Threads

Repo: https://github.com/karlmorris/AndroidThreads
Overview

- Purpose of threads in Android
- Uses and restrictions
- Message passing
Threads

- Operations in an application can at times disrupt its flow and impact user experience
  - Heavy operations that could slow down or pause the main interface
  - Blocking operations
- If the desired operation has no requirement that the application wait on its completion, it is a candidate to be run in the background
- One mechanism to implement background operation execution is threads
Threads

- Concurrent unit of execution
  - Threads run asynchronously
- Each running application has at least one thread – Main
- Useful for offloading work from main (UI) thread
Threads cont.

- In some cases using threads is mandatory
  - Network operations
- They have no direct access to View objects in main thread
  - Attempts to access views in the main thread will throw an exception
- Must pass messages in order to interact with UI
Threads cont.

- Two ways to create a new thread
  - Subclass the Thread class and override `run()`
  - Create a new thread and pass it a `Runnable` object
- After creating, call `start()` to run thread
- Thread class contains mechanisms for thread control, such as pausing execution.
Handlers

- Allows a worker thread to send and process messages (or code) on the main (UI) thread.
- Each handler belongs to the thread that created it.
Handlers cont.

• Have two main uses
  – Schedule messages and runnables to be executed at some time in the future
  – Enqueue an action to be performed on some thread other than your own

• The latter is useful for interacting with app UI
Handlers cont.

Basic implementation

// Main thread
Handler messageHandler = new Handler(new Handler.Callback() {
    @Override
    public boolean handleMessage(Message msg) {
        textView.setText(msg.arg1);
        return false;
    }
});

-----------------------------------------------------------------------------------------

// Worker thread
Message msg = Message.obtain();
msg.arg1 = messageContent;
messageHandler.sendMessage(msg);