Process and multithreads

Program: A program is a set of instructions and data stored in an executable file.

Process: A process is a program in execution/action.

When a program becomes a process, OS assigns resources to it

- Program Counter
- CPU registers
- Stacks (used for function call, parameter, local variables etc)
- States (Ready, running, waiting, terminated)

(প্রোসেস এর নিজস্ব মেমোরি থাকে, Resource থাকে।)

Thread: Thread is a single flow of execution inside a process. Multiple threads can run concurrently within same process. Threads in a process share the same virtual memory address space.

chatGPT

Key features of thread: Shared resources within the same process.

All threads in a process share:

- Virtual memory
- Global Variables
- Open files
- Heap memory

Each thread has its own:-

- Program Counter
- Stack
- Local variables
- CPU registers

chatGPT

Process	Thread
Independent execution unit	Execution unit inside a process
Has own memory	Shares memory with other threads
Slower inter-process communication	Faster intra-process communication

Why use Threads?

ChatGPT

(i) improve performance with parallelism

(ii) efficient resource usage.

Process creation and removal:

Creation: process is automatically created by Android System When first component of an app is run and there is currently no process for that App.

Removal: Remove old processes to reclaim memory for new or more important processes.

- **Which process should be killed?**

Process Hierarchy (High to Low Priority)

Priority	Type	Description
1	Foreground	On-going interactions; activity, service, broadcast
2	Visible	Not in the foreground, but visible
3	Service	Playing music, downloading stuffs
4	Background	onStop() called, not visible, LRU kill → least recently used
5	Empty	Lowest priority ← kill them first

- **Basic Thread:**
  - Acts as like usual Java threads.
  - Can't act directly on external UI objects

calledFromWrongThreadException → Throws the Exception

Only the original thread that created a view hierarchy can touch the views.

- Can't be stopped by destroy() nor stop()

Uses instead interrupt() or join() (by case)

- Only the Main(UI) thread can update the UI. (user interface)

- 2 main ways to create and Run threads

→ Providing a new class that extends Thread and overriding its run() method.

```
class MyThread extends Thread {
        public void run(){
          // Your background task here
        }
}

MyThread t = new MyThread();
t.start(); // Starts the thread
```

→ Providing a new Thread instance with a Runnable object during its creation

```
Runnable myrunnable = new Runnable() {
    public void run(){
      // Your background task here
    }
};
Thread t = new Thread(myRunnable);
t.start(); // Start the thread
```

→ In both cases, ~~calling~~ .start() method must be called to actually execute the new Thread.

Base thread example:-1

```
protected void startDownloadThread(){
    Thread t = new Thread(){
        public void run(){
            mResult = "This is new Result";
        }
    };
    t.start();
}
```

Base Thread Example-2:

```
class Multi3 implements Runnable{
    public void run(){
        System.out.println("thread is running...");
    }
    public static void main(String args[]){
    Multi3 m1 = new Multi3();
    //Using the constructor Thread(runnable r)
    Thread t1 = new Thread(m1);
    t1.start();
    }
}
```

Main Thread: Created when App is launched. Also called UI thread.

Problem: UI threads (Main thread) is responsible for

ChatGPT
- → Drawing the screen, Input event handling
- → Handling touch and click events
- → Running lifecycle methods, light calculation

If you run long tasks here (like - heavy loops, database queries, file processing, or network calls):—

→ the UI freezes after ~5 seconds, Android shows ANR (Application Not Responding).

Worker Thread: A background thread that performs long-running / time-consuming tasks, so that it can make UI thread light-weight and responsive (Prevent ANR errors).

- **Limitations:** Cannot access UI components (e.g. Views, Button, Textview) from another thread. (cannot directly update UI components).

```
public void onClick(View v){                          → Runs in UI thread
    new Thread(new Runnable(){                        Runs in the new worken thread
        public void run(){
            Bitmap b = loadImageFromNetwork("http://example.com/i.png");
            mImageView.setImageBitmap(b);
        }
    }).start();
}
```

Child thread থেকে UI element update allow করে না

Do not try this (accessing an ImageView which was created in UI thread and now being accessed in a worken thread)

→ Do not access Android UI toolkit from outside the UI thread.

Solution-1: Handler (Main thread / UI thread)

- A main thread handler object schedules messages and runnable to be executed at some point in the future.
- Works by putting messages or Runnable into the UI thread's message queue.

Handler.post()
Handler.postDelayed()

Example:

```
public class MainActivity extends Activity{
    Handler myHandler;
    @Override
    protected void onCreate(Bundle savedInstanceState){
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        myHandler = new Handler();
        new Thread(new Runnable(){
            @Override
            public void run(){ //perform long-running task here
                myHandler.post(new Runnable(){
                    @Override
                    public void run(){
                        mProgressBar.setProgress(
                            currentProgressCount);
                    }
                });
            }
        }).start();
    }
}
```

Handler Post Object : Post Delayed

```
public class mainActivity extends Activity{
    Handler myHandler;
@override
  public static void onCreate(Bundle savedInstanceState){
  super.onCreate(savedInstanceState);
  setContentView(R.layout.activity_main);
  myHandler = new Handler();
  new Thread(new Runnable(){
    @override
      public void run(){
        //perform long running task here
    myHandler.postDelay(new Runnable(){
    @override
      public void run(){
       mProgressBar.setProgress(currentProgressCount);
      }
    },1000);
  }
 }).start();
}}
```

1000 ms → 1s

Soln-2: Posting Runnable objects to the main View

- To add an action into a queue performed on a different thread.