Background Tasks and Services

[Group 1] Hanh Tran
hanh.usth@gmail.com
20 Nov 2016

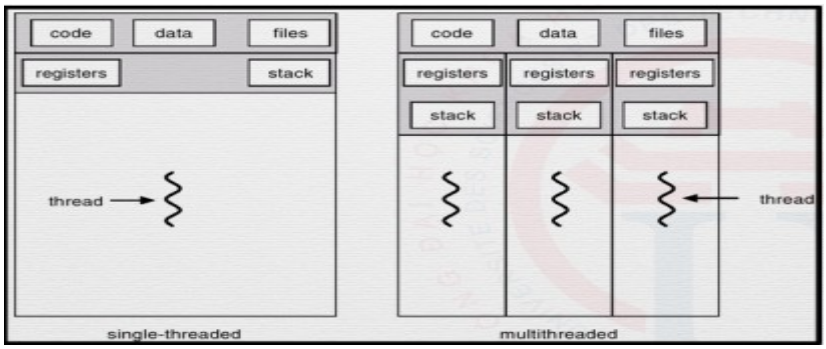
I. Chapter objectives

- Have basic understanding about background tasks and services.
- In this chapter, we study about threading by answer these questions: What is a thread? Why thread? Threads vs applications.

II. Background Task and Service

Threading

- Definition: a kind of subprocess
    - Example: In the applications, 1 thread play the UI (main thread) and 1 thread play the music (worker thread)
- Types:
    - Description: Single-threaded vs multithreaded
    - Threads share the memory: code, data and files
    - Each thread does each task
    - Example:
        - In Firefox: there is one thread to process the iframe and share so threads in the same process can easily access each other



- Why thread?
    - Threads can be executed in the same time, same process and faster than to create new process.

Pros	Cons
• Better CPU utilization • Separation of tasks • Responsiveness	• Complication • Synchronization • Thread pool. . .

- Context switches: very important to provide smooth UI.
    - In multitasking OS: at anytime there can be many programs running
- Why not thread?
    - Synchronization
        - 2 clients access to the same region of memory at the same time can lead to inconsistency (Example: When 2 clients share the same bank account, 1 saves $2000 but 1 spends $1000)
    - Thread pool
    - Complication
        - Architecture
        - Load balancing (Example: If we create each thread to serve 1 client at the same time, it takes much more time to decide what next to do rather than to work itself. So we cannot create too many threads)
- Android Thread Model

Main thread	**Worker threads**

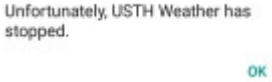
<table>
<tr>
<td>

- <u>Description:</u>
  - Drawing widgets
  - Dispatching user inputs
  - Widget toolkit not thread-safe
  - Don’t slow things on main thread

- <u>Example:</u> Calculation like image processing may cause the apps not responding.

</td>
<td>

- <u>Description:</u>
  - Don’t manipulate Views on worker thread
  - Crash



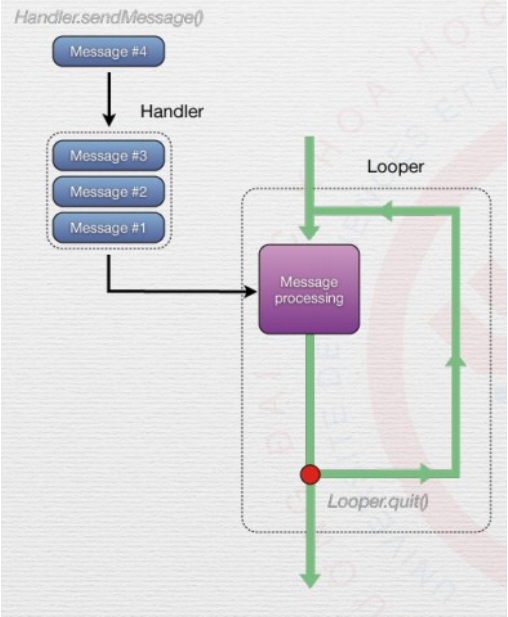
- <u>Example:</u> Create new worker thread

```
public class WeatherActivity extends Activity {

    @Override
    public void onCreate(Bundle savedInstanceState) {
        Thread t = new Thread(new Runnable() {
            @Override
            public void run() {
                // do something heavy in the new thread.
                // don't access UI Views here.
            }
        });
        t.start();
    }
}
```

</td>
</tr>
</table>

- Handler
  - <u>Description</u>:
    - A way to communicate with main thread
    - **Handler.handleMessage()** is executed on main thread
    - **Handler.sendMessage()** is called in worker thread
  - Example: The process:



2. Background Tasks

- Definition: “AsyncTask” is an encapsulation of Handler and Thread that also allows the worker thread to report its work progres to the UI
- 3 Generic Types: AsyncTask<Param, Progress, Result> (Example: AsyncTask<String, Interger, Bitmap>)
  - Params: param type to pass to the worker thread
  - Progress: type to report progress back
  - Result: result type to be delivered
- Methods:
  - [optional] onPreExecute(): for preparation
  - [required] doInBackground(): do the real work
  - [optional] onProgressUpdate(): do updating progress to UI
  - [optional] onPostExecute(): for delivering results