CS 696 Mobile Phone Application Development
Fall Semester, 2009
Doc 10 Threads
Oct 1, 2009

References

The Busy Coder's Guide to Android Development, V2.1, Mark L. Murphy

Code Quality

Formating & Names

```
public void cal(){
	//messageText01.setText(messageText.getText());
	Double billAmount = null;
	Double tipPercent = null;
	Double totalAmount = null;
	if(messageText.getText().toString().length() < 1)

	{
		clr();

	}



	else
	{
```

Names

```
messageText = (EditText) findViewById(R.id.edit);
messageText01 = (EditText) findViewById(R.id.EditText01);
messageText02 = (EditText) findViewById(R.id.EditText02);
cal = (Button) findViewById(R.id.Cal);
clr = (Button) findViewById(R.id.Clear);
```

Threads

Activity code runs in UI thread

Do not interact with Views in your threads

Android Background Tools

- Java threads
- Handler
  - Messages
  - Runnables
- AsyncTask

- Services

Handler

Attached to thread it is created in

Processes
- Message objects
- Runnable objects

Main Uses

Schedule messages/runnables to be executed as in the future

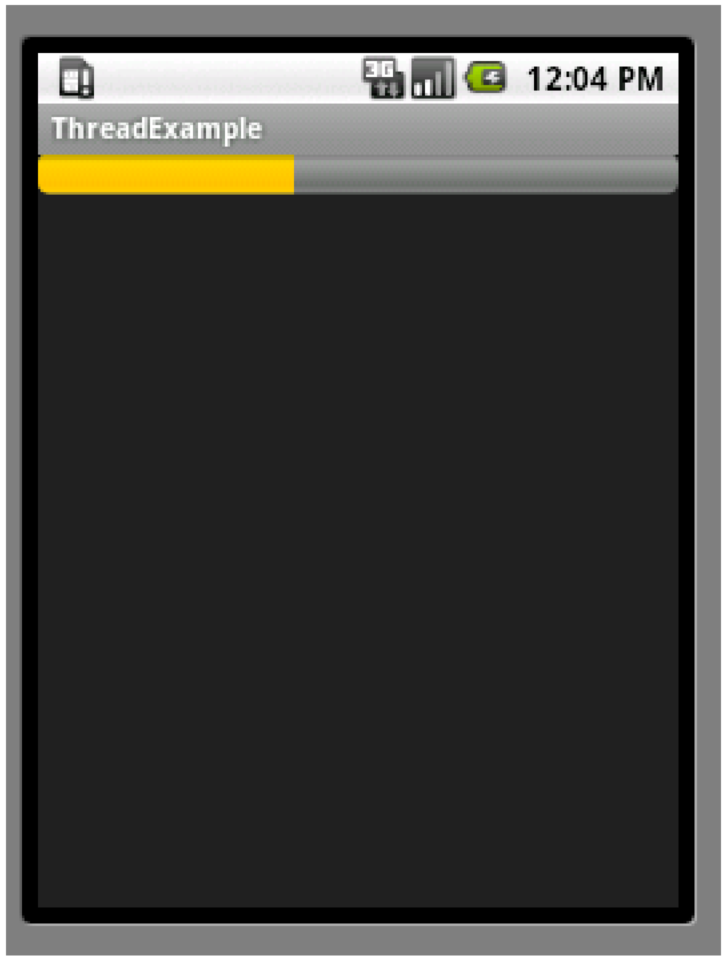
Enqueue an action to be performed on a different thread

Handler Scheduling

post(Runnable)
postAtTime(Runnable, long)
postDelayed(Runnable, long)
sendEmptyMessage(int)
sendMessage(Message)
sendMessageAtTime(Message, long)
sendMessageDelayed(Message, long)

ProgressBar Example

Just shows a progress bar progressing



ThreadExample

```
public class ThreadExample extends Activity {
	ProgressBar progressView;
	boolean isRunning = false;

	Handler handler = new Handler() {
		public void handleMessage(Message empty) {
			progressView.incrementProgressBy(5);
		}
	};

	public void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.main);
		progressView = (ProgressBar) findViewById(R.id.progress);
	}

	public void onStop() {
		super.onStop();
		isRunning = false;
	}
```

ThreadExample

```
    public void onStart() {
        super.onStart();
        progressView.setProgress(0);

        Thread background = new Thread(new Runnable() {
            public void run() {
                try {
                    for (int i = 0; i < 20 && isRunning; i++) {
                        Thread.sleep(1000);
                        handler.sendMessage(handler.obtainMessage());
                    }
                } catch (Throwable t) {// just end    }
            }
        });
        isRunning = true;
        background.start();
    }

}
```

Sending Text Messages to the future

Rather than use a thread use sendMessageDelayed

Sends data in the message using Bundle

Sending Text The Hard Way

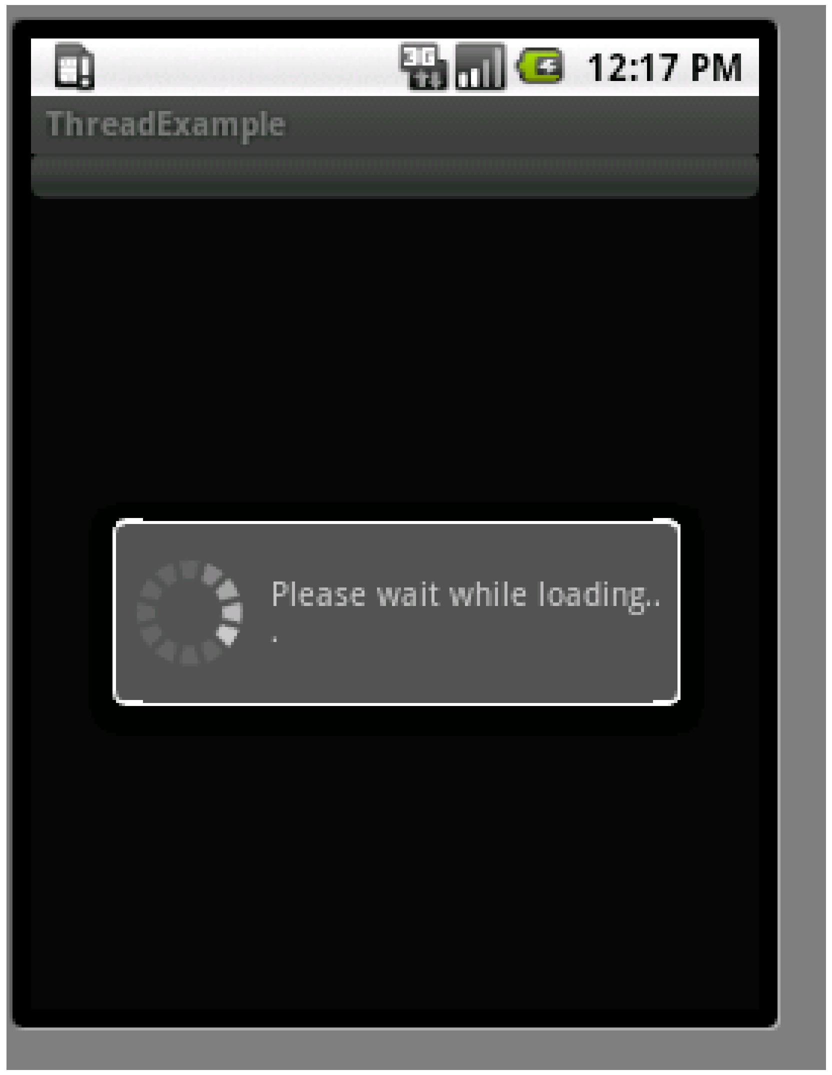
```
public class ThreadExample extends Activity {
	Handler handler = new Handler() {
		public void handleMessage(Message word) {
			String text = word.getData().getString("key");
			Toast.makeText(ThreadExample.this, text, Toast.LENGTH_SHORT).show();
		}
	};

	public void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.main);
	}

	public void onStart() {
		super.onStart();
		String[] text = { "Bat", "cat", "dat", "fat", "hat", "mat" };
		for (int i = 0; i < text.length; i++) {
			Bundle data = new Bundle();
			data.putString("key", text[i]);
			Message word = new Message();
			word.setData(data);
			handler.sendMessageDelayed(word, 1000 * (i + 1));
		}
```

Read the documentation to find the easy way

Progress Dialog

Displays a Progress Dialog

Uses Message what to transmit data



Please wait while loading...

ThreadExample

```
public class ThreadExample extends Activity {
    ProgressDialog waitDialog;
    private static final int WAIT_DIALOG_KEY = 0;
    Handler handler = new Handler() {
        public void handleMessage(Message command) {
            if (command.what == 0)
                showDialog(WAIT_DIALOG_KEY);
            else
                waitDialog.dismiss();
        }
    };

    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
    }
```

ThreadExample

```
    protected Dialog onCreateDialog(int id) {
        switch (id) {
        case WAIT_DIALOG_KEY: {
            waitDialog = new ProgressDialog(this);
            waitDialog.setMessage("Please wait while loading...");
            waitDialog.setIndeterminate(true);
            waitDialog.setCancelable(true);
            return waitDialog;
        }
        }
        return null;
    }

    public void onStart() {
        super.onStart();
        Message on = new Message();
        on.what = 0;
        handler.sendMessageDelayed(on, 1000);
        Message off = new Message();
        off.what = 1;
        handler.sendMessageDelayed(off, 8000);
    }
}
```

AsyncTask

AsyncTask

Replaces threads & Messages

Android 1.5

Subclass AsyncTask

onPreExecute()
- Run in UI thread
- Done first

doInBackground(Params...)
- Run in seperate thread

publishProgress(Progress...)
- Call in doInBackground() to register progress

onProgressUpdate(Progress...)
- Run in UI thread
- Called by publisheProgress

onPostExecute(Result)
- Run in UI thread
- Run after doInBackground ends

Rules

The AsyncTask sublcass instance must be created on the UI thread

execute(Params...)
- Starts the task
- Must be invoked on the UI thread

Do not call manually
- onPreExecute(), onPostExecute(Result), doInBackground(Params...), onProgressUpdate(Progress...)

The task can be executed only once

AsyncTask Types

private class SampleTask extends AsyncTask<Params, Progress, Result>

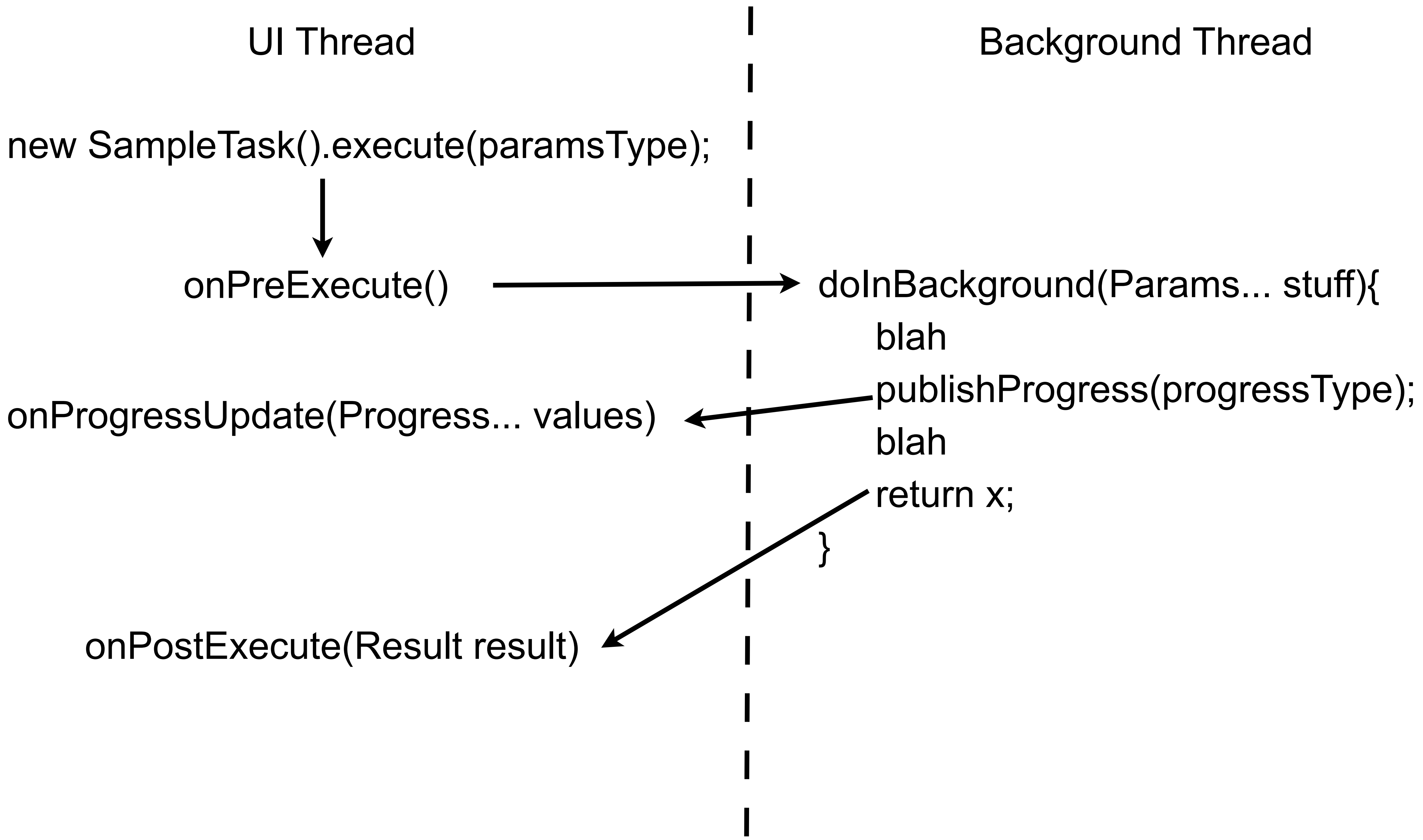
Params
- Type of argument for
- doInBackground()
- execute()

Progress
- Type of argument for
- publishProgress()
- onProgressUpdate()

Result
- Return type for doInBackground()
- Type of argument for onPostExecute()

How it Works

```
private class SampleTask extends AsyncTask<Params, Progress, Result>
```

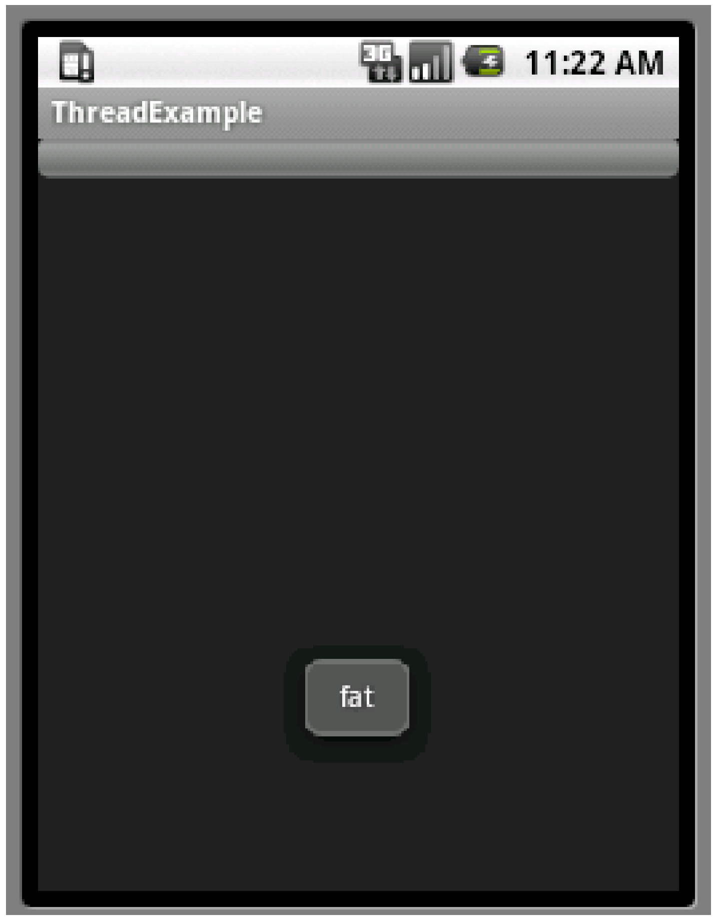


```
new SampleTask().execute(paramsType);

onPreExecute()

onProgressUpdate(Progress... values)

onPostExecute(Result result)
```

```
doInBackground(Params... stuff){
    blah
    publishProgress(progressType);
    blah
    return x;
}
```

Example

Loops in the background and displays Toast



ThreadExample

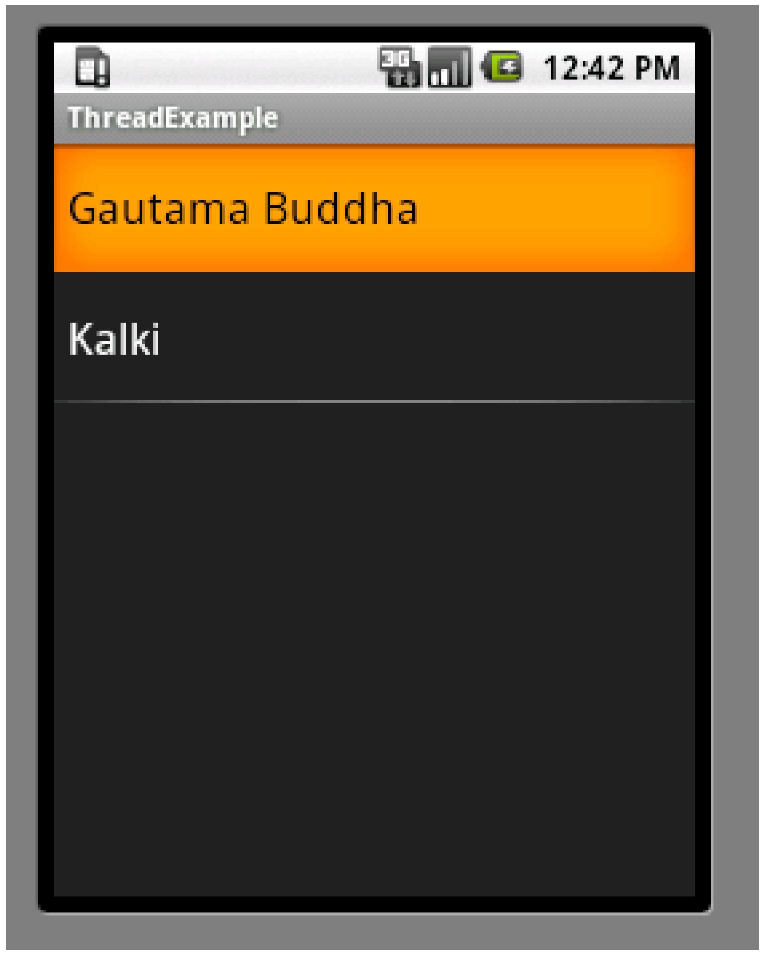
```
public class ThreadExample extends Activity {

    private class SampleTask extends AsyncTask<String, String, Void> {
        protected Void doInBackground(String... words) {
            for (String word : words) {
                publishProgress(word);
                SystemClock.sleep(1000);
            }
            return (null);
        }

        protected void onPostExecute(Void unused) {
            Toast.makeText(ThreadExample.this, "Done", Toast.LENGTH_SHORT)
                        .show();
        }
```

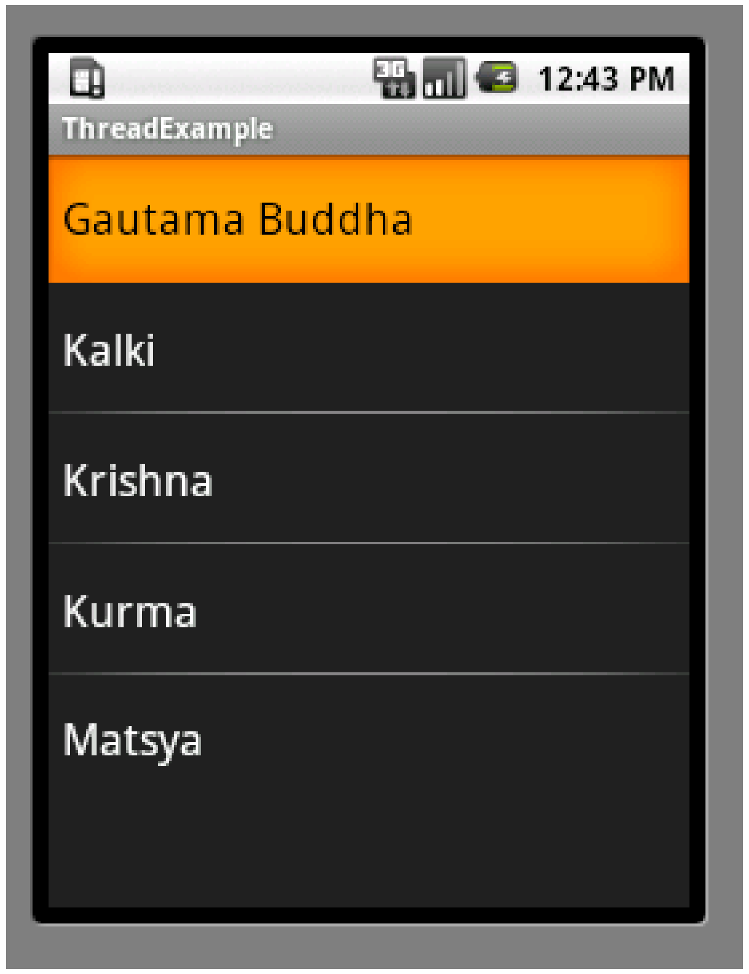
ThreadExample

```
        protected void onPreExecute() {
            Toast.makeText(ThreadExample.this, "Start", Toast.LENGTH_SHORT)
                    .show();
        }

        protected void onProgressUpdate(String... word) {
            Toast.makeText(ThreadExample.this, word[0], Toast.LENGTH_SHORT)
                    .show();
        }
    }

    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
    }

    public void onStart() {
        super.onStart();
        String[] text = { "Bat", "cat", "dat", "fat", "hat", "mat" };
        new SampleTask().execute(text);
    }
}
```

Just For Fun

Dynamically update
List View


12:42 PM
ThreadExample
Gautama Buddha
Kalki


12:43 PM
ThreadExample
Gautama Buddha
Kalki
Krishna
Kurma
Matsya

The Task

```
public class ThreadExample extends ListActivity {
        private class SampleTask extends AsyncTask<Void, String, Void> {
                String[] items = { "Gautama Buddha", "Kalki", "Krishna", "Kurma",
                                "Matsya", "Narasimha", "Parashurama", "Rama", "Vamana",
                                "Varaha" };

                protected Void doInBackground(Void... notused) {
                        for (String word : items) {
                                publishProgress(word);
                                SystemClock.sleep(2500);
                        }
                        return (null);
                }

                protected void onPostExecute(Void unused) {
                        Toast.makeText(ThreadExample.this, "Done", Toast.LENGTH_SHORT)
                                        .show();
                }

                protected void onProgressUpdate(String... word) {
                        listAdapter.add(word[0]);
                }
        }
```

The Activity

```
        private ArrayAdapter<String> listAdapter;

        public void onCreate(Bundle savedInstanceState) {
                super.onCreate(savedInstanceState);
                setContentView(R.layout.main);
                listAdapter = new ArrayAdapter<String>(this,
                                android.R.layout.simple_list_item_1, new ArrayList<String>());
                setListAdapter(listAdapter);
                new SampleTask().execute();
        }
}
```