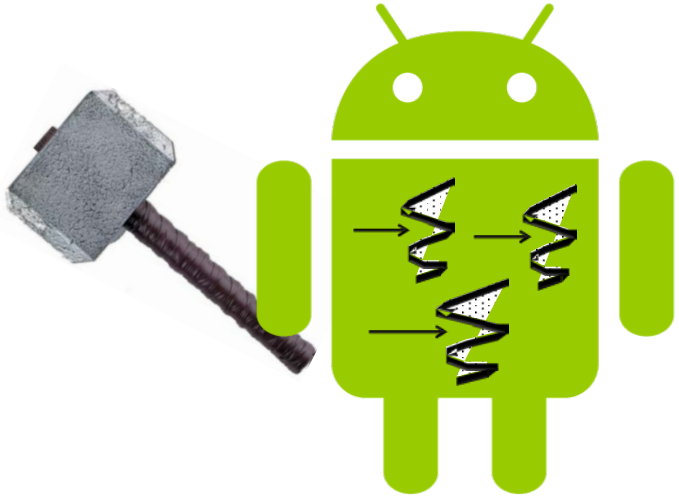
Android Concurrency: Programming with Android Concurrency Frameworks

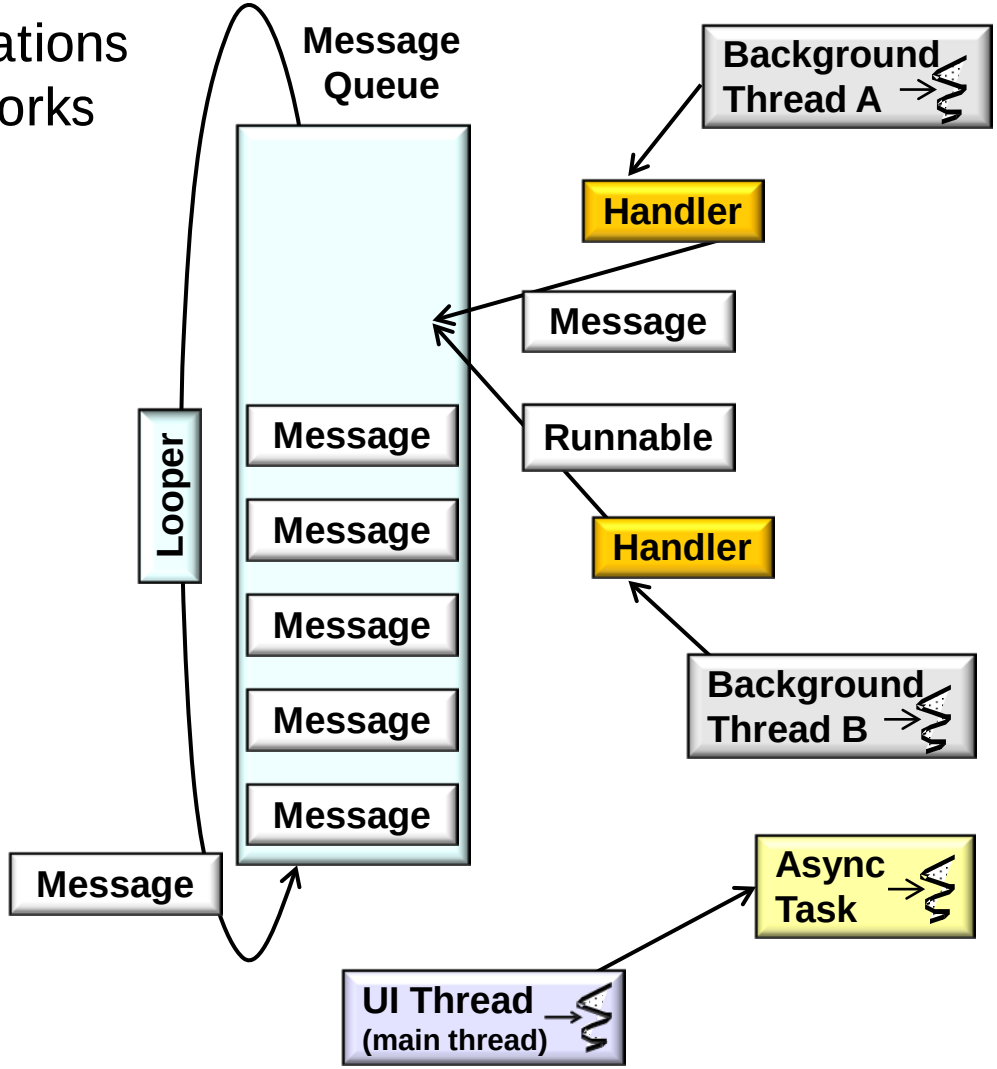
Douglas C. Schmidt
d.schmidt@vanderbilt.edu
www.dre.vanderbilt.edu/~schmidt

Institute for Software Integrated Systems
Vanderbilt University
Nashville, Tennessee, USA

Learning Objectives in this Part of the Module

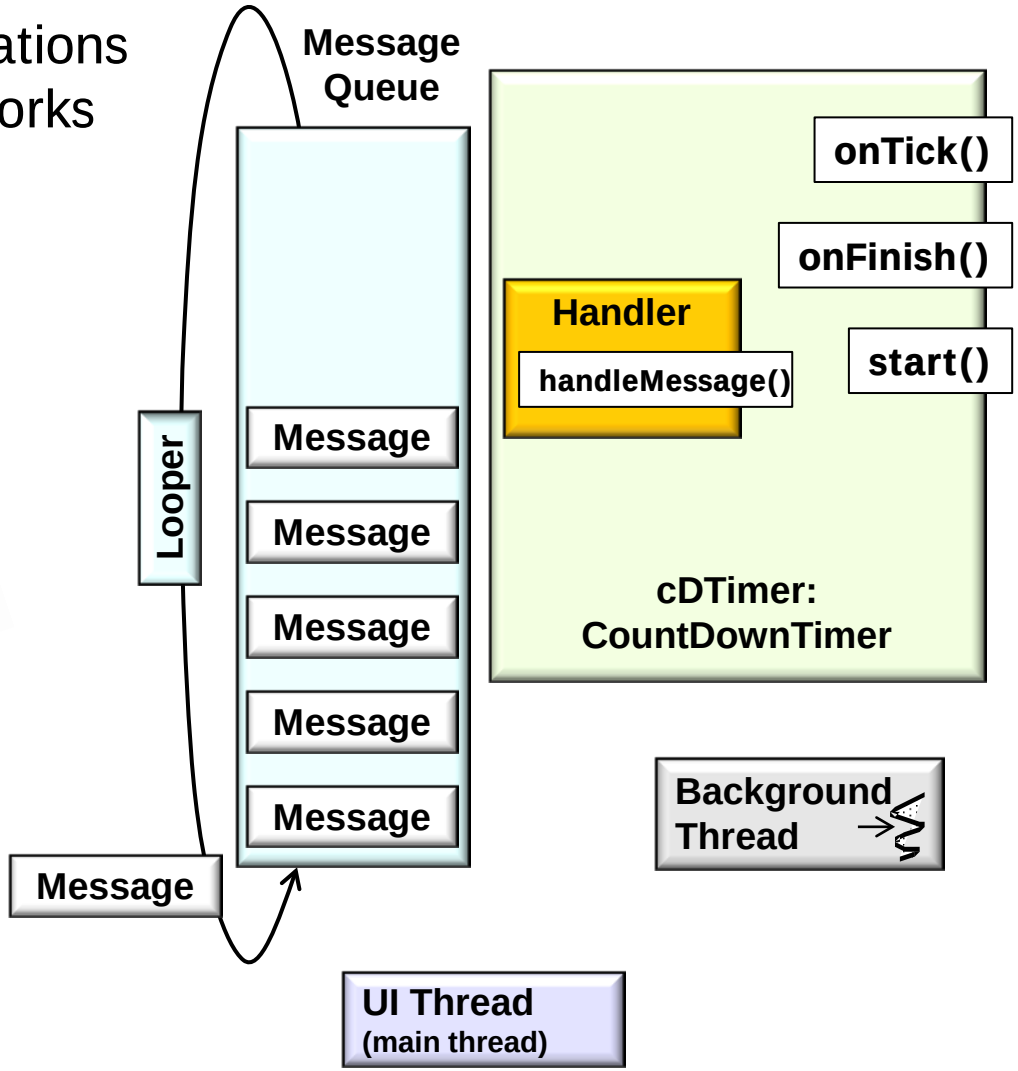
- Understand how to program applications with Android's concurrency frameworks





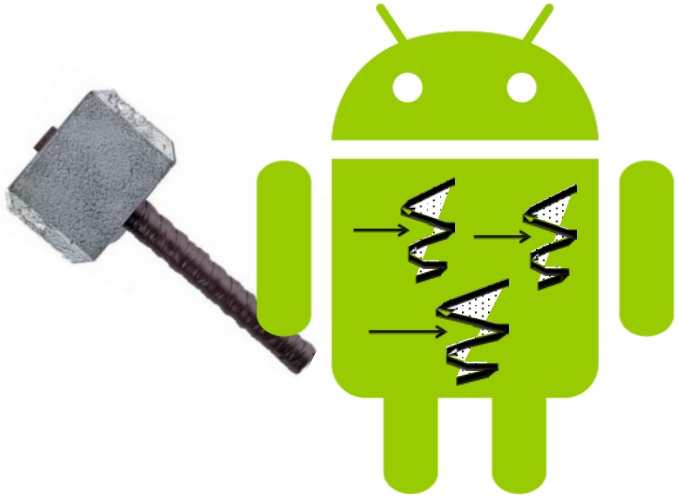
Learning Objectives in this Part of the Module

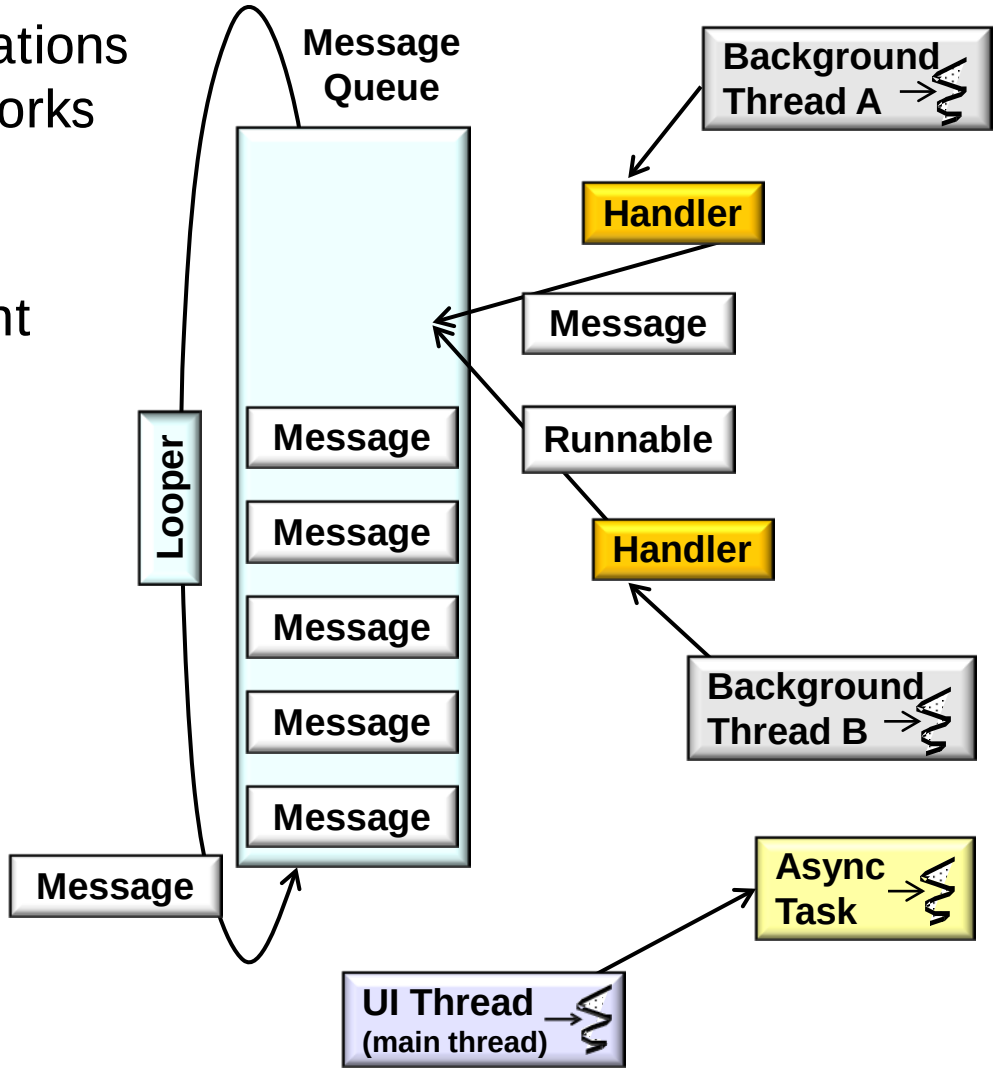
- Understand how to program applications with Android's concurrency frameworks



Learning Objectives in this Part of the Module

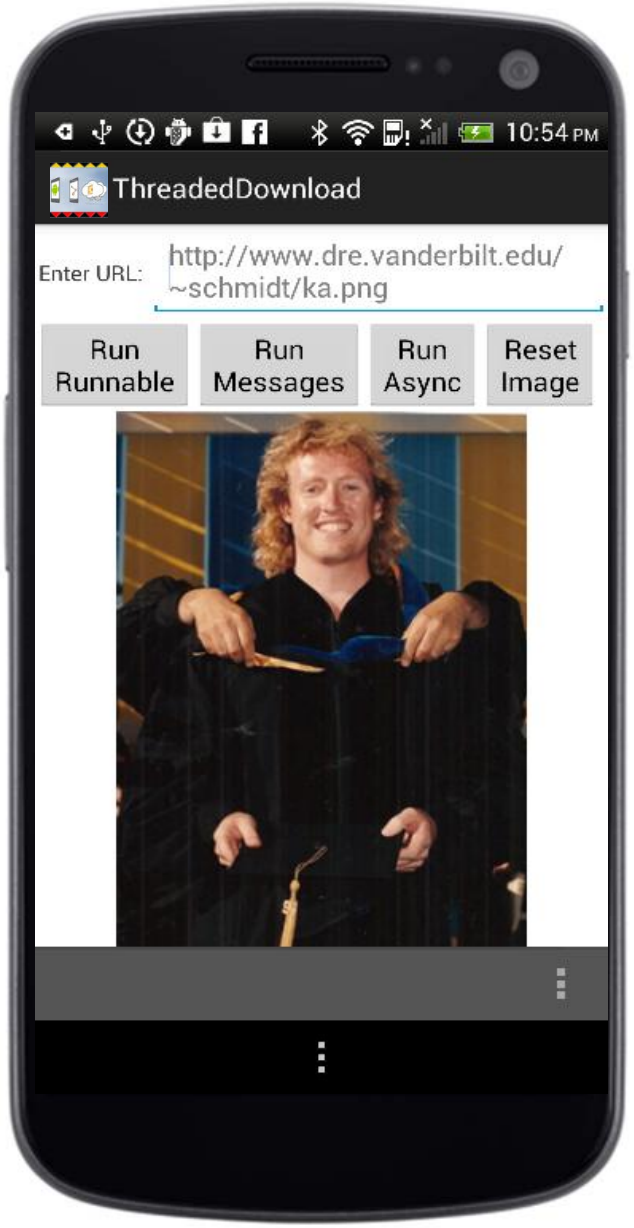
- Understand how to program applications with Android's concurrency frameworks
  - Experiment with the HaMeR & AsyncTask frameworks in *your* Android development environment





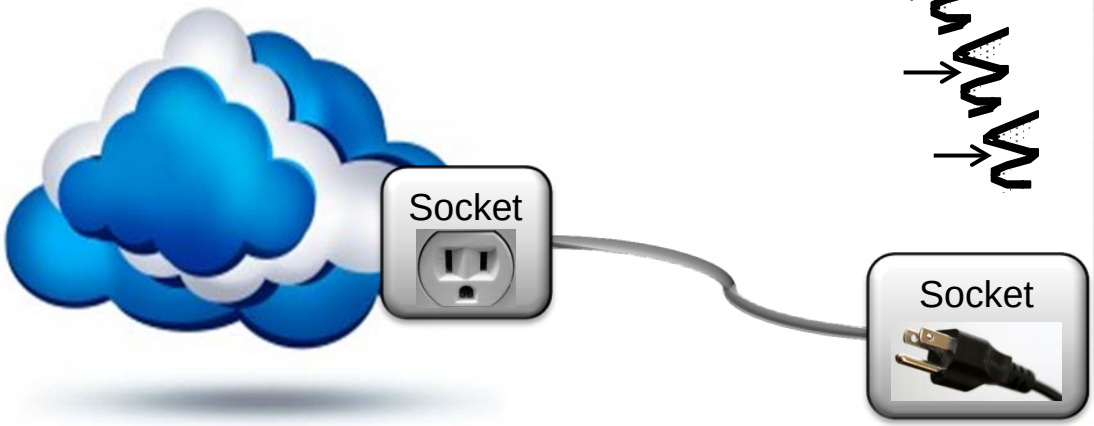
Learning Objectives in this Part of the Module

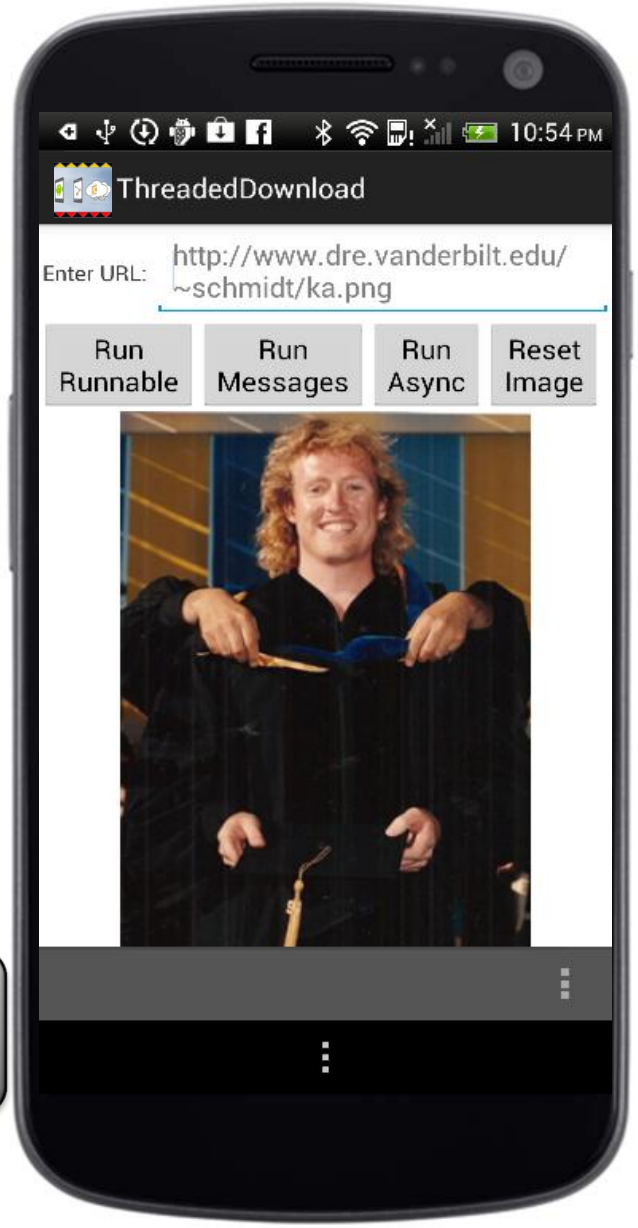
- Understand how to program applications with Android's concurrency frameworks
- Recognize the structure & functionality of the ThreadedDownloads application



Learning Objectives in this Part of the Module

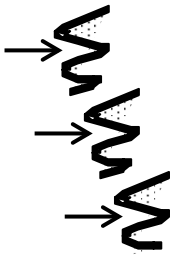
- Understand how to program applications with Android's concurrency frameworks
- Recognize the structure & functionality of the ThreadedDownloads application

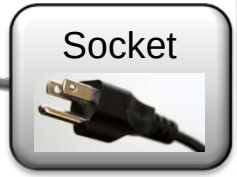


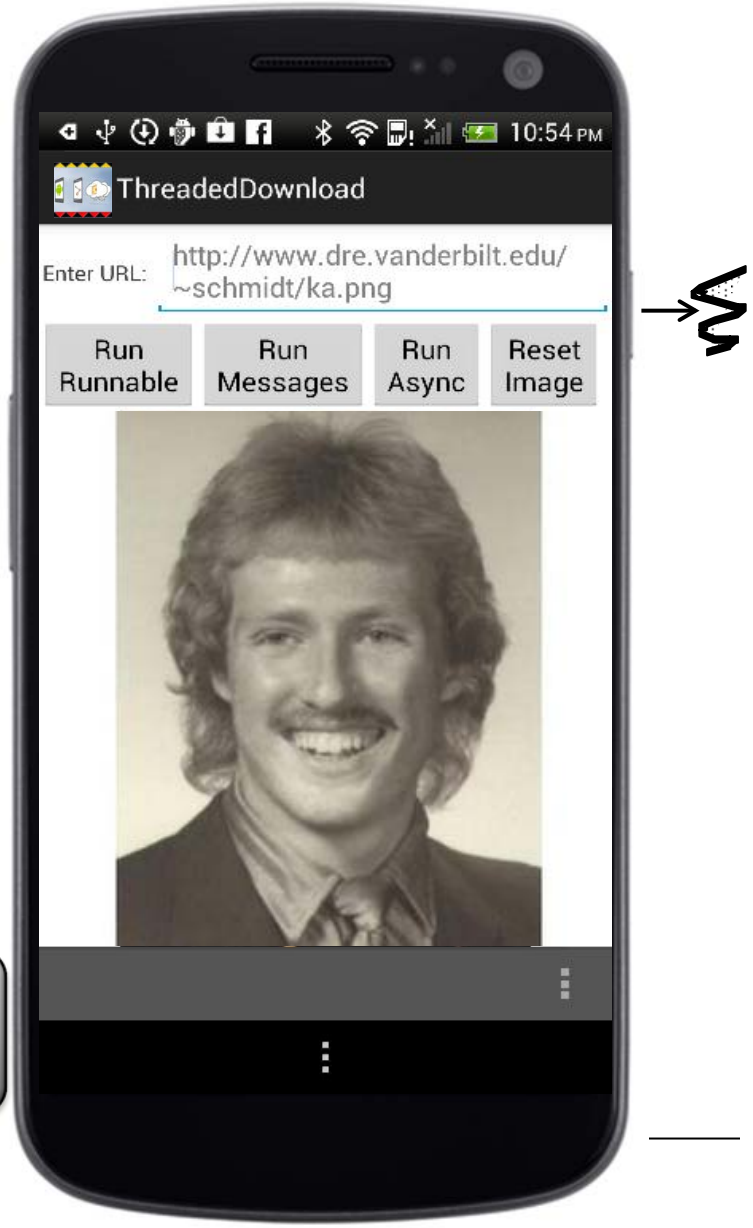


Learning Objectives in this Part of the Module

- Understand how to program applications with Android's concurrency frameworks
- Recognize the structure & functionality of the ThreadedDownloads application

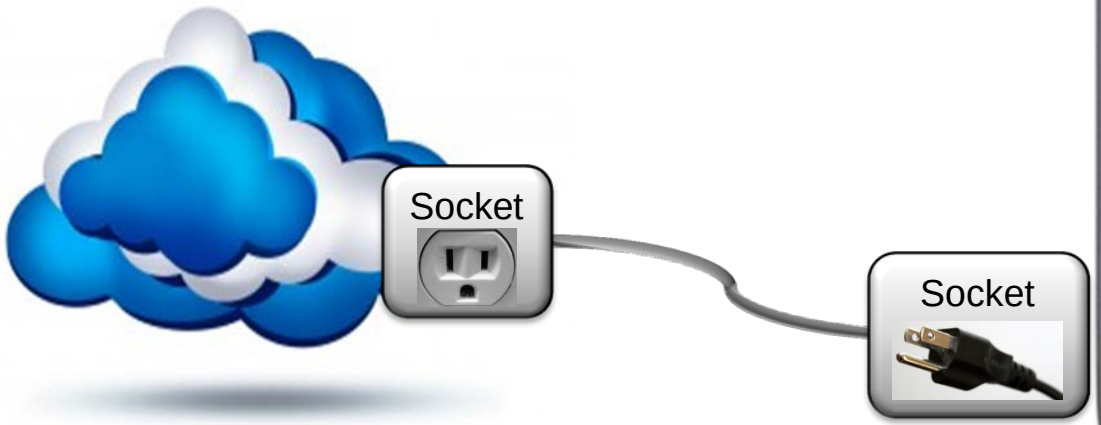


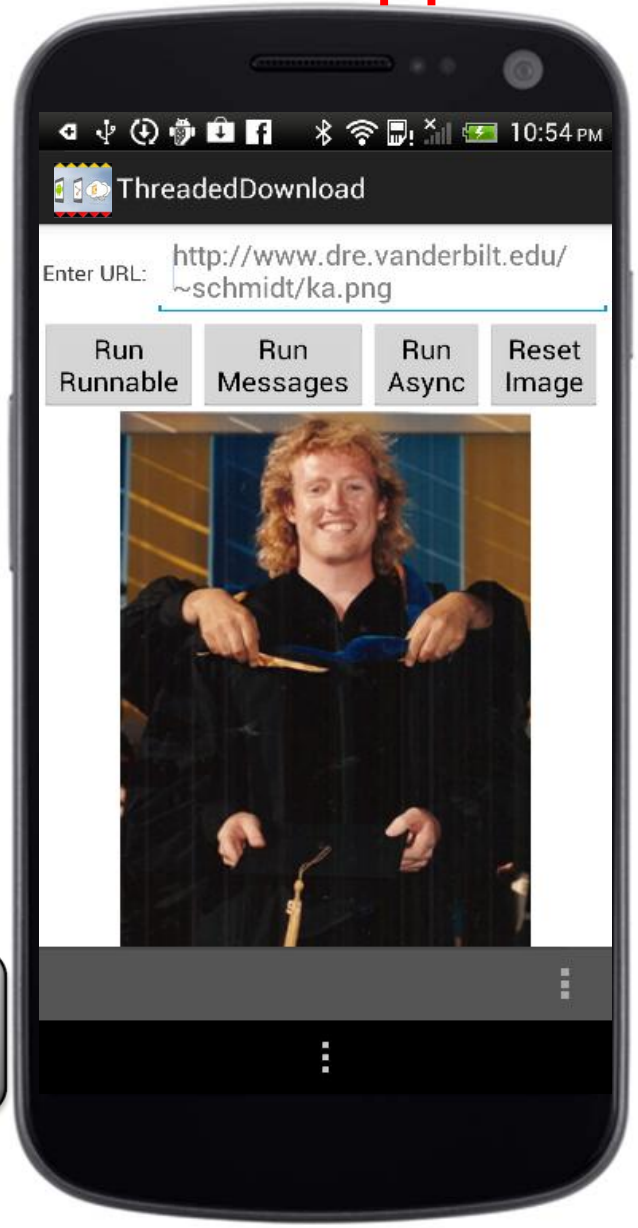




Overview of the Threaded Downloads Application

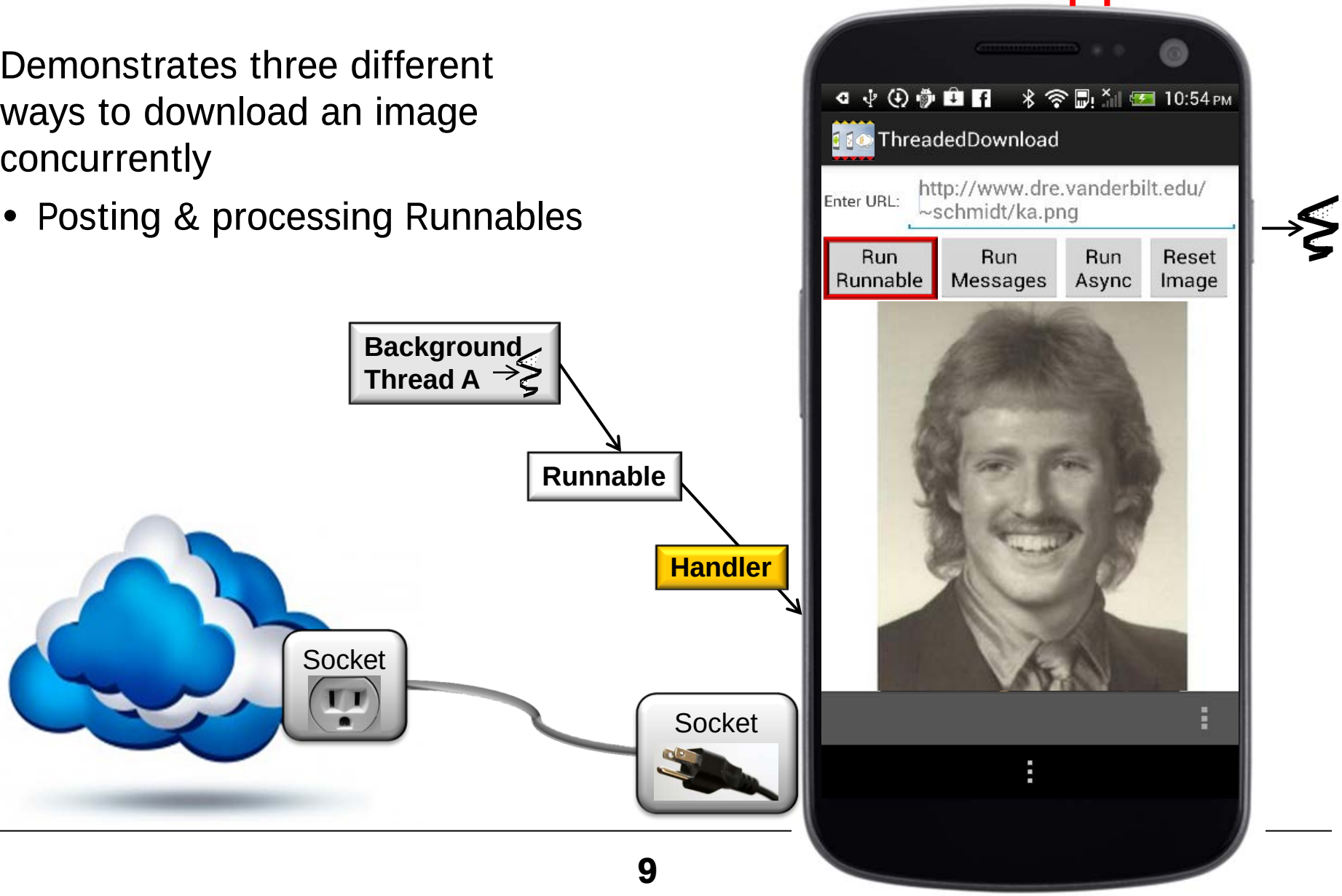
- Demonstrates three different ways to download an image concurrently





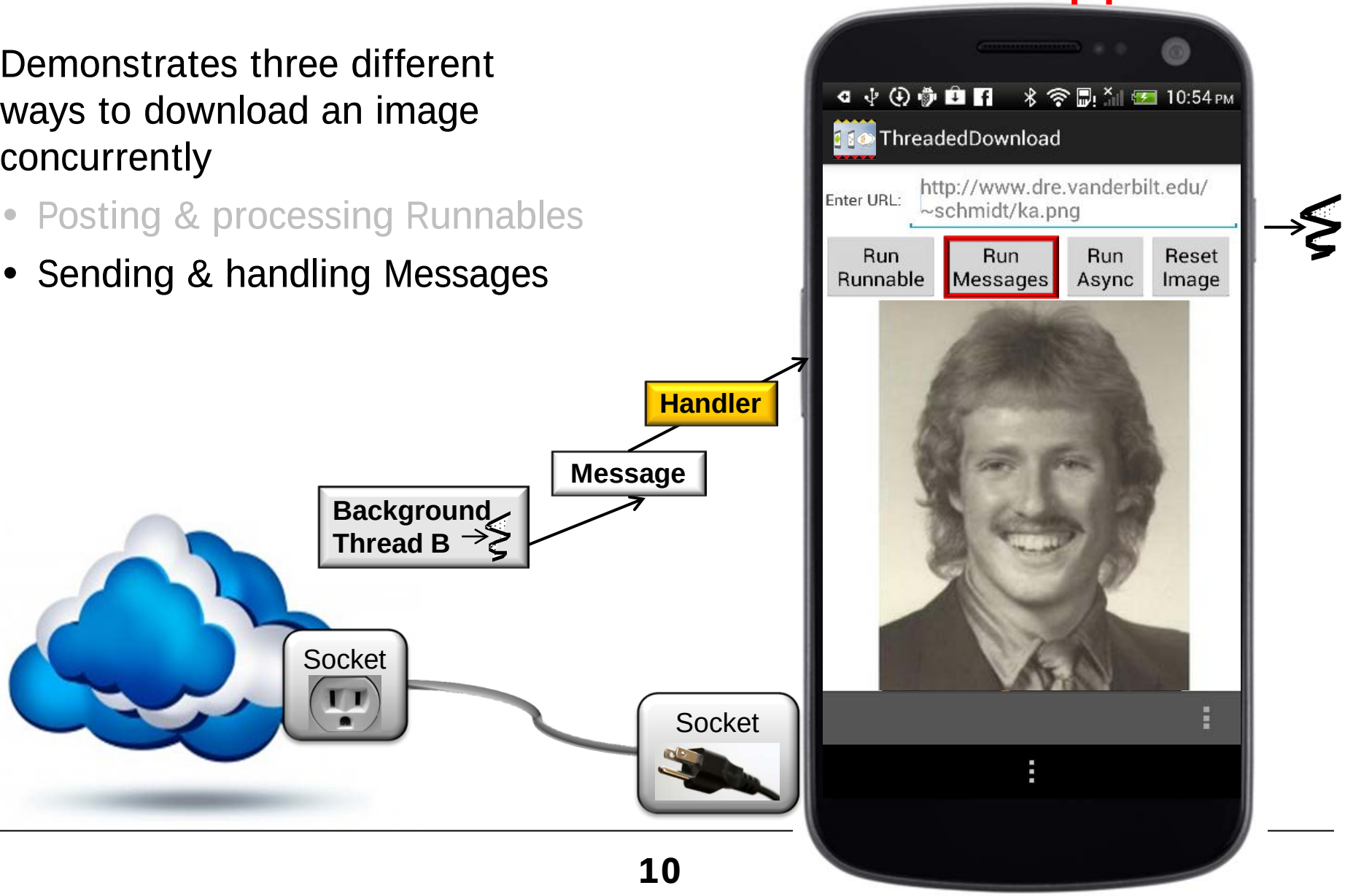
Overview of the Threaded Downloads Application

- Demonstrates three different ways to download an image concurrently
  - Posting & processing Runnables



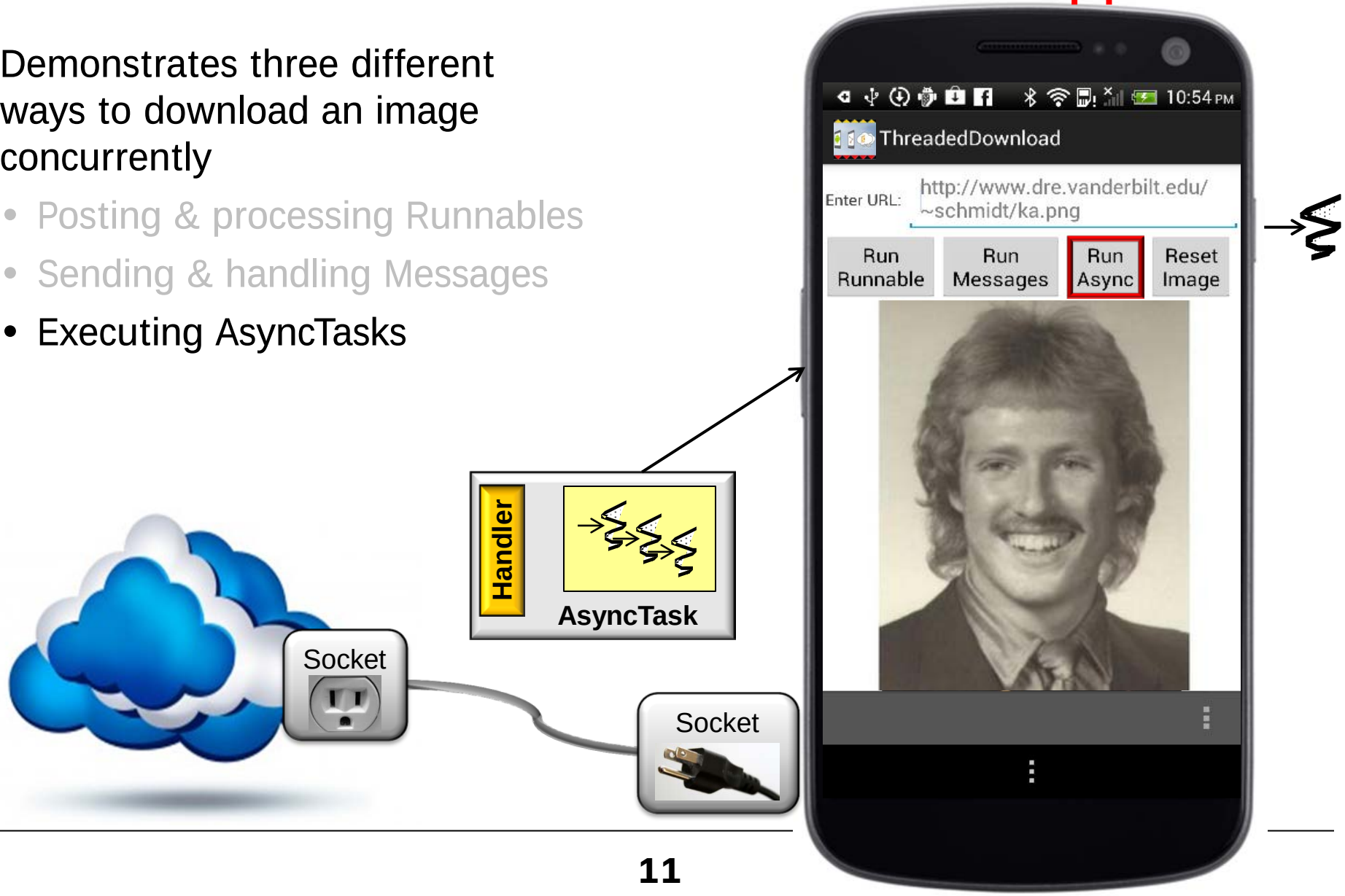
Overview of the Threaded Downloads Application

- Demonstrates three different ways to download an image concurrently
  - Posting & processing Runnables
  - Sending & handling Messages



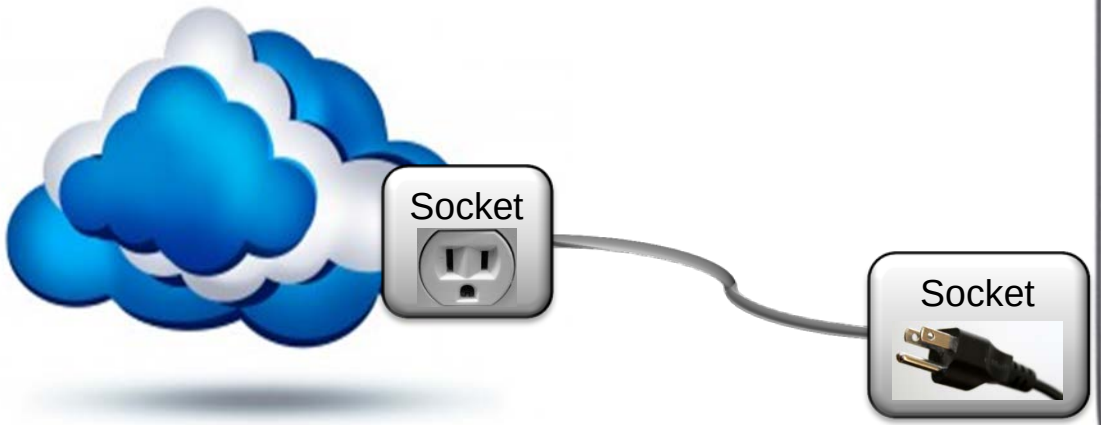
Overview of the Threaded Downloads Application

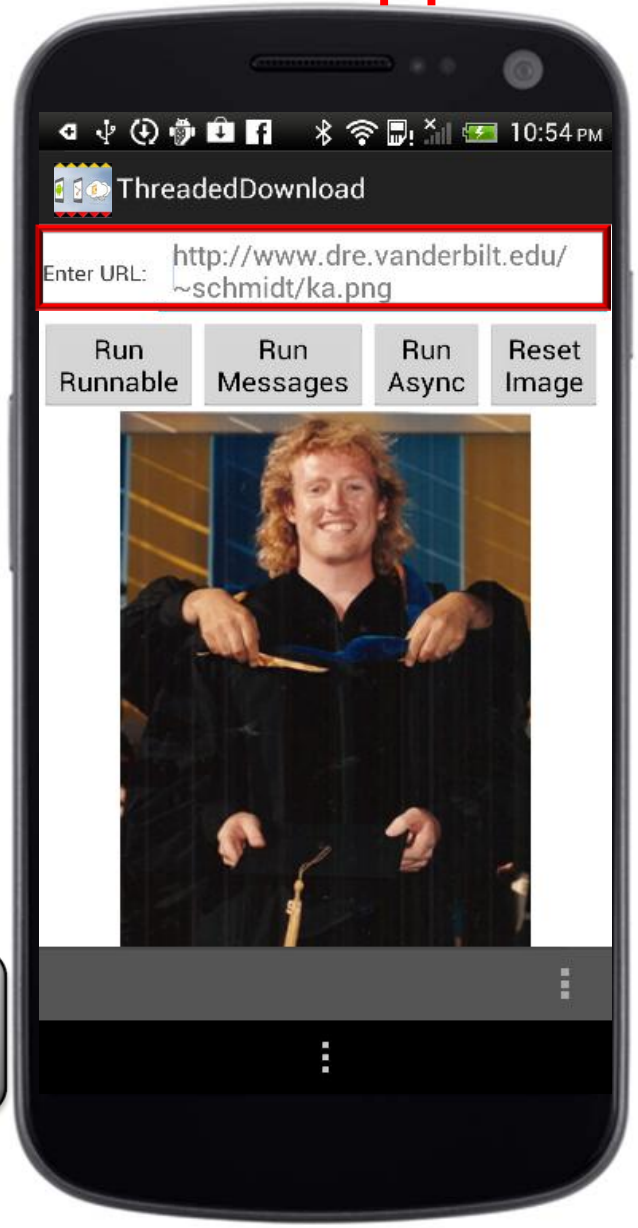
- Demonstrates three different ways to download an image concurrently
  - Posting & processing Runnables
  - Sending & handling Messages
  - Executing AsyncTasks



Overview of the Threaded Downloads Application

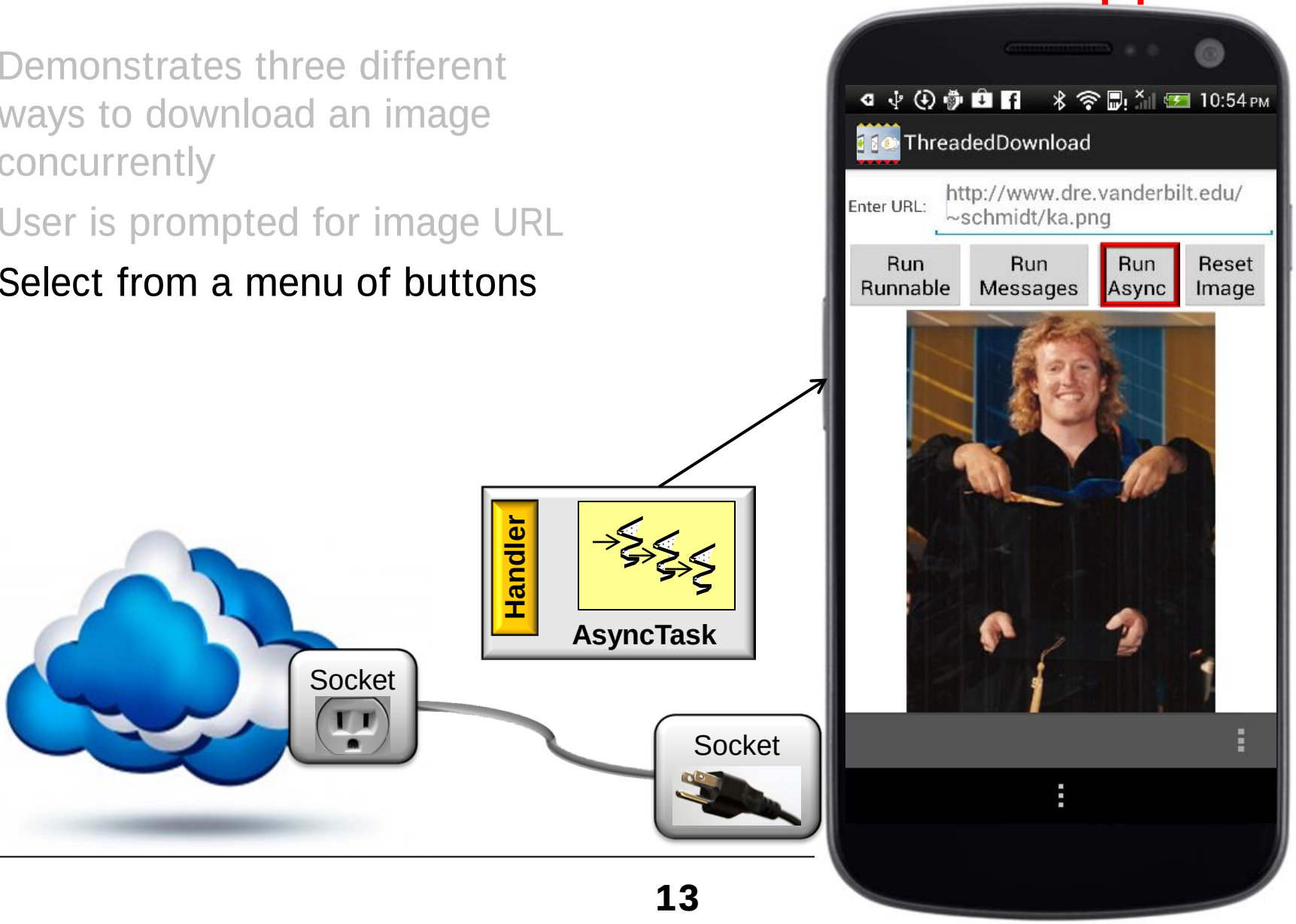
- Demonstrates three different ways to download an image concurrently
- User is prompted for image URL





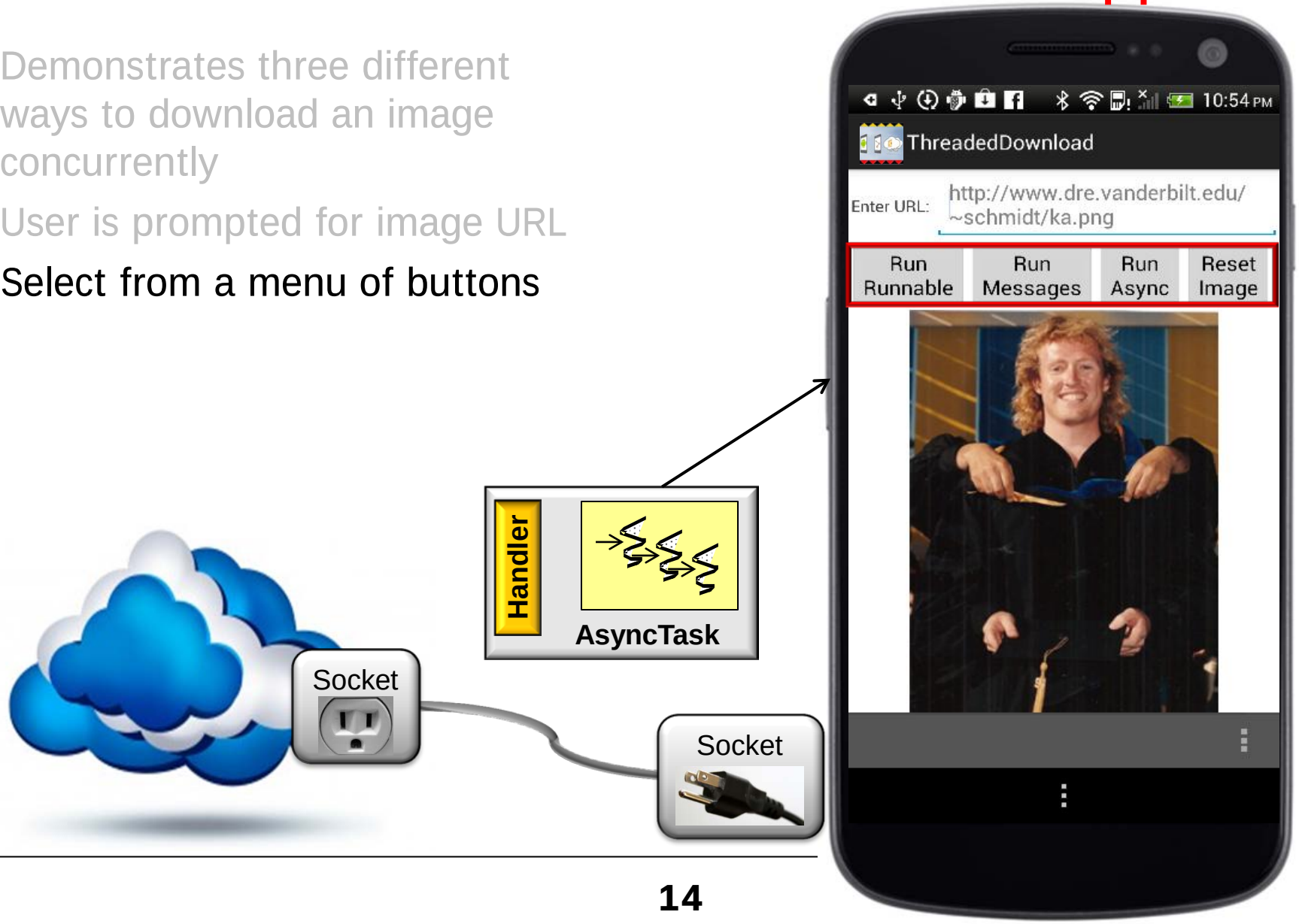
Overview of the Threaded Downloads Application

- Demonstrates three different ways to download an image concurrently
- User is prompted for image URL
- Select from a menu of buttons



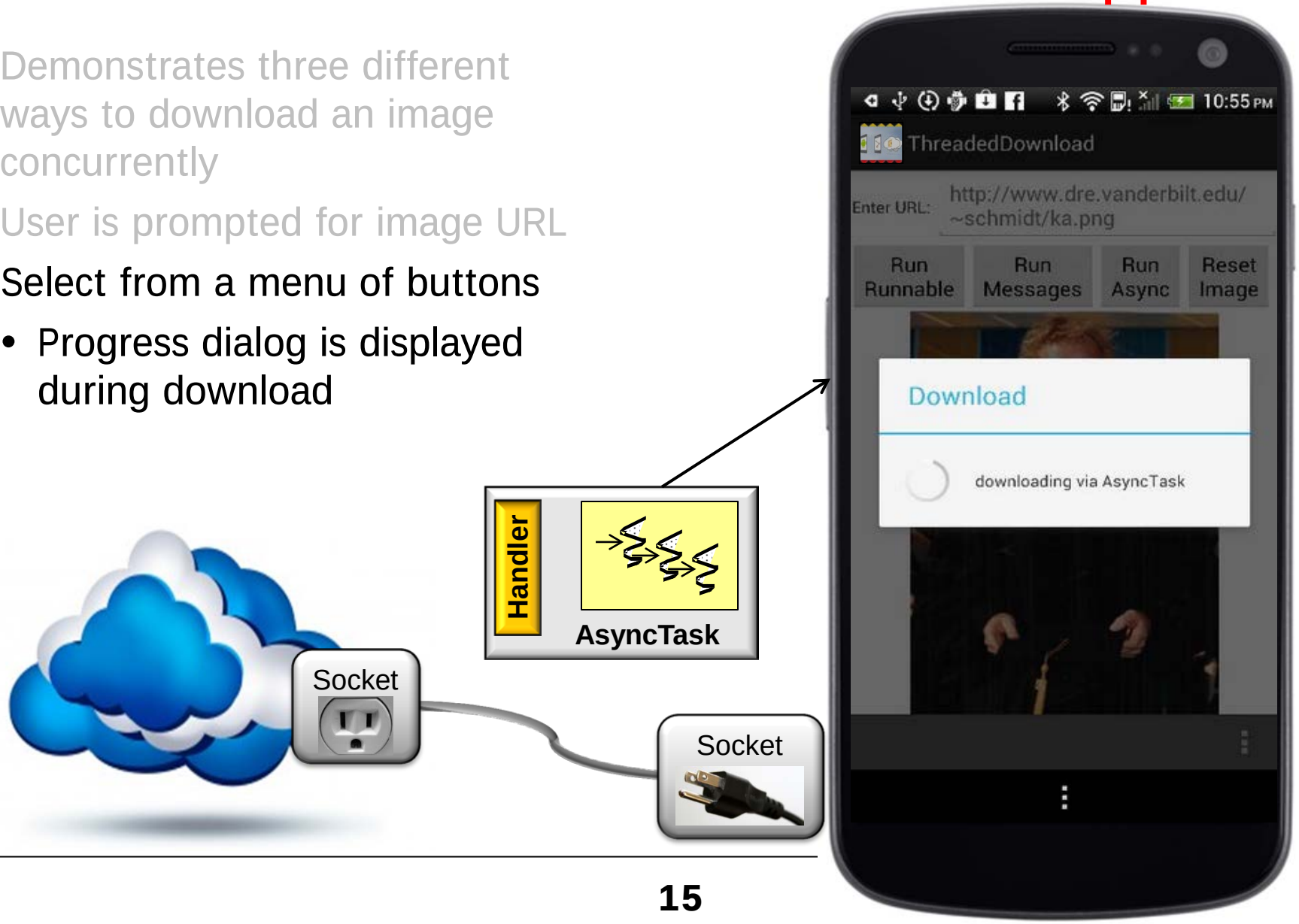
Overview of the Threaded Downloads Application

- Demonstrates three different ways to download an image concurrently
- User is prompted for image URL
- Select from a menu of buttons



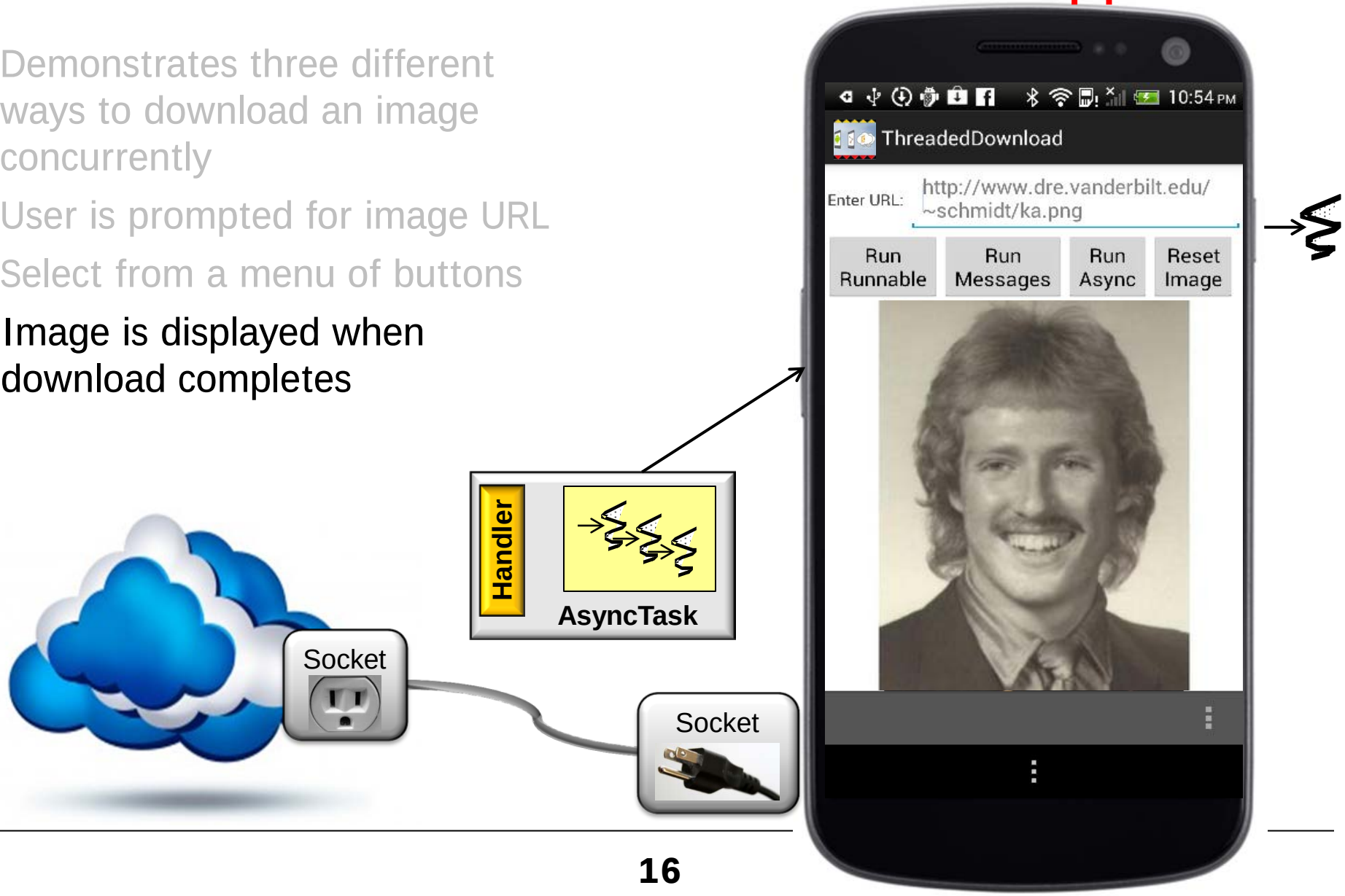
Overview of the Threaded Downloads Application

- Demonstrates three different ways to download an image concurrently
- User is prompted for image URL
- Select from a menu of buttons
  - Progress dialog is displayed during download



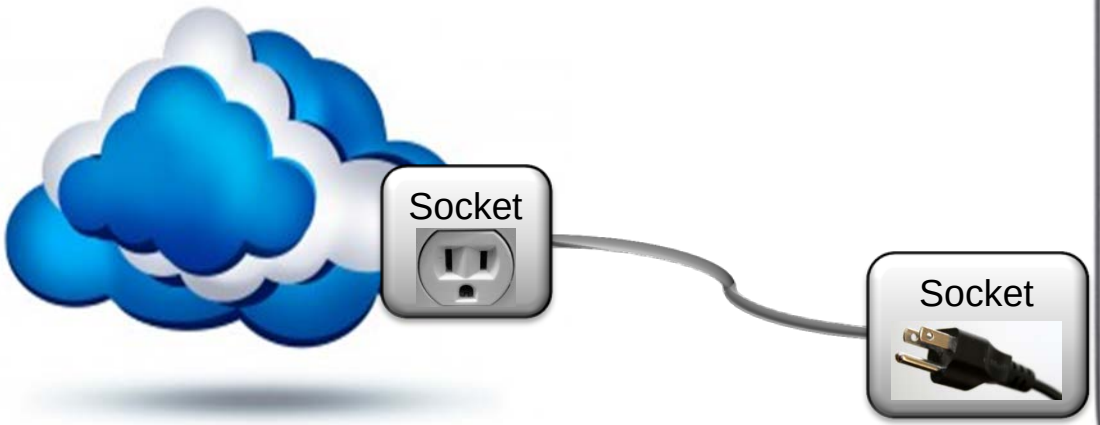
Overview of the Threaded Downloads Application

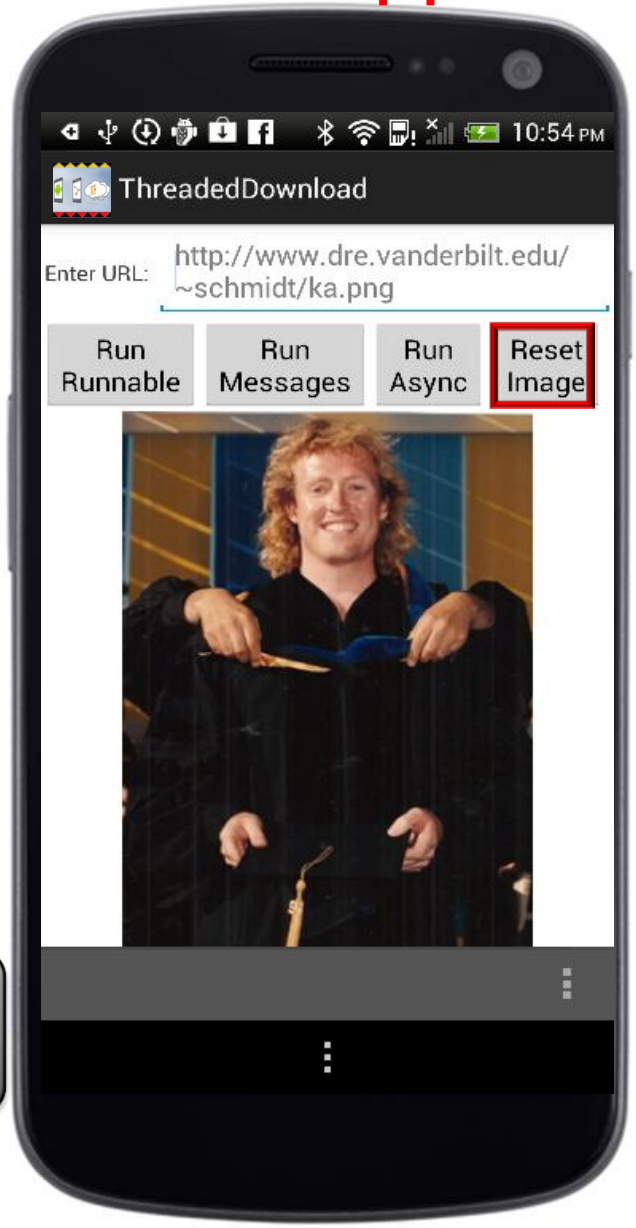
- Demonstrates three different ways to download an image concurrently
- User is prompted for image URL
- Select from a menu of buttons
- Image is displayed when download completes



Overview of the Threaded Downloads Application

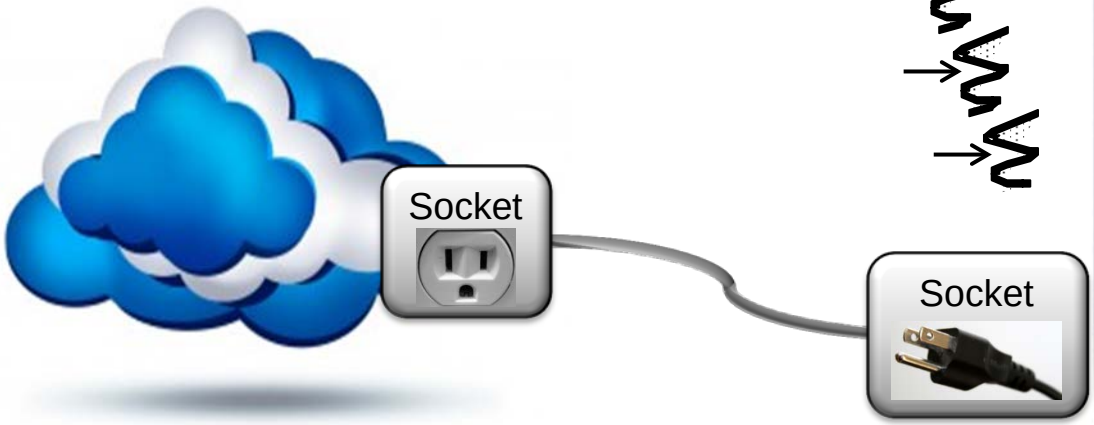
- Demonstrates three different ways to download an image concurrently
- User is prompted for image URL
- Select from a menu of buttons
- Image is displayed when download completes
- Default image can be reset

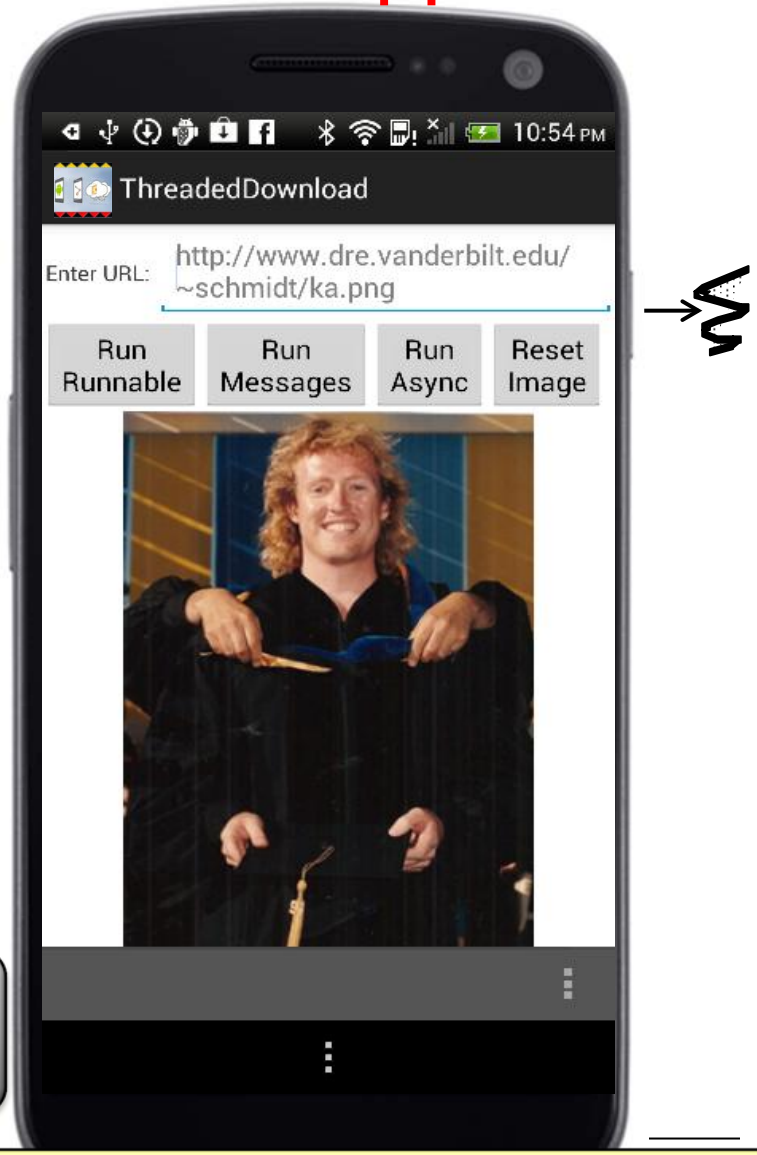




Overview of the Threaded Downloads Application

```
public class ThreadedDownloads
             extends Activity {
  ...
  Bitmap downloadBitmap (String url)
    {...}
  public void runRunnable(View view)
    {...}
  public void runMessages(View view)
    {...}
  public void runAsyncTask(View view)
    {...}
```





See github.com/douglascraigschmidt/POSA-14/tree/master/ex/ThreadedDownload

The Structure & Functionality of the Threaded Downloads Project

Threaded Downloads Structure & Functionality

- Analyze structure & functionality of Threaded Downloads Project

Java - ThreadedDownloads/src/edu/vuum/mocca/ThreadedDownloads.java - ADT

File Edit Refactor Source Navigate Search Project Run Window Help

Packag... Navigat...

- AsyncTaskExample [mooc-ap
- AsyncTaskExampleWithPopu
- BuggyQueue
- CommandPatternExample [m
- HandlerExample [mooc-app
- iRemember [mooc-app mast
- iRememberContentProvider
- PingPongActivity
- PingPongConsole
- PingPongRight
- ServiceAIDLExample [mooc-
- ServicecAsyncExample [moo
- ThreadedDownloads
  - Android 4.2.2
  - src
    - edu.vuum.mocca
      - ThreadedDownload
  - gen [Generated Java Files]
  - assets
  - bin
  - libs
  - res
  - AndroidManifest.xml
  - ic_launcher-web.png

main.xml | ThreadedDownloads Manifest | ThreadedDownloads.java

```
package edu.vuum.mocca;

import java.io.InputStream;

/**
 * @class ThreadedDownloads
 *
 * @brief A class that allows a user to download a bitmap image using
 *        any of the following concurrency models: Handlers and
 *        Runnables, Handlers and Messages, and AsyncTask.
 *
 */
public class ThreadedDownloads extends Activity
{
    /**
     * The default URL to download.
     */
    private final String mDefaultURL = "http://www.dre.vanderbilt.edu/~schmidt/ka.png";

    /**
     * Contains user's selection of the URL to download.
     */
    private EditText mUrlEditText;

    /**
```

Outline

- edu.vuum.mocca
- ThreadedDownloads
  - mDefaultURL : String
  - mUrlEditText : EditText
  - mImageView : ImageView
  - mProgressDialog : ProgressDialog
  - TAG : String
  - onCreate(Bundle) : void
  - showErrorToast(String) : void
  - downloadImage(String) : Bitmap
    - new Runnable() {...}
  - runRunnable(View) : void
  - RunnableWithHandlers
  - runMessages(View) : void
  - SHOW_DIALOG : int
  - DISMISS_DIALOG : int
  - DISPLAY_IMAGE : int
  - MessageHandler
  - messageHandler : MessageHandle
  - RunnableWithMessages
    - mUrl : String
    - RunnableWithMessages(String)

Problems | Javadoc | Declaration | Console | LogCat

Writable | Smart Insert | 350 : 1 | 227M of 323M

Threaded Downloads Structure & Functionality

- Analyze structure & functionality of Threaded Downloads Project
- Three main elements:

Threaded Downloads Structure & Functionality

- Analyze structure & functionality of Threaded Downloads Project
- Three main elements:
  - Java source code

```
public class ThreadedDownloads
                extends Activity {
  ...
  Bitmap downloadBitmap
    (String url) {...}
  public void runRunnable
    (View view) {...}
  public void runMessages
    (View view) {...}
  public void runAsyncTask
    (View view) {...}
  ...
```

The most creative & “free form” portion of an Android application

Threaded Downloads Structure & Functionality

- Analyze structure & functionality of Threaded Downloads Project
- Three main elements:
  - Java source code
  - Resources that provide additional files & static content used by Java code
    - e.g., bitmaps, UI layouts, internationalized strings, etc.

```
<LinearLayout
  android:layout_width=
    "fill_parent"
  android:layout_height=
    "wrap_content"
  android:orientation=
    "horizontal">

  <Button
    android:id="@+id/button1"
    android:layout_width=
      "wrap_content"
    android:layout_height=
      "wrap_content"
    android:onClick=
      "runRunnable"
    android:text=
      "@string/runRunnable" />
  ...
```

developer.android.com/guide/topics/resources/index.html has more info

Threaded Downloads Structure & Functionality

- Analyze structure & functionality of Threaded Downloads Project
- Three main elements:
  - Java source code
  - Resources that provide additional files & static content used by Java code
  - XML Manifest file containing information Android needs to execute the application

```
<manifest>  ...
  <application>
    <activity>
      <intent-filter>
       <action />  ...  <data />
      </intent-filter>  ...
    </activity>
    <service>
      <intent-filter>  ...
      </intent-filter>
    </service>
    <receiver>
      <intent-filter>  ...
      </intent-filter>
    </receiver>
    <provider>
      <grant-uri-permission />
    </provider>  ...
```

developer.android.com/guide/topics/manifest/manifest-intro.html has more

Threaded Downloads Structure & Functionality

- Analyze structure & functionality of Threaded Downloads Project
- Three main elements
- XML Manifest file for Threaded Downloads application contains essential information

```
<manifest ...
  package="edu.vuum.mocca"
  ...
```

Threaded Downloads Structure & Functionality

- Analyze structure & functionality of Threaded Downloads Project
- Three main elements
- XML Manifest file for Threaded Downloads application contains essential information
  - Grants permission to use the Internet

```
<manifest ...
  package="edu.vuum.mocca"
  <uses-permission
   android:name=
     "android.permission.INTERNET">
  </uses-permission>
  ...
```

Threaded Downloads Structure & Functionality

- Analyze structure & functionality of Threaded Downloads Project
- Three main elements
- XML Manifest file for Threaded Downloads application contains essential information
  - Grants permission to use the Internet
  - Indicates the main Activity & declares with Intents it handles

```
<manifest ...
  package="edu.vuum.mocca"
  <application
    android:icon=
      "@drawable/ic_launcher"
    android:label=
      "@string/app_name"
    ...
    <activity android:name=
      ".ThreadedDownloads"
      <intent-filter>
        <action android:name=
          "android.intent.
           action.MAIN" />
        ...
      </intent-filter>
    </activity>
  </application> ...
```

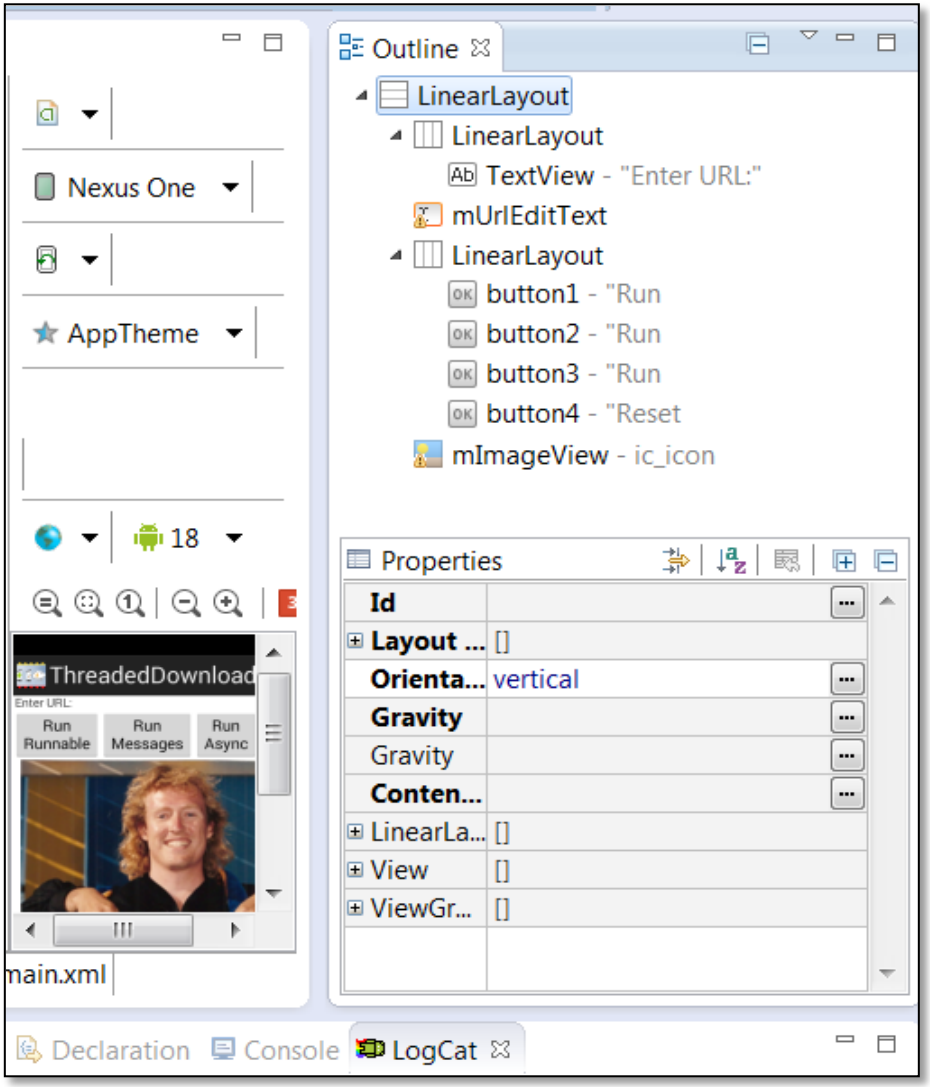
Threaded Downloads Structure & Functionality

- Analyze structure & functionality of Threaded Downloads Project
- Three main elements
- XML Manifest file for Threaded Downloads application contains essential information
  - Grants permission to use the Internet
  - Indicates the main Activity & declares with Intents it handles

```
<manifest ...
  package="edu.vuum.mocca"
  <application
    android:icon=
      "@drawable/ic_launcher"
    android:label=
      "@string/app_name"
    ...
    <activity android:name=
      ".ThreadedDownloads"
      <intent-filter>
        <action android:name=
          "android.intent.
           action.MAIN" />
        ...
      </intent-filter>
    </activity>
  </application> ...
```

Threaded Downloads Structure & Functionality

- Analyze structure & functionality of Threaded Downloads Project
- Three main elements
- XML Manifest file for Threaded Downloads application contains essential information
- The main.xml resource file specifies the application layout



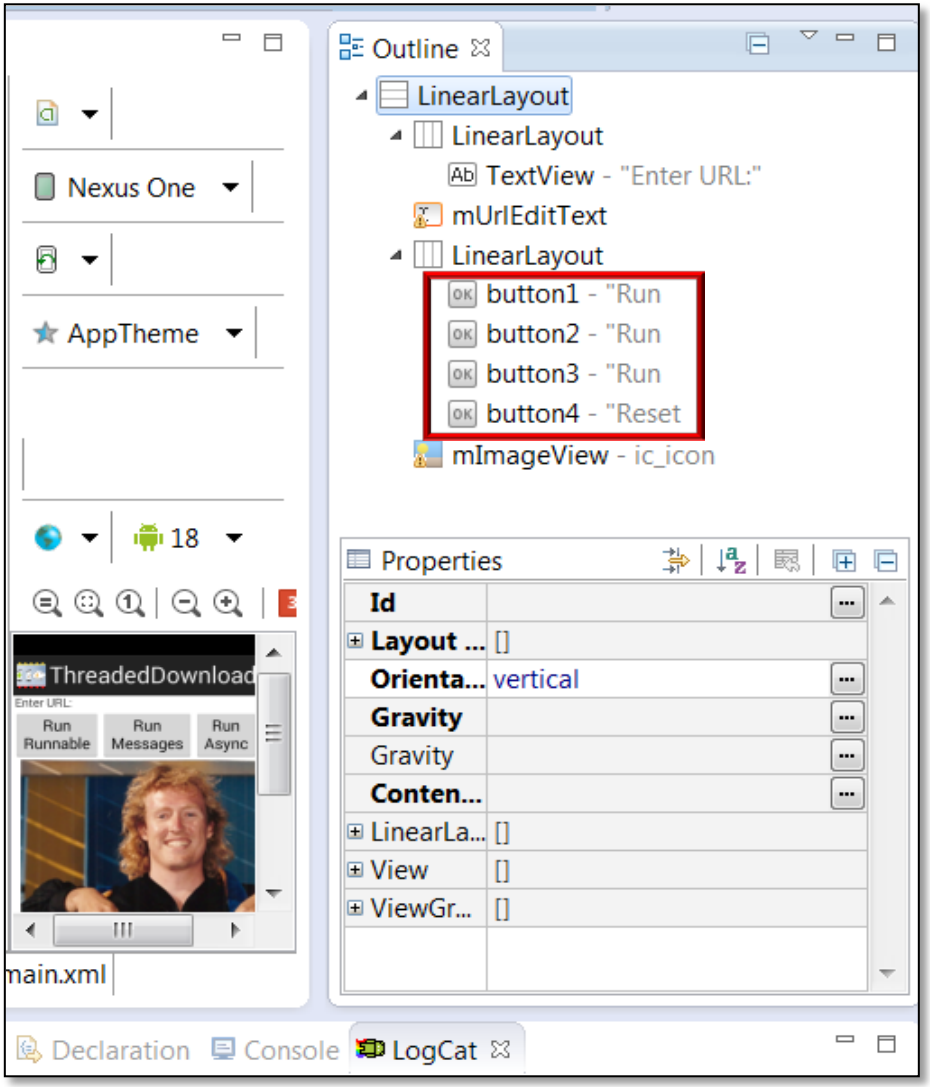
Threaded Downloads Structure & Functionality

- Analyze structure & functionality of Threaded Downloads Project
- Three main elements
- XML Manifest file for Threaded Downloads application contains essential information
- The main.xml resource file specifies the application layout
  - Dictates how text & buttons appears on the screen

```
<TextView
  android:layout_width=
   "wrap_content"
  android:layout_height=
   "wrap_content"
  android:text="@string/location“
   ... />

<EditText
  android:id="@+id/mUrlEditText"
  android:layout_height=
   "wrap_content"
  android:hint="@string/defaultURL“
  .../>

<Button
  android:id="@+id/button1“
  ...
```

Threaded Downloads Structure & Functionality

- Analyze structure & functionality of Threaded Downloads Project
- Three main elements
- XML Manifest file for Threaded Downloads application contains essential information
- The main.xml resource file specifies the application layout
  - Dictates how text & buttons appears on the screen
  - Maps methods to buttons



Threaded Downloads Structure & Functionality

- Analyze structure & functionality of Threaded Downloads Project
- Three main elements
- XML Manifest file for Threaded Downloads application contains essential information
- The main.xml resource file specifies the application layout
  - Dictates how text & buttons appears on the screen
  - Maps methods to buttons

```
<Button
  ...
  android:onClick="runRunnable"
  android:text="@string/runRunnable" />

<Button
  ...
  android:onClick="runMessages"
  android:text="@string/runMessages" />

<Button
  ...
  android:onClick="runAsyncTask"
  android:text="@string/runAsyncTask" />

<Button
  ...
  android:onClick="resetImage"
  android:text="@string/resetImage" />
```

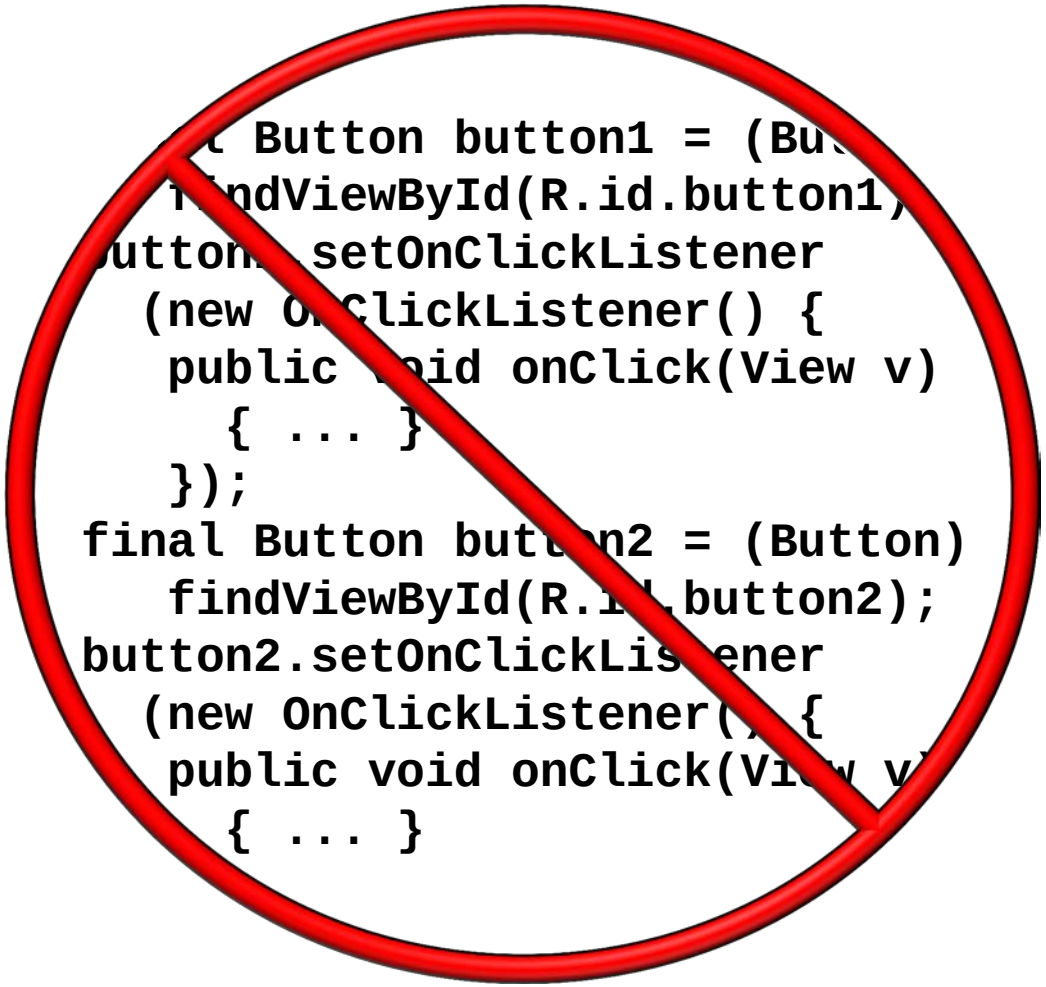
Threaded Downloads Structure & Functionality

- Analyze structure & functionality of Threaded Downloads Project
- Three main elements
- XML Manifest file for Threaded Downloads application contains essential information
- The main.xml resource file specifies the application layout
  - Dictates how text & buttons appears on the screen
  - Maps methods to buttons

```
<Button
  ...
  android:onClick="runRunnable"
  android:text="@string/runRunnable" />

<Button
  ...
  android:onClick="runMessages"
  android:text="@string/runMessages" />

<Button
  ...
  android:onClick="runAsyncTask"
  android:text="@string/runAsyncTask" />

<Button
  ...
  android:onClick="resetImage"
  android:text="@string/resetImage" />
```

Threaded Downloads Structure & Functionality

- Analyze structure & functionality of Threaded Downloads Project
- Three main elements
- XML Manifest file for Threaded Downloads application contains essential information
- The main.xml resource file specifies the application layout
  - Dictates how text & buttons appears on the screen
  - Maps methods to buttons
  - Avoids hard-coding UI components into the ThreadedDownload class



Overview of the ThreadedDownloads Class

Overview of the ThreadedDownloads Class

- ThreadedDownloads is the only Activity defined in Threaded Downloads app

```
<manifest ...
  package="edu.vuum.mocca“
  ...
  <application
    android:icon="@drawable/ic_launcher"
    android:label="@string/app_name"
    ...
    <activity android:name=".ThreadedDownloads"
      <intent-filter>
        <action android:name=
          "android.intent.action.MAIN" />
        ...
      </intent-filter>
    </activity>
  </application> ...
```

ThreadedDownloads/AndroidManifest.html has the XML source code

Overview of the ThreadedDownloads Class

- Implementation of ThreadedDownloads class & data members

```
public class ThreadedDownloads extends Activity {
  ...
  private final String mDefaultURL = ...

  private EditText mUrlEditText;

  private ImageView mImageView;

  private ProgressDialog mProgressDialog;
```

Overview of the ThreadedDownloads Class

- Implementation of ThreadedDownloads class & data members

```
public class ThreadedDownloads extends Activity {
  ...
  private final String mDefaultURL = ...

  private EditText mUrlEditText;

  private ImageView mImageView;

  private ProgressDialog mProgressDialog;
```

Overview of the ThreadedDownloads Class

- Implementation of ThreadedDownloads class & data members

```
public class ThreadedDownloads extends Activity {
  ...

  private final String mDefaultURL = ...

  private EditText mUrlEditText;

  private ImageView mImageView;

  private ProgressDialog mProgressDialog;
```

ThreadedDownloads/src/edu/vuum/mocca/ThreadedDownloads.java has code

Overview of the ThreadedDownloads Class

- Implementation of ThreadedDownloads class & data members

```
public class ThreadedDownloads extends Activity {
  ...
```

Default URL to download

```
  private final String mDefaultURL = ...

  private EditText mUrlEditText;

  private ImageView mImageView;

  private ProgressDialog mProgressDialog;
```

Overview of the ThreadedDownloads Class

- Implementation of ThreadedDownloads class & data members

```
public class ThreadedDownloads extends Activity {
  ...

  private final String mDefaultURL = ...

                    User's selection of URL to  download

  private EditText mUrlEditText;


  private ImageView mImageView;


  private ProgressDialog mProgressDialog;
```

Overview of the ThreadedDownloads Class

- Implementation of ThreadedDownloads class & data members

```
public class ThreadedDownloads extends Activity {
  ...

  private final String mDefaultURL = ...


  private EditText mUrlEditText;


  private ImageView mImageView;


  private ProgressDialog mProgressDialog;
```

Image that's been downloaded

Overview of the ThreadedDownloads Class

- Implementation of ThreadedDownloads class & data members

```
public class ThreadedDownloads extends Activity {
  ...

  private final String mDefaultURL = ...


  private EditText mUrlEditText;


  private ImageView mImageView;


  private ProgressDialog mProgressDialog;
```

Display progress of download

Overview of the ThreadedDownloads Class

- Implementation of onCreate() hook method
  - Called by Android runtime to initialize an Activity when it's first created

```
public class ThreadedDownloads extends Activity {
  ...
  public void onCreate(Bundle savedInstanceState) {
    ...


    setContentView(R.layout.main);

    mUrlEditText =
      (EditText) findViewById(R.id.mUrlEditText);



    mImageView =
      (ImageView) findViewById(R.id.mImageView);
```

Overview of the ThreadedDownloads Class

- Implementation of onCreate() hook method
  - Called by Android runtime to initialize an Activity when it's first created

```
public class ThreadedDownloads extends Activity {
  ...
  public void onCreate(Bundle savedInstanceState) {
    ...
```

Set content view specified in main.xml

```
    setContentView(R.layout.main);

    mUrlEditText =
      (EditText) findViewById(R.id.mUrlEditText);



    mImageView =
      (ImageView) findViewById(R.id.mImageView);
```

Overview of the ThreadedDownloads Class

- Implementation of onCreate() hook method
  - Called by Android runtime to initialize an Activity when it's first created

```
public class ThreadedDownloads extends Activity {
  ...
  public void onCreate(Bundle savedInstanceState) {
    ...


    setContentView(R.layout.main);

    mUrlEditText =
      (EditText) findViewById(R.id.mUrlEditText);


    mImageView =
      (ImageView) findViewById(R.id.mImageView);
```

Cache references to EditText & ImageView in data members to optimize subsequent access

Overview of the ThreadedDownloads Class

- Implementation of downloadImage()
  - Shared by all concurrency models

```
public class ThreadedDownloads extends Activity {
  ...
  private Bitmap downloadImage(String url) {
    ...
    try {


      InputStream is = (InputStream) new URL(url).getContent();




      return BitmapFactory.decodeStream(is);
    } catch (Exception e) {
      ...
```

Overview of the ThreadedDownloads Class

- Implementation of downloadImage()
  - Shared by all concurrency models

```
public class ThreadedDownloads extends Activity {
  ...
  private Bitmap downloadImage(String url) {
    ...
    try {

      InputStream is = (InputStream) new URL(url).getContent();



      return BitmapFactory.decodeStream(is);
    } catch (Exception e) {
      ...
```

Runs in background threads & can block without incurring "ANRs"

Overview of the ThreadedDownloads Class

- Implementation of downloadImage()
  - Shared by all concurrency models

```
public class ThreadedDownloads extends Activity {
  ...
  private Bitmap downloadImage(String url) {
    ...
    try {
```

Connect to remote server, download image contents & provide access via an Input Stream

```
      InputStream is = (InputStream) new URL(url).getContent();



      return BitmapFactory.decodeStream(is);
    } catch (Exception e) {
      ...
```

developer.android.com/reference/java/net/URL.html has more info

Overview of the ThreadedDownloads Class

- Implementation of downloadImage()
  - Shared by all concurrency models

```
public class ThreadedDownloads extends Activity {
  ...
  private Bitmap downloadImage(String url) {
    ...
    try {


      InputStream is = (InputStream) new URL(url).getContent();


      Bitmap image = BitmapFactory.decodeStream(is);
      return image;
    } catch (Exception e) {
      ...
```

Decode an InputStream into a Bitmap

See developer.android.com/reference/android/graphics/BitmapFactory.html

Overview of the ThreadedDownloads Class

- Implementation of downloadImage()
  - Shared by all concurrency models

```
public class ThreadedDownloads extends Activity {
  ...
  private Bitmap downloadImage(String url) {
    ...
    try {


      InputStream is = (InputStream) new URL(url).getContent();


      Bitmap image = BitmapFactory.decodeStream(is);
      return image;
    } catch (Exception e) {
      ...
```

Decode an InputStream into a Bitmap

See developer.android.com/reference/android/graphics/Bitmap.html

Overview of the ThreadedDownloads Class

- Implementation of downloadImage()
  - Shared by all concurrency models

```
public class ThreadedDownloads extends Activity {
  ...
  private Bitmap downloadImage(String url) {
    ...
    try {


      InputStream is = (InputStream) new URL(url).getContent();
```

Return Bitmap object containing the image on success

```
      Bitmap image = BitmapFactory.decodeStream(is);
      return image;
    } catch (Exception e) {
      ...
```

Overview of the ThreadedDownloads Class

- Implementation of downloadImage()
  - Shared by all concurrency models

```
public class ThreadedDownloads extends Activity {
  ...
  private Bitmap downloadImage(String url) {
    ...
    try {
      ...
    } catch (Exception e) {
      this.runOnUiThread(new Runnable() {
        public void run() {


          showErrorToast("Error downloading image,"
               + " please check the requested URL.");
        }});
      ...
```

Overview of the ThreadedDownloads Class

- Implementation of downloadImage()
  - Shared by all concurrency models

```
public class ThreadedDownloads extends Activity {
  ...
  private Bitmap downloadImage(String url) {
    ...
    try {
      ...
    } catch (Exception e) {
      this.runOnUiThread(new Runnable() {
        public void run() {
```

Use Toast to inform user something has gone wrong

```
          showErrorToast("Error downloading image,"
                + " please check the requested URL.");
        }});
      ...
```

developer.android.com/guide/topics/ui/notifiers/toasts.html has more info

Overview of the ThreadedDownloads Class

- Implementation of downloadImage()
  - Shared by all concurrency models

```
public class ThreadedDownloads extends Activity {
  ...
  private Bitmap downloadImage(String url) {
    ...
    try {
      ...
    } catch (Exception e) {
      this.runOnUiThread(new Runnable() {
        public void run() {
```

Failures occur in background Thread, but Toasts run in UI Thread

```
          showErrorToast("Error downloading image,"
                + " please check the requested URL.");
        }});
      ...
```

Overview of the ThreadedDownloads Class

- Implementation of downloadImage()
  - Shared by all concurrency models

```
public class ThreadedDownloads extends Activity {
  ...
  private Bitmap downloadImage(String url) {
    ...
    try {
      ...
    } catch (Exception e) {
      this.runOnUiThread(new Runnable() {
        public void run() {


          showErrorToast("Error downloading image,"
               + " please check the requested URL.");
        }});
      ...
```

Post error reports to the UI Thread via Runnable commands

Overview of the ThreadedDownloads Class

- Implementation of downloadImage()
  - Shared by all concurrency models

```
public class ThreadedDownloads extends Activity {
  ...
  private Bitmap downloadImage(String url) {
    ...
    try {
      ...
    } catch (Exception e) {
      this.runOnUiThread(new Runnable() {
        public void run() {


          showErrorToast("Error downloading image,"
               + " please check the requested URL.");
        }});
      ...
```

Post error reports to the UI Thread via Runnable commands

Overview of the ThreadedDownloads Class

- Implementation of downloadImage()
  - Shared by all concurrency models

```
public class Activity ... {
  ...
  final Handler mHandler = new Handler();

  public final void runOnUiThread(Runnable action) {
    if (Thread.currentThread() != mUiThread) {
            mHandler.post(action);
        } else {
            action.run();
        }
    }
  }
```

Post a command to UI Thread via an internal Handler

See earlier part on “Posting & Processing Runnables with Android Handler”

Overview of the ThreadedDownloads Class

- Implementation of downloadImage()
  - Shared by all concurrency models

```
public class ThreadedDownloads extends Activity {
  ...
  private Bitmap downloadImage(String url) {
    ...
    try {
      ...
    } catch (Exception e) {
      this.runOnUiThread(new Runnable() {
        public void run() {
```

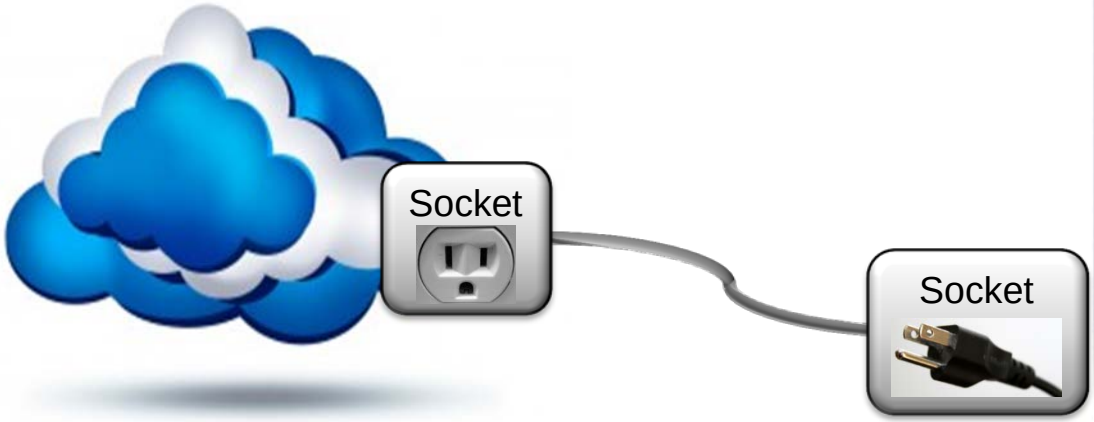
Inform user that something's gone wrong with the download

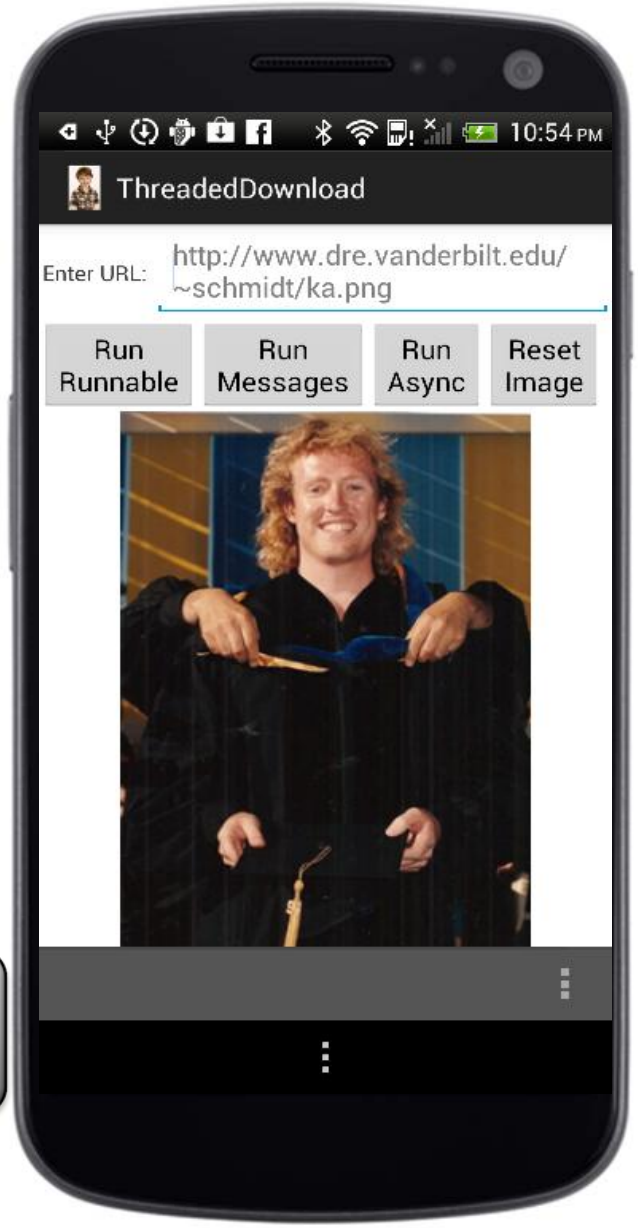
```
          showErrorToast("Error downloading image,"
                + " please check the requested URL.");
        }});
      ...
```

Summary

Summary

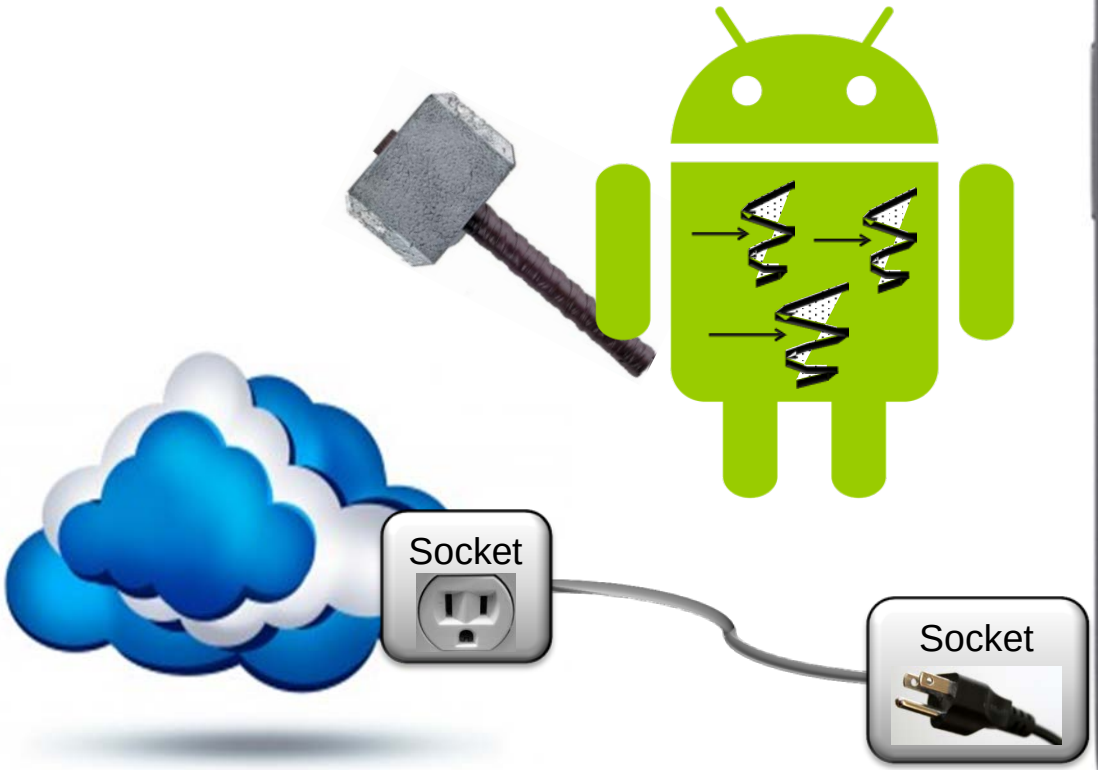
- The Threaded Downloads application retrieves images from remote servers

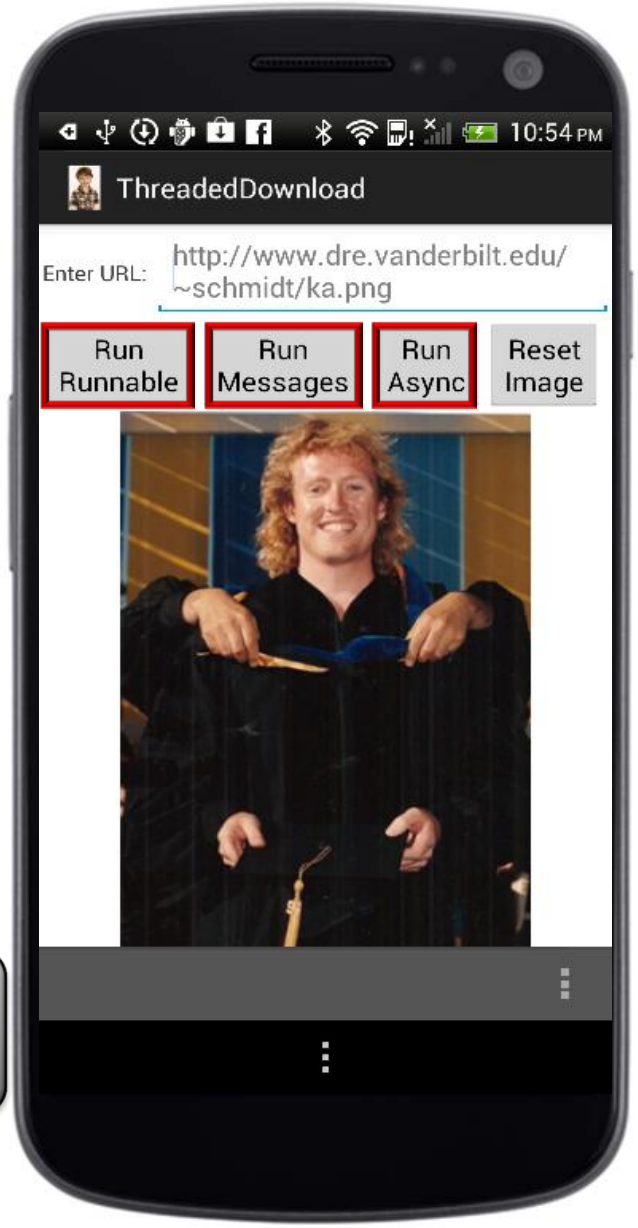




Summary

- The Threaded Downloads application retrieves images from remote servers
  - It uses three concurrency models based on the HaMeR & AsyncTask frameworks





Summary

- The Threaded Downloads application retrieves images from remote servers
- Analyzed structure & functionality of the Threaded Downloads project

File Edit Refactor Source Navigate Search Project Run Window Help

Quick Access | Java | DDMS

Package Explorer | Navigator

- AtomicLong
- ThreadedDownloads
  - Android 4.2.2
  - src
    - edu.vuum.mocca
      - ThreadedDownloads.java
  - gen [Generated Java Files]
  - assets
  - bin
  - libs
  - res
    - drawable-hdpi
    - drawable-ldpi
    - drawable-mdpi
    - drawable-xhdpi
    - layout
      - main.xml
    - menu
    - values
    - values-v11
    - values-v14
  - AndroidManifest.xml
  - ic_launcher-web.png
  - proguard-project.txt

main.xml | ThreadedDownloads Manifest | ThreadedDownloads.java

```
 *
 * @brief A class that allows a user to download a bitmap image using
 *        any of the following concurrency models: Handlers and
 *        Runnables, Handlers and Messages, and AsyncTask.
 *
 */
public class ThreadedDownloads extends Activity
{
    /**
     * Default URL to download
     */
    private final String mDefaultURL = "http://www.dre.vanderbilt.edu/~schmidt/

    /**
     * User's selection of URL to download
     */
    private EditText mUrlEditText;

    /**
     * Image that's been downloaded
     */
    private static ImageView mImageView;

    /**
     * Display progress of download
     */
    private static ProgressDialog mProgressDialog;

    /**
```

Outline

- edu.vuum.mocca
- ThreadedDownloads
  - mDefaultURL : String
  - mUrlEditText : EditText
  - mImageView : ImageView
  - mProgressDialog : ProgressDialog
  - TAG : String
  - mThisActivityContext : Context
  - onCreate(Bundle) : void
  - showErrorToast(String) : void
  - displayImage(Bitmap) : void
  - downloadImage(String) : Bitmap
    - new Runnable() {...}
  - runRunnable(View) : void
  - RunnableWithHandlers
  - runMessages(View) : void
  - MessageHandler
    - SHOW_DIALOG : int
    - DISMISS_DIALOG : int
    - DISPLAY_IMAGE : int
    - mActivity : WeakReference<ThreadedDownloa
    - MessageHandler(ThreadedDownloads)
    - handleMessage(Message) : void
  - messageHandler : MessageHandler
  - RunnableWithMessages

160M of 389M

Summary

- The Threaded Downloads application retrieves images from remote servers
- Analyzed structure & functionality of the Threaded Downloads project

File Edit Refactor Source Navigate Search Project Run Window Help

Package Explorer | Navigator

- AtomicLong
- ThreadedDownloads
  - Android 4.2.2
  - src
    - edu.vuum.mocca
      - ThreadedDownloads.java
  - gen [Generated Java Files]
  - assets
  - bin
  - libs
  - res
    - drawable-hdpi
    - drawable-ldpi
    - drawable-mdpi
    - drawable-xhdpi
    - layout
      - main.xml
    - menu
    - values
    - values-v11
    - values-v14
  - AndroidManifest.xml
  - ic_launcher-web.png
  - proguard-project.txt

main.xml | ThreadedDownloads Manifest | ThreadedDownloads.java

```
 *
 * @brief A class that allows a user to download a bitmap image using
 *        any of the following concurrency models: Handlers and
 *        Runnables, Handlers and Messages, and AsyncTask.
 *
 */
public class ThreadedDownloads extends Activity
{
    /**
     * Default URL to download
     */
    private final String mDefaultURL = "http://www.dre.vanderbilt.edu/~schmidt/

    /**
     * User's selection of URL to download
     */
    private EditText mUrlEditText;

    /**
     * Image that's been downloaded
     */
    private static ImageView mImageView;

    /**
     * Display progress of download
     */
    private static ProgressDialog mProgressDialog;

    /**
```

Outline

- edu.vuum.mocca
- ThreadedDownloads
  - mDefaultURL : String
  - mUrlEditText : EditText
  - mImageView : ImageView
  - mProgressDialog : ProgressDialog
  - TAG : String
  - mThisActivityContext : Context
  - onCreate(Bundle) : void
  - showErrorToast(String) : void
  - displayImage(Bitmap) : void
  - downloadImage(String) : Bitmap
    - new Runnable() {...}
  - runRunnable(View) : void
  - RunnableWithHandlers
  - runMessages(View) : void
  - MessageHandler
    - SHOW_DIALOG : int
    - DISMISS_DIALOG : int
    - DISPLAY_IMAGE : int
    - mActivity : WeakReference<ThreadedDownloa
    - MessageHandler(ThreadedDownloads)
    - handleMessage(Message) : void
  - messageHandler : MessageHandler
  - RunnableWithMessages

160M of 389M

Summary

- The Threaded Downloads application retrieves images from remote servers
- Analyzed structure & functionality of the Threaded Downloads project

File Edit Refactor Source Navigate Search Project Run Window Help

Package Explorer | Navigator

- AtomicLong
- ThreadedDownloads
  - Android 4.2.2
  - src
    - edu.vuum.mocca
      - ThreadedDownloads.java
  - gen [Generated Java Files]
  - assets
  - bin
  - libs
  - res
    - drawable-hdpi
    - drawable-ldpi
    - drawable-mdpi
    - drawable-xhdpi
    - layout
      - main.xml
    - menu
    - values
    - values-v11
    - values-v14
  - AndroidManifest.xml
  - ic_launcher-web.png
  - proguard-project.txt

main.xml | ThreadedDownloads Manifest | ThreadedDownloads.java

```
 *
 * @brief A class that allows a user to download a bitmap image using
 *        any of the following concurrency models: Handlers and
 *        Runnables, Handlers and Messages, and AsyncTask.
 *
 */
public class ThreadedDownloads extends Activity
{
    /**
     * Default URL to download
     */
    private final String mDefaultURL = "http://www.dre.vanderbilt.edu/~schmidt

    /**
     * User's selection of URL to download
     */
    private EditText mUrlEditText;

    /**
     * Image that's been downloaded
     */
    private static ImageView mImageView;

    /**
     * Display progress of download
     */
    private static ProgressDialog mProgressDialog;

    /**
```

Outline

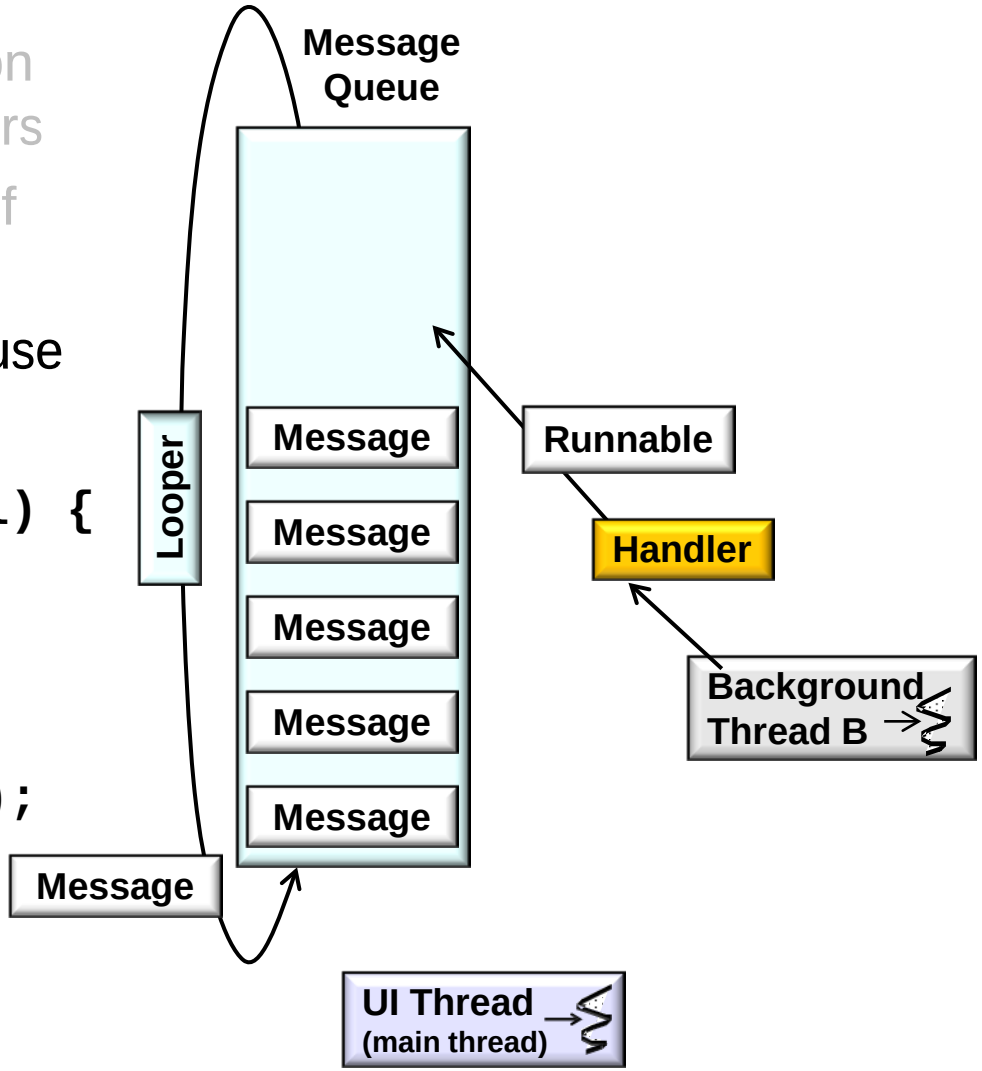
- edu.vuum.mocca
- ThreadedDownloads
  - mDefaultURL : String
  - mUrlEditText : EditText
  - mImageView : ImageView
  - mProgressDialog : ProgressDialog
  - TAG : String
  - mThisActivityContext : Context
  - onCreate(Bundle) : void
  - showErrorToast(String) : void
  - displayImage(Bitmap) : void
  - downloadImage(String) : Bitmap
    - new Runnable() {...}
  - runRunnable(View) : void
  - RunnableWithHandlers
  - runMessages(View) : void
  - MessageHandler
    - SHOW_DIALOG : int
    - DISMISS_DIALOG : int
    - DISPLAY_IMAGE : int
    - mActivity : WeakReference<ThreadedDownloa
    - MessageHandler(ThreadedDownloads)
    - handleMessage(Message) : void
  - messageHandler : MessageHandler
  - RunnableWithMessages

160M of 389M

Summary

- The Threaded Downloads application retrieves images from remote servers
- Analyzed structure & functionality of the Threaded Downloads project
- Showed simple HaMeR framework use

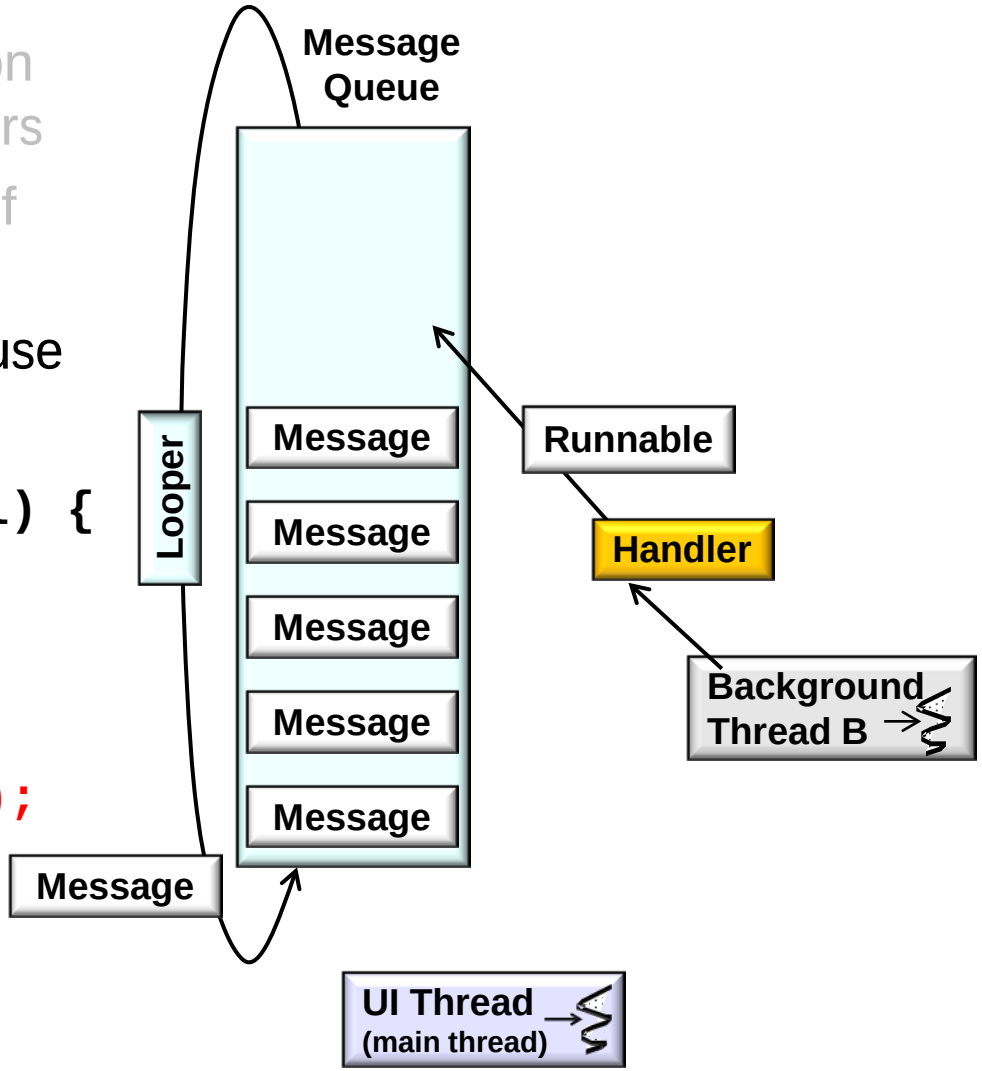
```
private Bitmap downloadImage
                    (String url) {
  ...
  this.runOnUiThread
    (new Runnable() {
      public void run() {
        showErrorToast("...");
    }
  });
  ...
```



Summary

- The Threaded Downloads application retrieves images from remote servers
- Analyzed structure & functionality of the Threaded Downloads project
- Showed simple HaMeR framework use

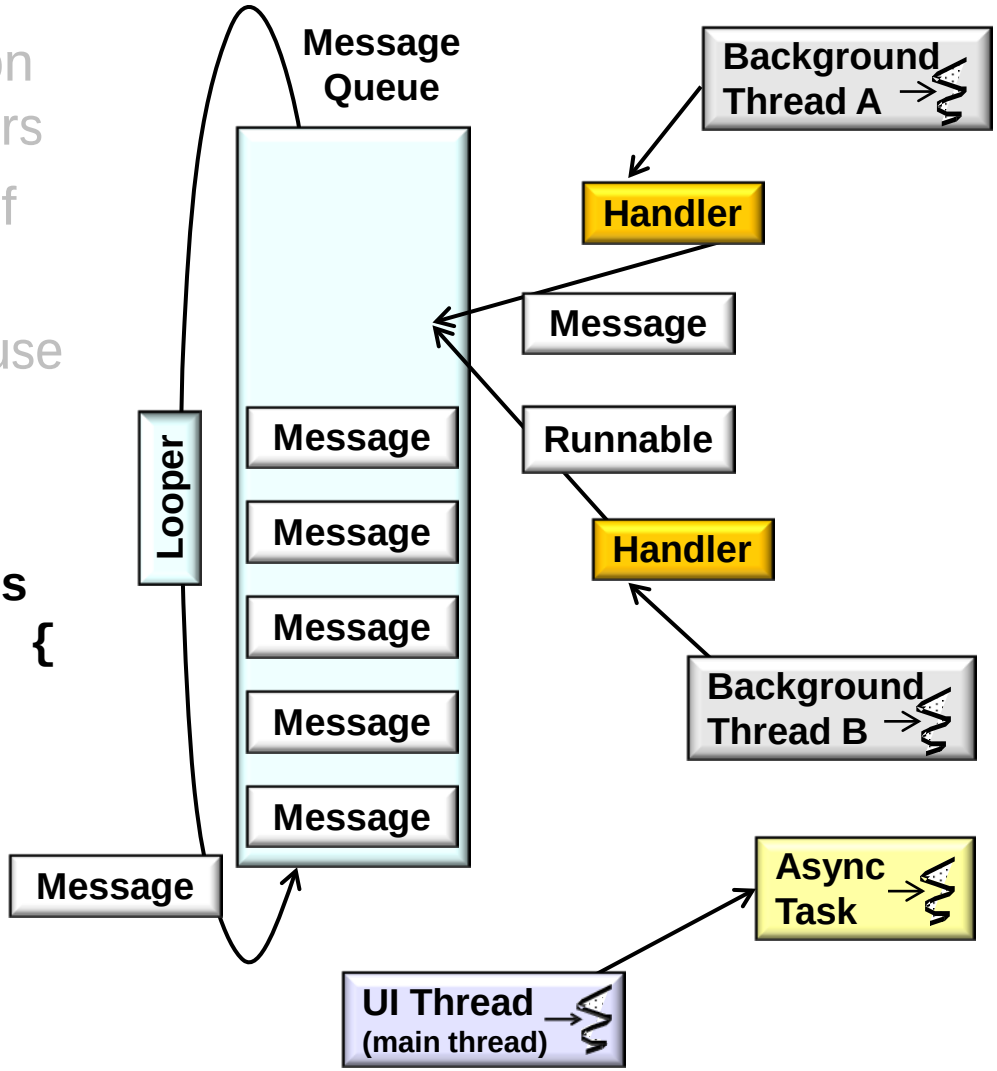
```
private Bitmap downloadImage
                    (String url) {
  ...
  this.runOnUiThread
    (new Runnable() {
      public void run() {
        showErrorToast("...");
      }
  });
  ...
```



Summary

- The Threaded Downloads application retrieves images from remote servers
- Analyzed structure & functionality of the Threaded Downloads project
- Showed simple HaMeR framework use
- Other methods use more HaMeR & AsyncTask framework capabilities

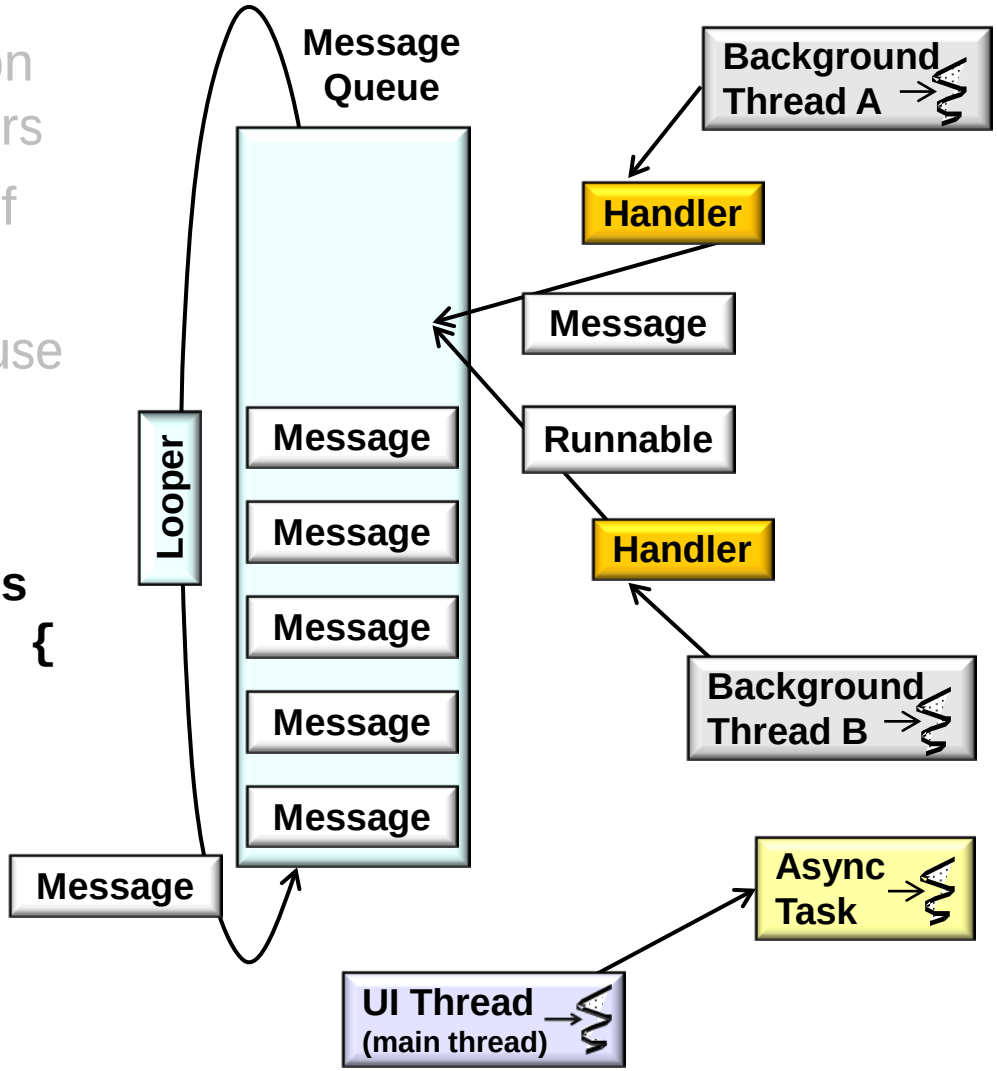
```
public class ThreadedDownloads
              extends Activity {
  ...
  public void runRunnable
    (View view) {...}
  public void runMessages
    (View view) {...}
  public void runAsyncTask
    (View view) {...}
```



Summary

- The Threaded Downloads application retrieves images from remote servers
- Analyzed structure & functionality of the Threaded Downloads project
- Showed simple HaMeR framework use
- Other methods use more HaMeR & AsyncTask framework capabilities

```
public class ThreadedDownloads
              extends Activity {
  ...
  public void runRunnable
    (View view) {...}
  public void runMessages
    (View view) {...}
  public void runAsyncTask
    (View view) {...}
```



See next Part 2 on “Programming with Android Concurrency Frameworks”