ANDROID DEVELOPMENT

THREADS AND SERVICES

Tan Cher Wah (isstcw@nus.edu.sg)

Agenda

- Threads
- AsyncTask

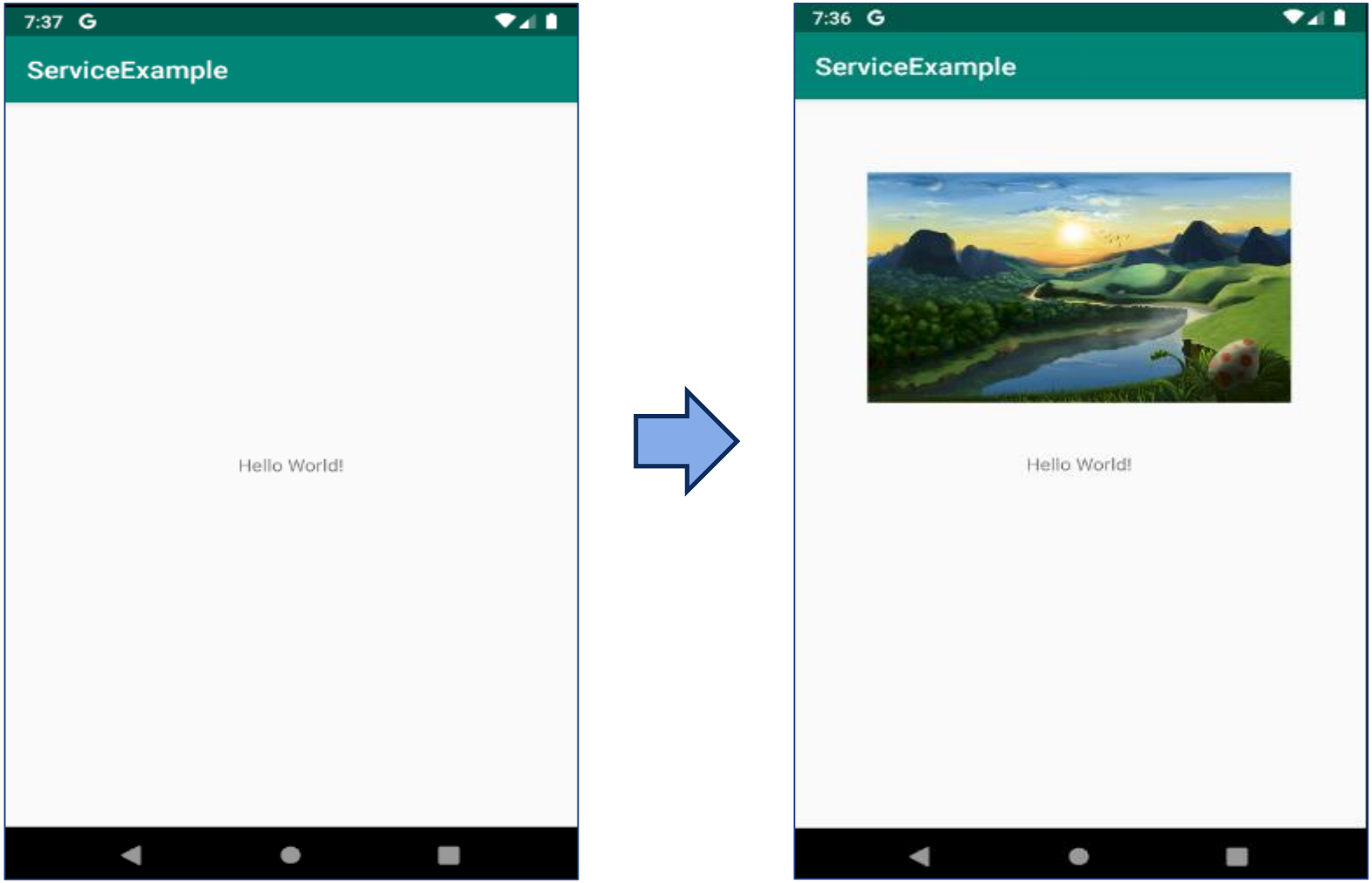
Do not hog the main UI thread

- Android does not allow network operations in its main UI thread
- We can bypass with strict mode off (**don't do this…**)

```
@Override
public void onClick(View v) {
    // use 10.0.2.2 for emulator to access machine's local web-server
    Bitmap img = loadImageFromNetwork("http://10.0.2.2/sample.png");

    ImageView imgView = findViewById(R.id.imgView1);
    imgView.setImageBitmap(img);
}

private Bitmap loadImageFromNetwork(String src) {
    StrictMode.ThreadPolicy policy = new StrictMode.ThreadPolicy.Builder().permitAll().build();
    StrictMode.setThreadPolicy(policy);

    try {
        URL url = new URL(src);
        HttpURLConnection connection = (HttpURLConnection) url.openConnection();
        connection.setDoInput(true);
        connection.connect();
        InputStream input = connection.getInputStream();
        return BitmapFactory.decodeStream(input);
    } catch (IOException e) {
        Resources res = getApplicationContext().getResources();
        Bitmap icon = BitmapFactory.decodeResource(res, R.mipmap.ic_launcher);
        return icon;
    }
}
```

After retrieving the image…

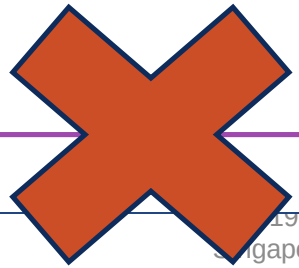
- The result from executing onClick()



Threading in an Activity

- Without the strict mode off, your newly created thread **cannot access UI elements** in the main UI Thread
- The following code **DOES NOT WORK…**

```
@Override
protected void onCreate(Bundle savedInstanceState)
{
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    TextView textView = findViewById(R.id.textView1);
    textView.setOnClickListener(this);
}

@Override
public void onClick(View v)
{
    new Thread(new Runnable() {
        public void run() {
            // use 10.0.2.2 for emulator to access machine's local web-server
            Bitmap img = loadImageFromNetwork("http://10.0.2.2/sample.png");

            ImageView imgView = findViewById(R.id.imgView1);
            imgView.setImageBitmap(img);
        }
    }).start();
}
```



AsyncTask

- To support Asynchronous operations
  - Download medias over the network
  - Perform lengthy computations

- Key abstractions
  - Utilize a **background thread** for async operations
  - Access the **main UI thread** during **progress updates** and when task is **completed**

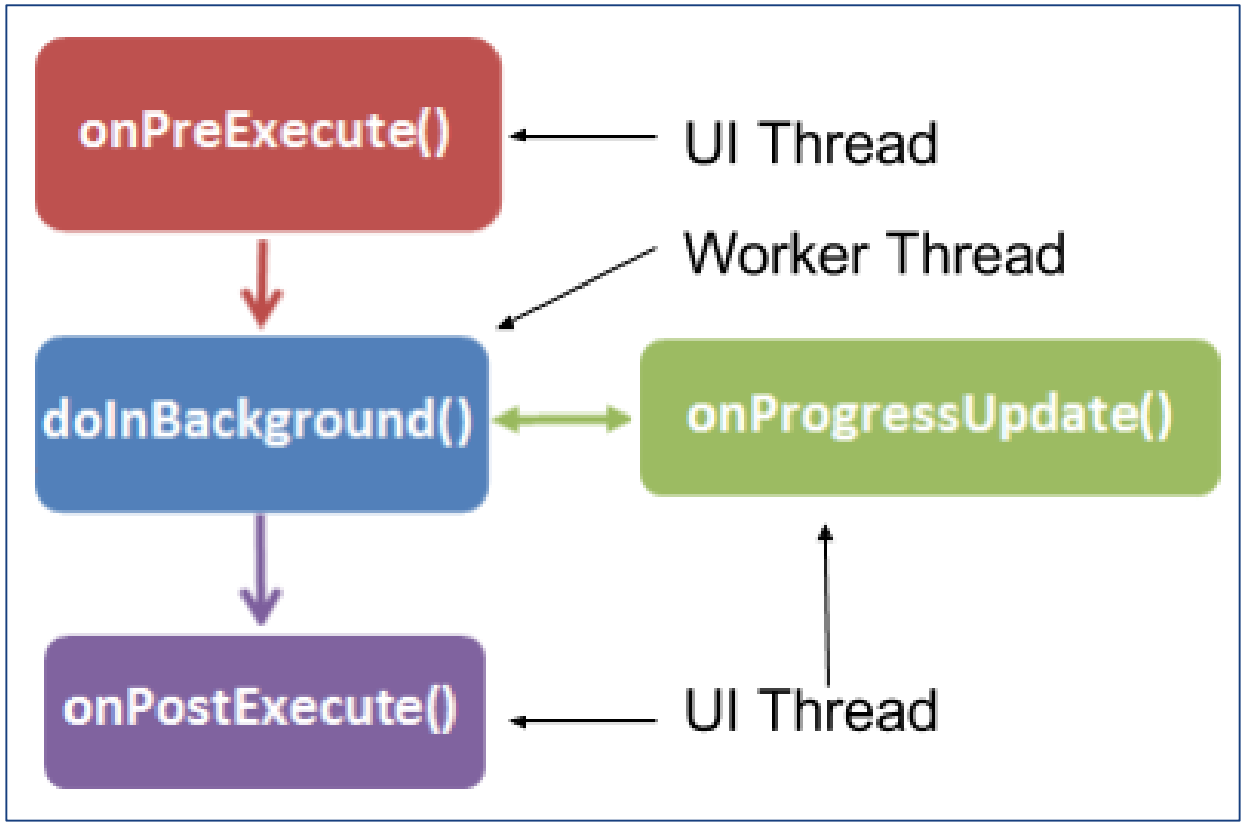
Threading with AsyncTask

- Override these 4 methods as needed

onPreExecute	Used for setting up your task (runs on UI thread)
doInBackground	Implement code here to execute the task (runs on Worker thread)
onProgressUpdate	Used for updating progress in UI (runs on UI thread)
onPostExecute	Used for updating results in UI once operation completes (runs on UI thread)

Threading in an Activity

- UI vs Worker thread in AsyncTask
- Only code in UI Thread can access UI elements



AsyncTask

- Invoking our asynchronous task in onClick()

```
public class MainActivity extends AppCompatActivity
        implements View.OnClickListener
{
    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        TextView textView = findViewById(R.id.textView1);
        textView.setOnClickListener(this);
    }

    @Override
    public void onClick(View v)
    {
        new MyDownloader(this).execute("http://10.0.2.2/sample.png");
    }
}
```

AsyncTask

- Implementing our Downloader as a subclass of AsyncTask

```
public class MyDownloader extends AsyncTask<String, Void, Bitmap>
{
   private WeakReference<MainActivity> parent = null;

   public MyDownloader(MainActivity parent)
   {
      this.parent = new WeakReference<>(parent);
   }

   @Override
   protected Bitmap doInBackground(String... urls)
   {
      try {
         URL url = new URL(urls[0]);
         HttpURLConnection connection = (HttpURLConnection) url.openConnection();
         connection.setDoInput(true);
         connection.connect();
         InputStream input = connection.getInputStream();
         return BitmapFactory.decodeStream(input);
      } catch (IOException e) {
         return null;
      }
   }

   protected void onPostExecute(Bitmap bitmap)
   {
      if (this.parent != null && bitmap != null) {
         MainActivity parent = this.parent.get();
         ImageView imgView = parent.findViewById(R.id.imgView1);
         imgView.setImageBitmap(bitmap);
      }
   }
}
```

AsyncTask's arguments

- The **parameter types**, for your AsyncTask class, are defined by you

Argument 1	Used as inputs to doInBackground()
Argument 2	Used as progress units published in doInBackground
Argument 3	Uses as returned value from doInBackground()

Reference

- Creating and Executing Async Tasks - https://guides.codepath.com/android/Creating-and-Executing-Async-Tasks

- Sending and Managing Network Requests - https://guides.codepath.com/android/Sending-and-Managing-Network-Requests#displaying-remote-images-the-hard-way

Active Recall