona una clase llamada AsyncTask (Tarea Asincrónica, si busco una traducción literal).

Leer de Internet

Esta clase no constituye un “framework” de manejo de hilos en sí y no debe ser usada como tal. Pero para los fines que comenté anteriormente, donde quizás se requiere poder realizar operaciones “cortas” y que se presenten en forma amigable al usuario, esta clase es ideal y es para lo que fue creada.

Leer de Internet

Debido a que esta clase define básicamente el proceso de ejecución de un hilo posee un ciclo de vida que se explica en 3 etapas:



Adicionalmente, este método puede hacer uso de publishProgress(Progress...) como una forma de indicar progreso.

Leer de Internet

onPreExecute()
Se invoca justamente antes de que la tarea principal se ejecute. Suele usarse para inicializar las variables que se usarán en la tarea principal e inclusive inicializar el indicador de progreso.

doInBackground(Params...)
Se invoca inmediatamente luego del onPreExecute y aquí es justamente donde se realiza la tarea principal. Este método toma los parámetros que se le pasan, realiza la operación que se programe y retorna el resultado en el siguiente paso.

onPostExecute(Result)
Es invocado por el hilo cuando se finaliza el proceso que realiza "doInBackground" junto con el resultado obtenido.

Leer de Internet

AsyncTask tiene 3 tipos de datos genéricos que requiere recibir en su constructor.

Params, indica el tipo de parámetro que la clase recibirá para comenzar la ejecución.

Progress, indica el tipo de dato que se utilizará publicar la información de progreso del proceso que se está realizado de fondo.

Result, indica el tipo de dato que la operación de fondo retornará como resultado.

Leer de Internet

Es decir, cuando creamos una clase que extiende AsyncTask, debemos definir estos 3 tipos de datos. Los dos primeros (Param y Progress) no son mandatorios, en ese caso se debe colocar “Void”. El único que es mandatorio es “Result”, lo cuál es obvio ya que si vamos a realizar una operación de hilo es porque algún resultado buscamos obtener.

A continuación, algunos ejemplos de como declaramos esta clase:

private class MyTask extends AsyncTask<Void, Void, String> { ... }

private class MyTask extends AsyncTask<String, Integer, Boolean> { ... }

Leer de Internet

```
private class DownloadFilesTask extends AsyncTask<URL, Integer, Long> {

    protected Long doInBackground(URL... urls) {
        int count = urls.length;
        long totalSize = 0;
        for (int i = 0; i < count; i++) {
            totalSize += Downloader.downloadFile(urls[i]);
            publishProgress((int) ((i / (float) count) * 100));
            // Escape early if cancel() is called
            if (isCancelled()) break;
        }
        return totalSize;
    }

    protected void onProgressUpdate(Integer... progress) {
        setProgressPercent(progress[0]);
    }

    protected void onPostExecute(Long result) {
        showDialog("Downloaded " + result + " bytes");
    }
}
```

Leer de Internet

```
new DownloadFilesTask().execute(url);
```