Handlers and Runnables

Mr. Michaud
Marist School

Creating a Timing Loop in Android Java

- Given that Android Java is event driven . . .
- We need a method to have the Activity ‘self call’ a function repeatedly.
  - Display a Series of Images
  - Timed Events
  - Animation Loops / Game Loops
- We achieve this with a “Handler – Runnable” combination.

Handler

- A Handler allows you to send and process Message and Runnable objects associated with a thread's MessageQueue. Each Handler instance is associated with a single thread and that thread's message queue. When you create a new Handler, it is bound to the thread / message queue of the thread that is creating it -- from that point on, it will deliver messages and runnables to that message queue and execute them as they come out of the message queue.

From: http://developer.android.com/reference/android/os/Handler.html#post(java.lang.Runnable)

Runnable

- Definition: (From: http://docs.oracle.com/javase/7/docs/api/java/lang/Runnable.html)
  - public interface **Runnable.** The Runnable interface should be implemented by any class whose instances are intended to be executed by a thread.
  - The class must define a method of no arguments called run.
  - This interface is designed to provide a common protocol for objects that wish to execute code while they are active.
  - For example, Runnable is implemented by classThread. Being active simply means that a thread has been started and has not yet been stopped.

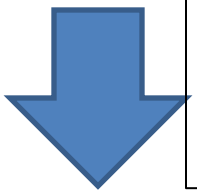
Handler / Runnable Structure

- Handler h = new Handler()
  - Declared in Fields and initialized in onCreate()
- Constant value ‘RATE’ – timing in milliseconds
- Runnable r = new Runnable()
  - Private Inner Class in activity
- Third Function that contains the h.postDelayed(); call.

Function / Handler / Runnable Cycle

1. External Function Call
2. Desired Action
3. Check Condition
4. Call the Runnable with Delay
5. Runnable Calls Function
6. If Check Condition is False - exit

Example

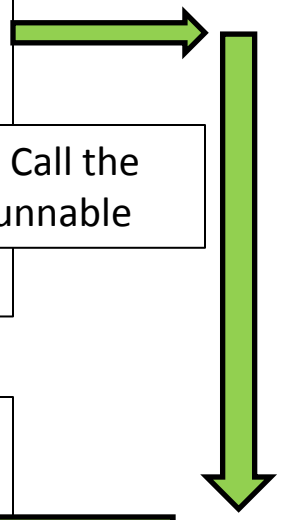


1. Function called first time from external event (Button, SeekBar . ..

```
// Function to be repeated (Loop)
public void displayImages() {
  // Action desired
  imageView.setImageResource(image[index]);
  index++; // increase counter
  // Check Condition
  if (index < image.length) {
    h.postDelayed(r, RATE); // Call the Runnable
  } // end if
  else {
    // Function Loop Completed
  } // end else
} // end displayImages
```

2. Desired action executed

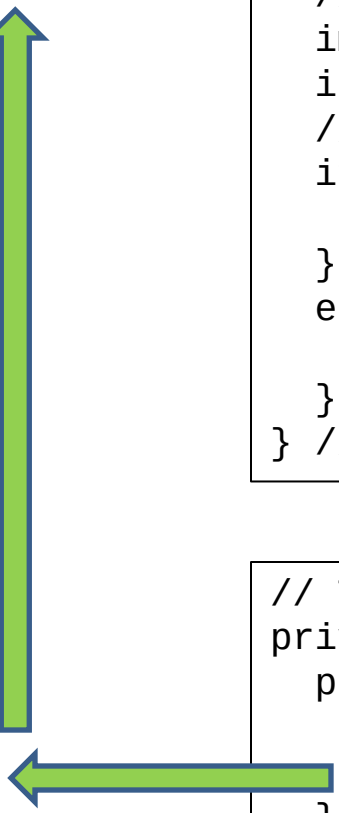
3. Check condition



4. Call the Runnable

6. Exit when done.

```
// The Runnable r
private Runnable r = new Runnable() {
  public void run() {
    // Call the Function to be looped
    displayImages()
  } // end run
}; // end private Runnable r
```



5. Function called again