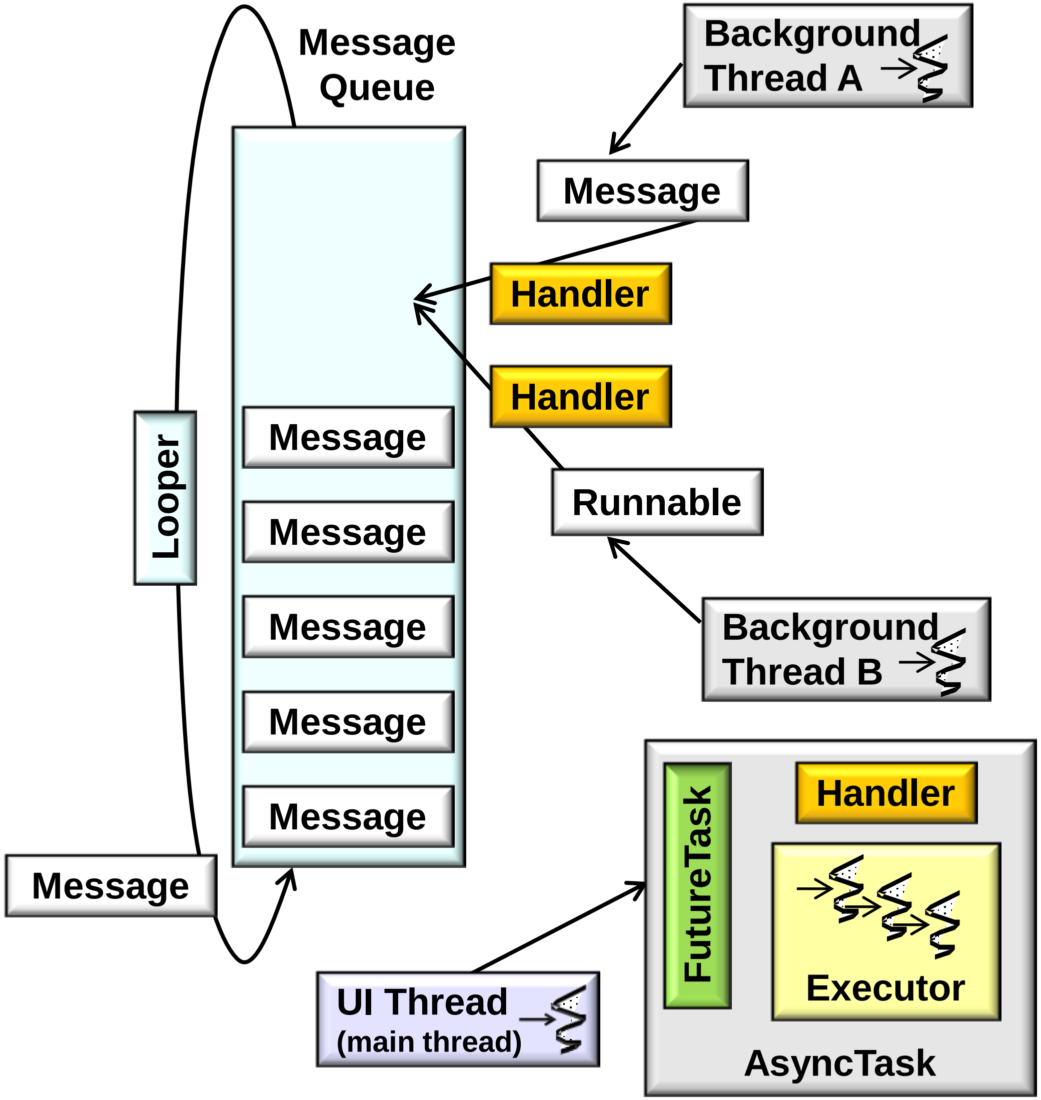
# Android Concurrency: Overview of Android Concurrency Frameworks & Idioms

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 the pattern-oriented structure & functionality of Android concurrency frameworks

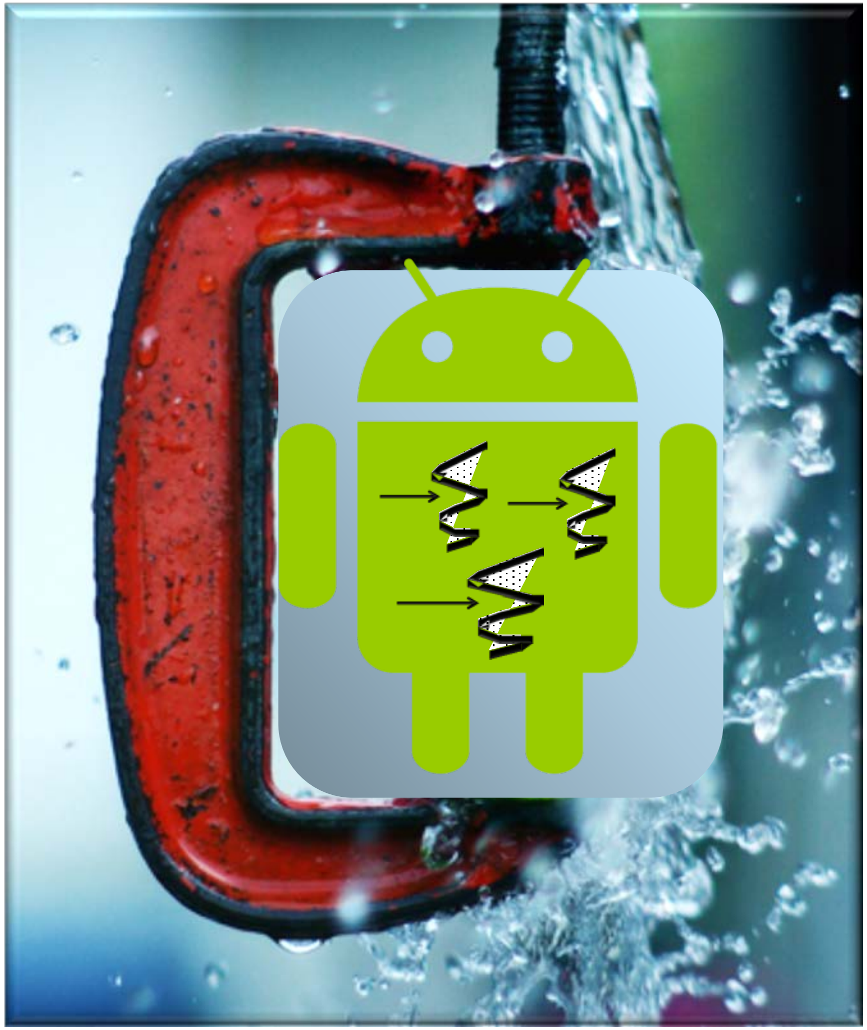


See earlier parts on “Overview of Patterns and Frameworks”

# Motivation for Android Concurrency Frameworks

# Motivation for Android Concurrency Frameworks

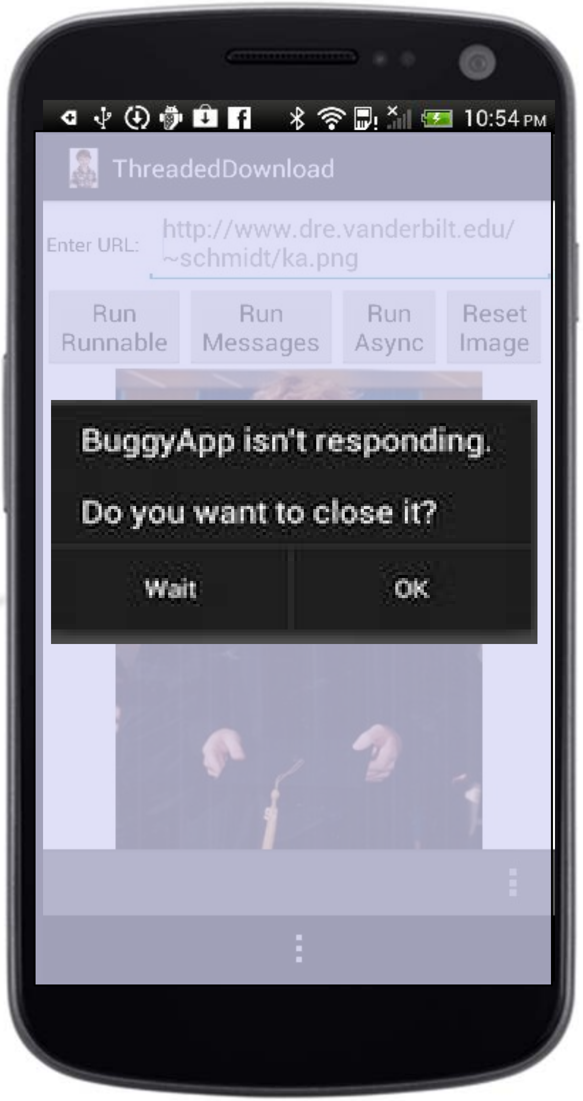
- Address design constraints of concurrent Android software



developer.android.com/guide/components/processes-and-threads.html#Threads

# Motivation for Android Concurrency Frameworks

- Address design constraints of concurrent Android software
  - An “Application Not Responding” dialog is generated if app’s UI Thread doesn’t respond to user input within a short time



BuggyApp isn't responding.

Do you want to close it?

*The UI Thread can’t block on long-duration operations*

See developer.android.com/training/articles/perf-anr.html for more on ANRs

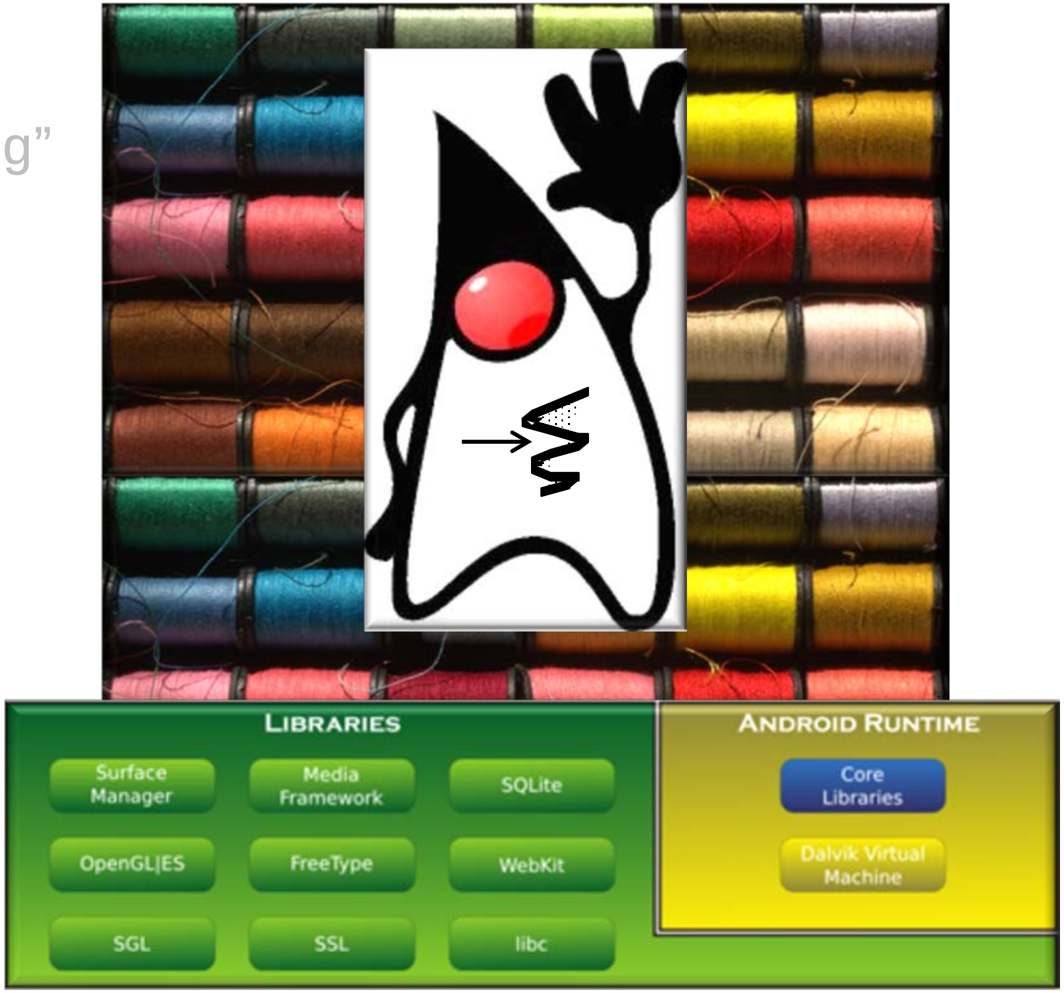
# Motivation for Android Concurrency Frameworks

- Address design constraints of concurrent Android software
  - An “Application Not Responding” dialog is generated if app’s UI Thread doesn’t respond to user input within a short time
  - Non-UI Threads can’t access components in the UI toolkit since they aren’t thread-safe

android-developers.blogspot.com/2009/05/painless-threading.html has more

# Motivation for Android Concurrency Frameworks

- Address design constraints of concurrent Android software
  - An “Application Not Responding” dialog is generated if app’s UI Thread doesn’t respond to user input within a short time
  - Non-UI Threads can’t access components in the UI toolkit since they aren’t thread-safe
  - Java concurrency mechanisms alone don’t address these constraints



See earlier Module on “Java Concurrency Mechanisms”

# Motivation for Android Concurrency Frameworks

- Address design constraints of concurrent Android software
- Improve software quality attributes

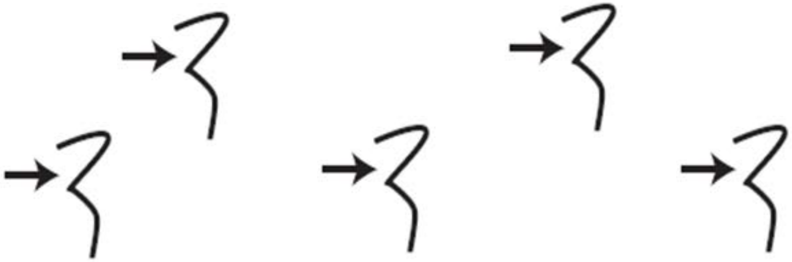
See earlier part on “Motivations for Concurrency”

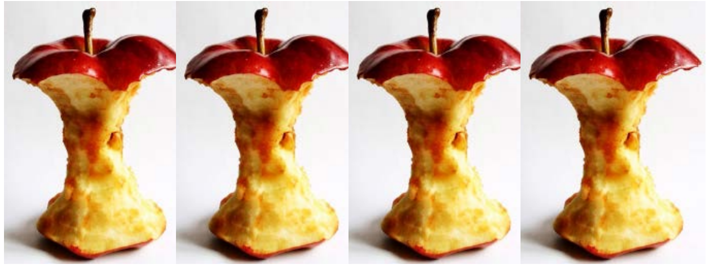
# Motivation for Android Concurrency Frameworks

- Address design constraints of concurrent Android software
- Improve software quality attributes, e.g.
  - Simplify program structure

# Motivation for Android Concurrency Frameworks

- Address design constraints of concurrent Android software
- Improve software quality attributes, e.g.
  - Simplify program structure
  - Increase performance



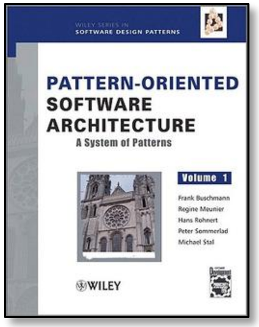


# Motivation for Android Concurrency Frameworks

- Address design constraints of concurrent Android software
- Improve software quality attributes, e.g.
  - Simplify program structure
  - Increase performance
  - Improve responsiveness

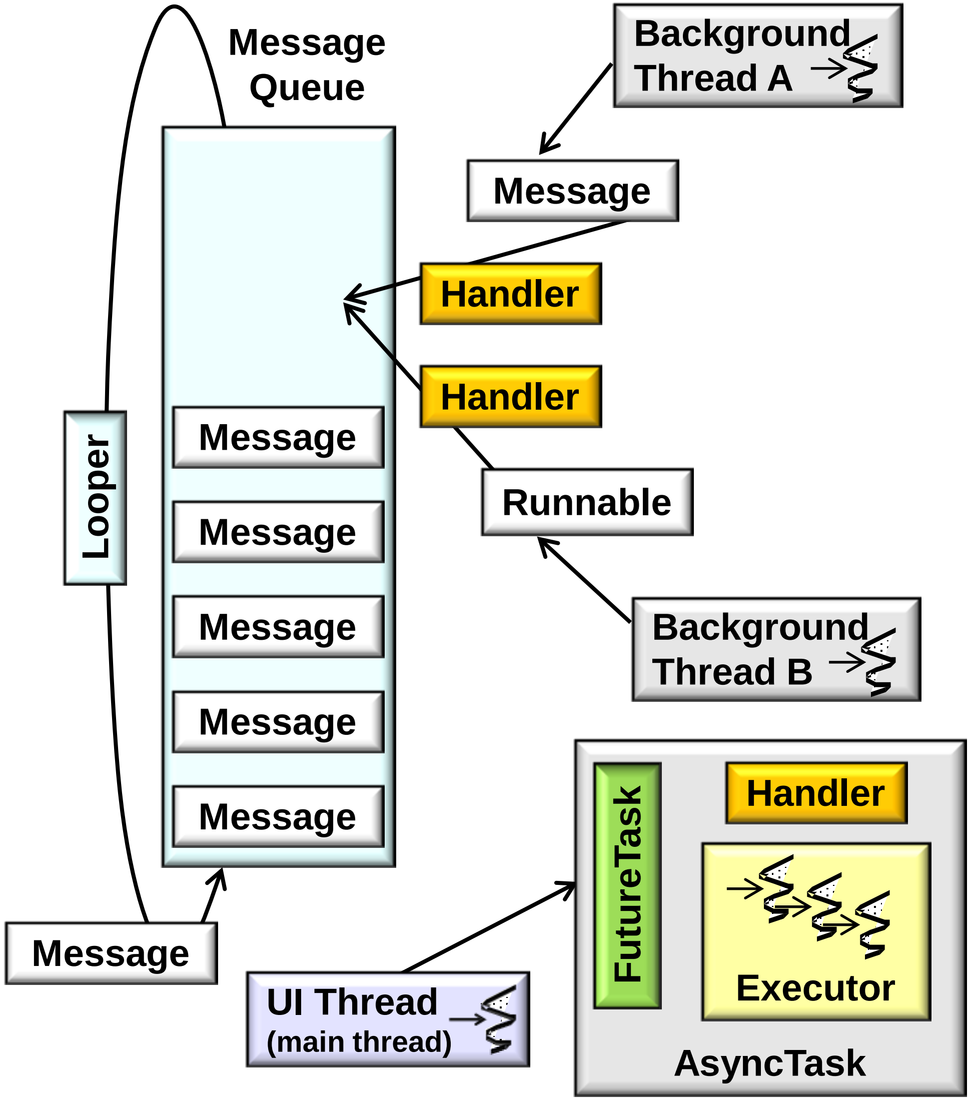
# Motivation for Android Concurrency Frameworks

- Address design constraints of concurrent Android software
- Improve software quality attributes
- Many patterns applied to overcome design constraints & provide other benefits of concurrency



Douglas Schmidt
Michael Stal
Hans Rohnert
Frank Buschmann
PATTERN-ORIENTED SOFTWARE ARCHITECTURE
Volume 2
Patterns for Concurrent and Networked Objects
WILEY
WILEY SERIES IN SOFTWARE DESIGN PATTERNS

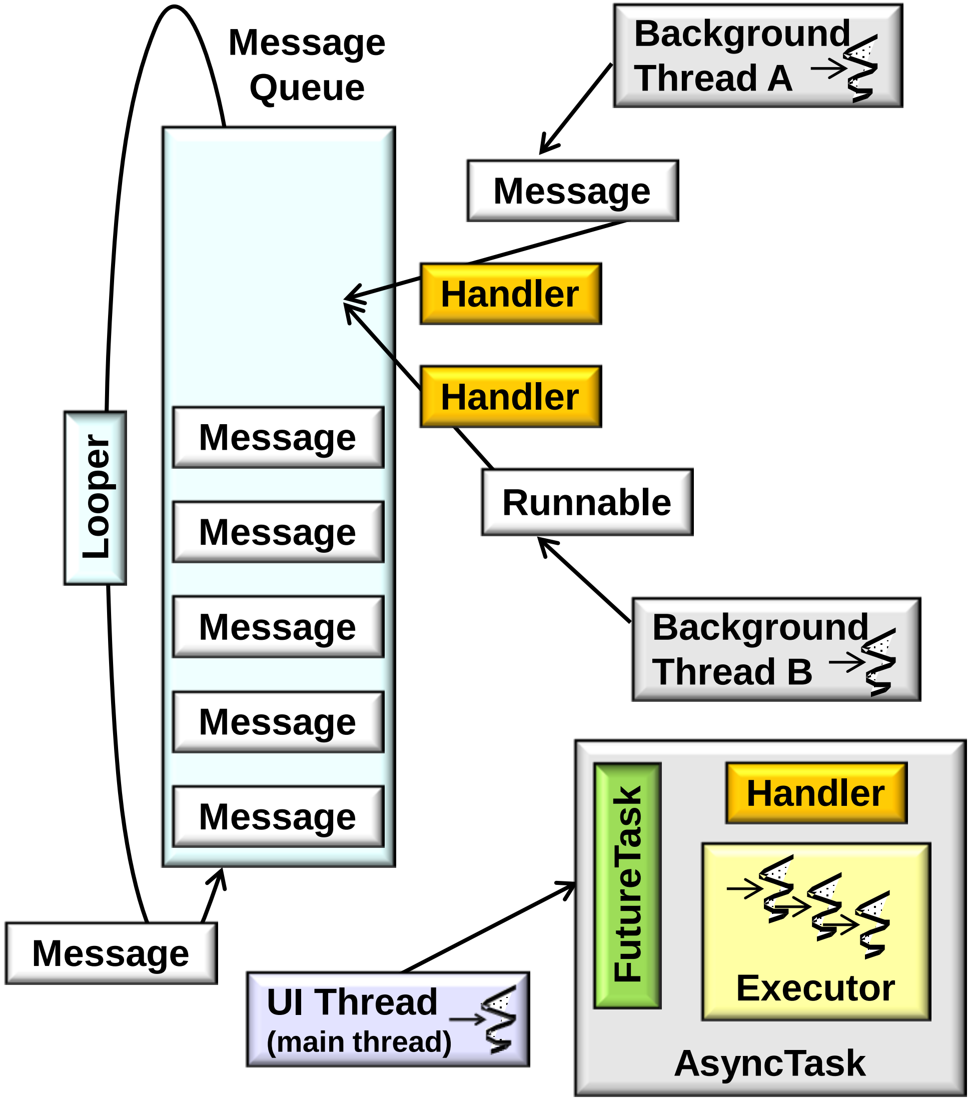
Design Patterns
Elements of Reusable Object-Oriented Software
Erich Gamma
Richard Helm
Ralph Johnson
John Vlissides
Foreword by Grady Booch
ADDISON-WESLEY PROFESSIONAL COMPUTING SERIES

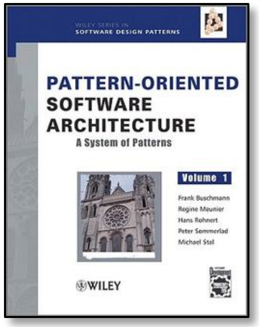


See earlier part on “Overview of Patterns and Frameworks (Part 2)”

# Motivation for Android Concurrency Frameworks

- Address design constraints of concurrent Android software
- Improve software quality attributes
- Many patterns applied to overcome design constraints & provide other benefits of concurrency



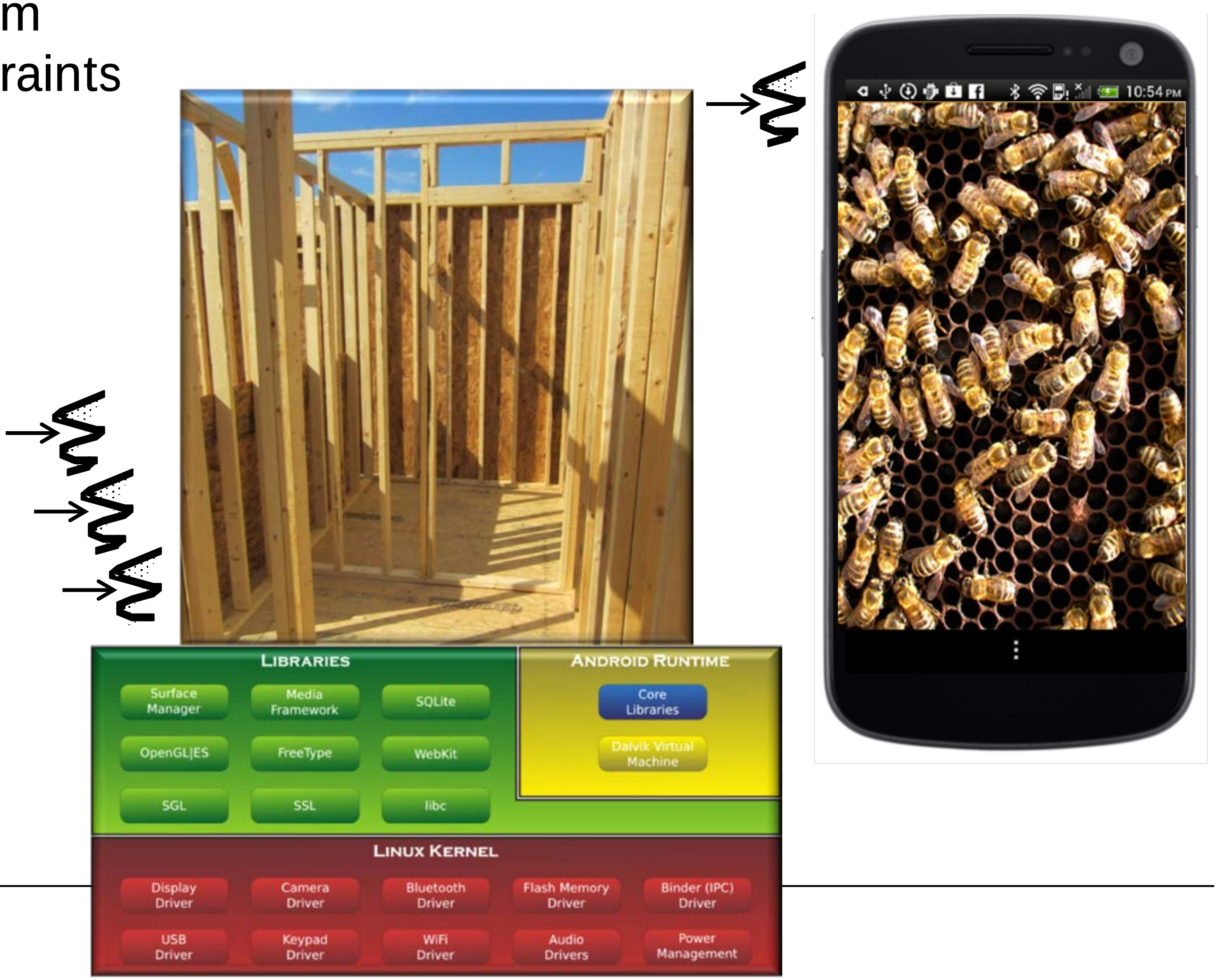


See upcoming section on “Concurrency & Communication Patterns in Android”

# Overview of Android Concurrency Frameworks

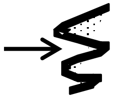
# Overview of Android Concurrency Frameworks

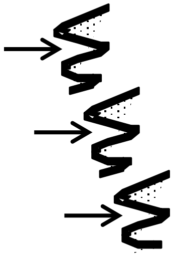
- Android provides several concurrency frameworks that
  - Shield developers from Android design constraints

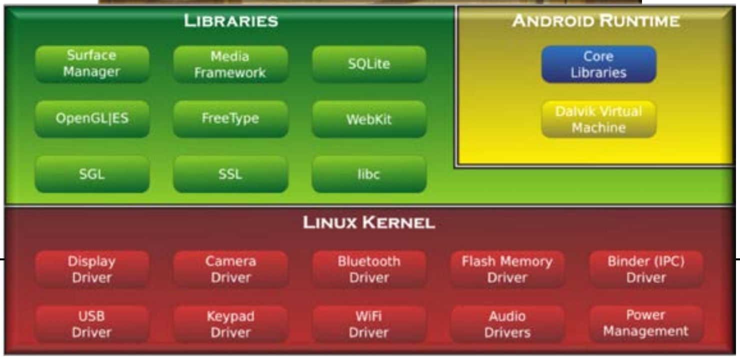


# Overview of Android Concurrency Frameworks

- Android provides several concurrency frameworks that
  - Shield developers from Android design constraints
  - Enhance software quality attributes



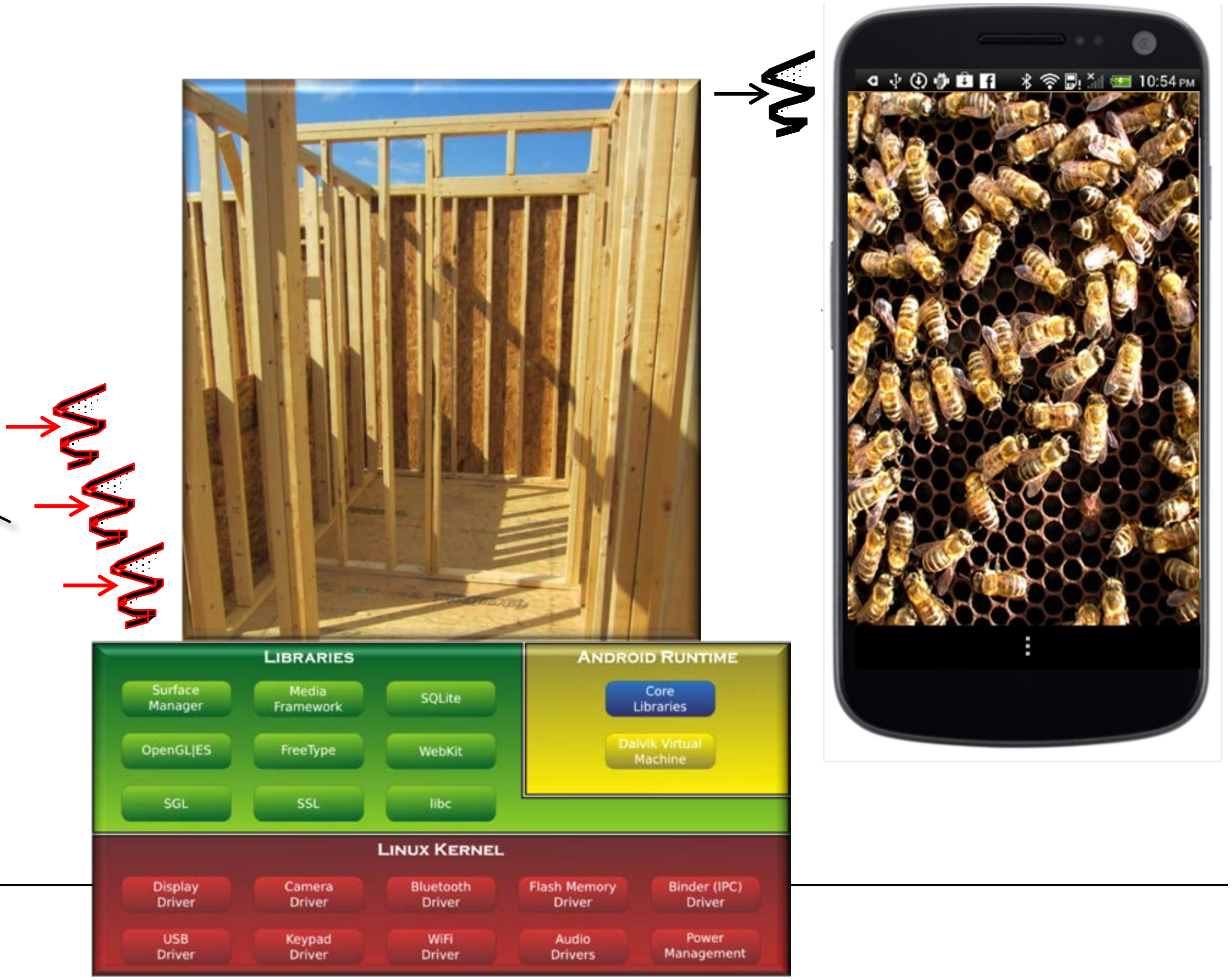




# Overview of Android Concurrency Frameworks

- Android provides several concurrency frameworks

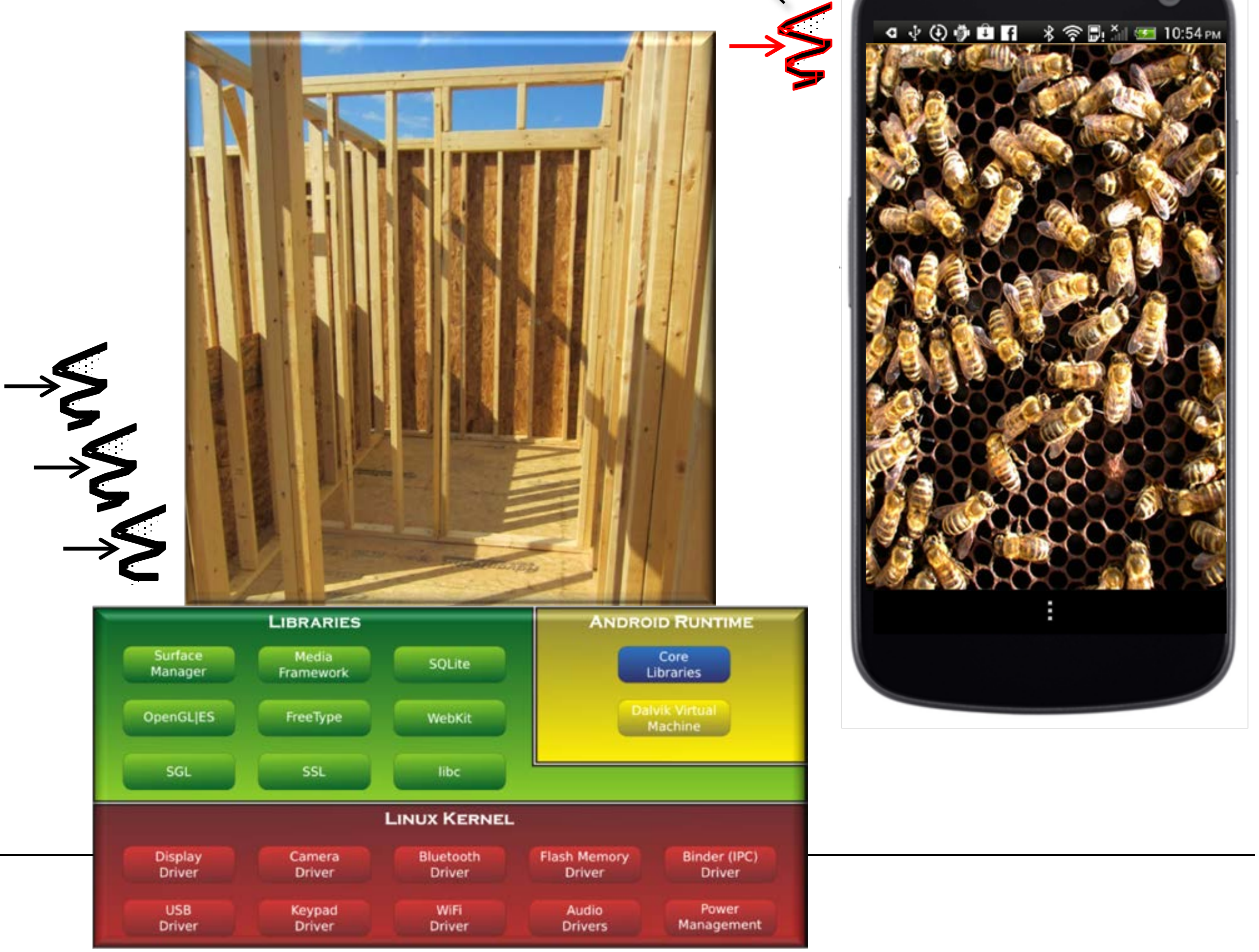
*Long-duration & potentially blocking operations run in background Threads*



# Overview of Android Concurrency Frameworks

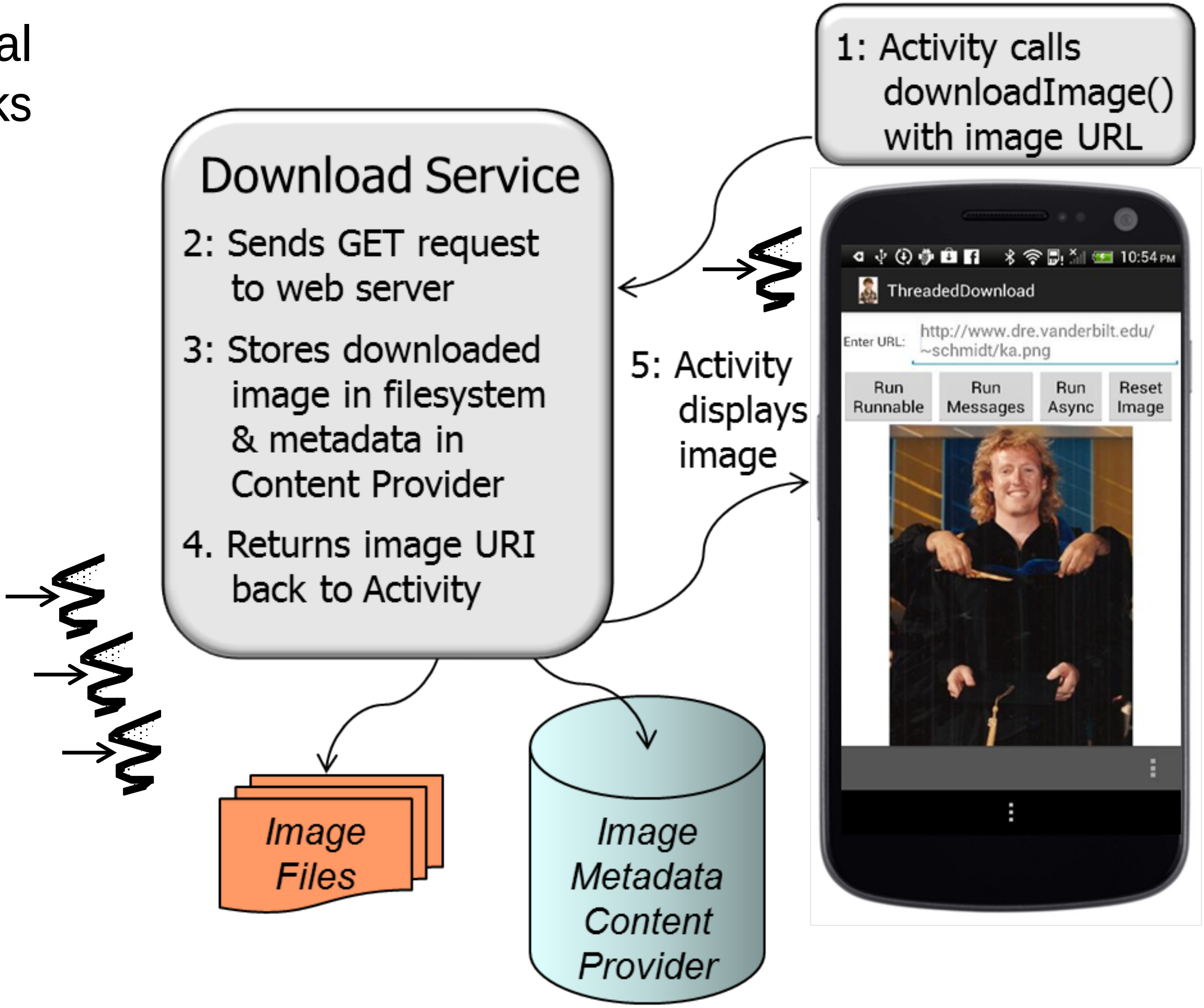
- Android provides several concurrency frameworks

*Short-duration, user-facing operations run in the UI Thread*



# Overview of Android Concurrency Frameworks

- Android provides several concurrency frameworks



1: Activity calls downloadImage() with image URL

Download Service

2: Sends GET request to web server

3: Stores downloaded image in filesystem & metadata in Content Provider

4. Returns image URI back to Activity

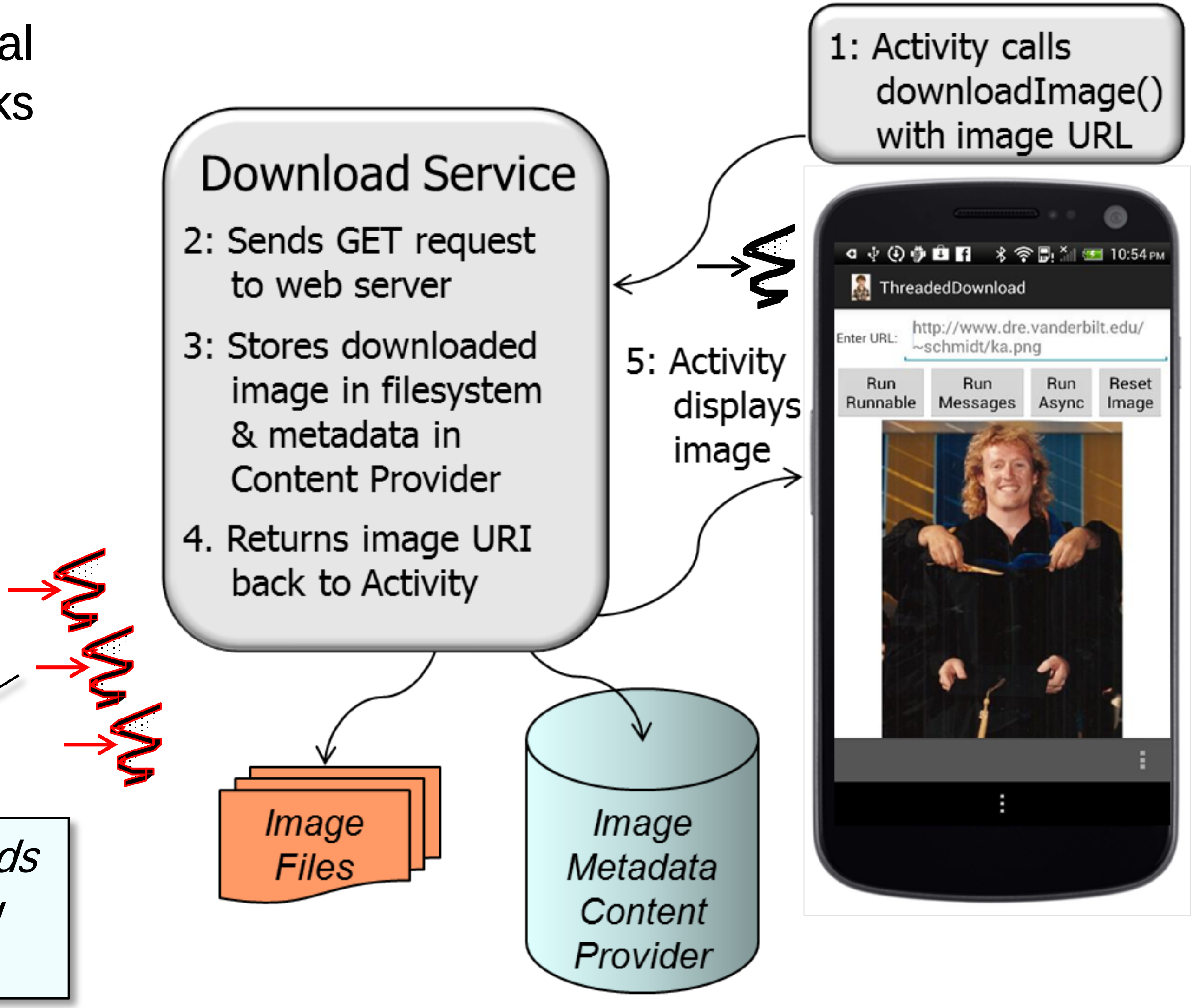
5: Activity displays image

Enter URL: http://www.dre.vanderbilt.edu/~schmidt/ka.png

See earlier part on "Motivations for Concurrency"

# Overview of Android Concurrency Frameworks

- Android provides several concurrency frameworks



1: Activity calls downloadImage() with image URL

Download Service

2: Sends GET request to web server

3: Stores downloaded image in filesystem & metadata in Content Provider

4. Returns image URI back to Activity

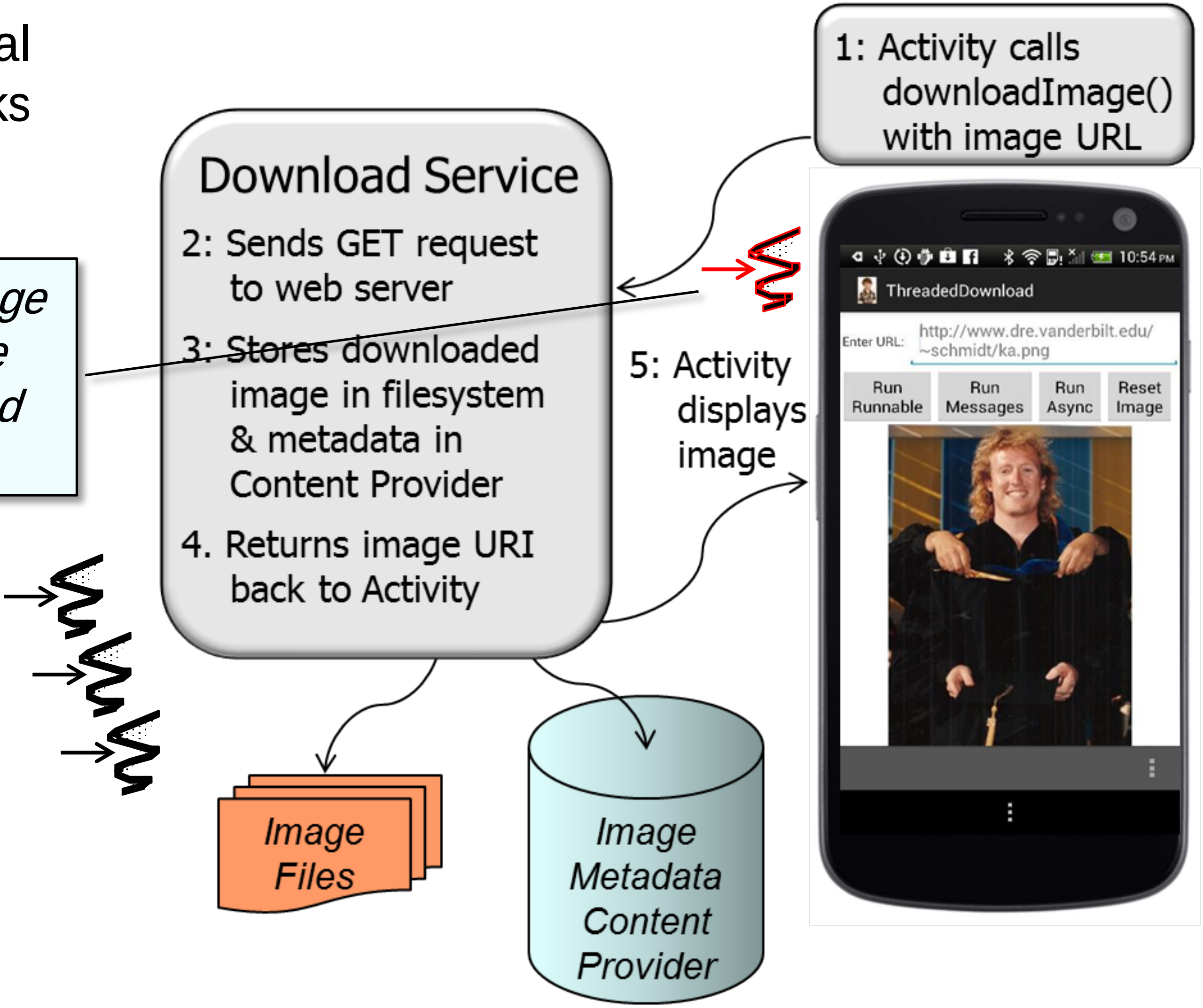
5: Activity displays image

*Spawn background threads to process long-running blocking operations*

# Overview of Android Concurrency Frameworks

- Android provides several concurrency frameworks

*Use synchronized message queue to communicate results from background threads to UI Thread*



Download Service

2: Sends GET request to web server

3: Stores downloaded image in filesystem & metadata in Content Provider

4. Returns image URI back to Activity

1: Activity calls downloadImage() with image URL

5: Activity displays image

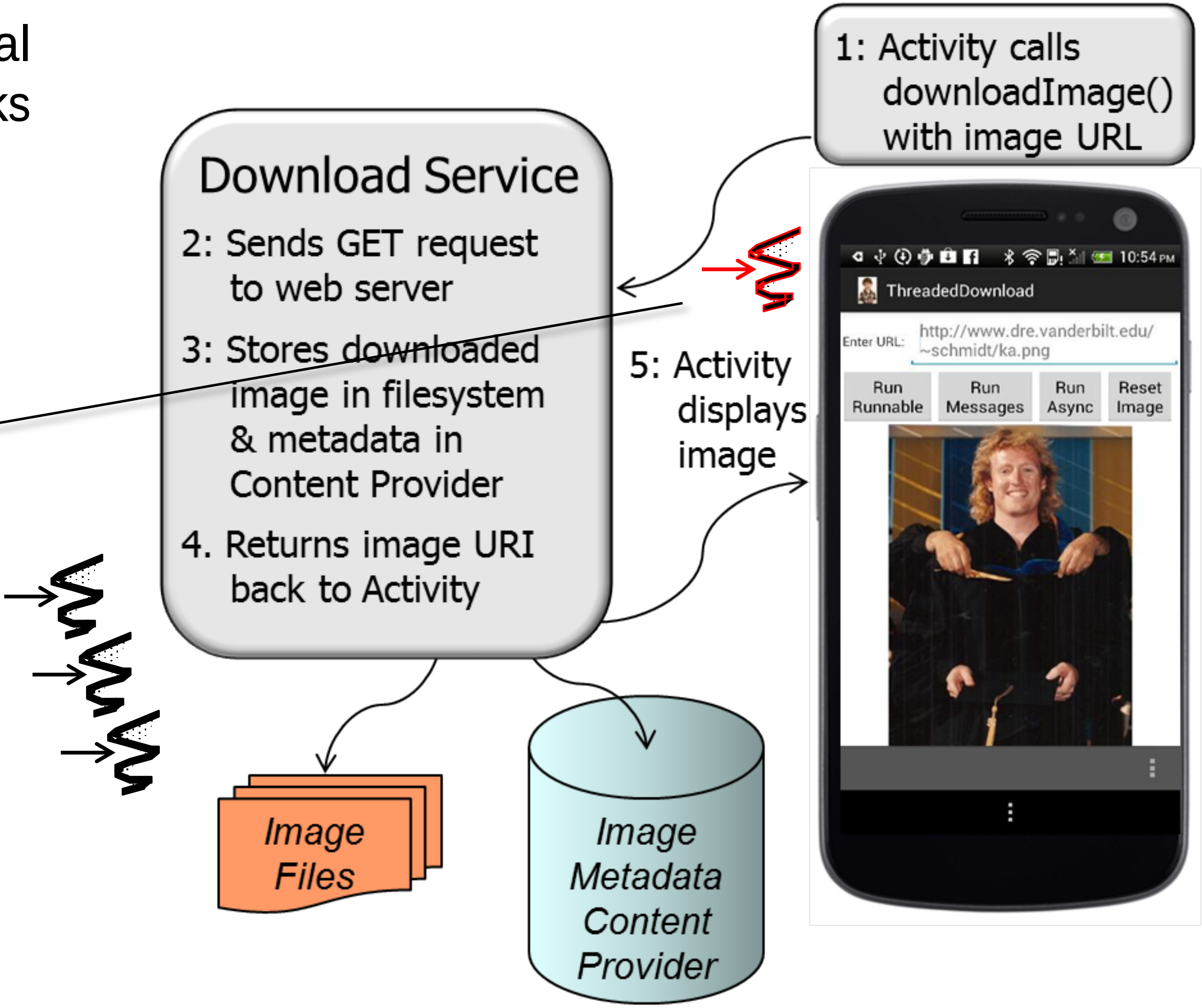
*Image Files*

*Image Metadata Content Provider*

# Overview of Android Concurrency Frameworks

- Android provides several concurrency frameworks

*The UI Thread can also pass messages to itself!*



1: Activity calls downloadImage() with image URL

Download Service

2: Sends GET request to web server

3: Stores downloaded image in filesystem & metadata in Content Provider

4. Returns image URI back to Activity

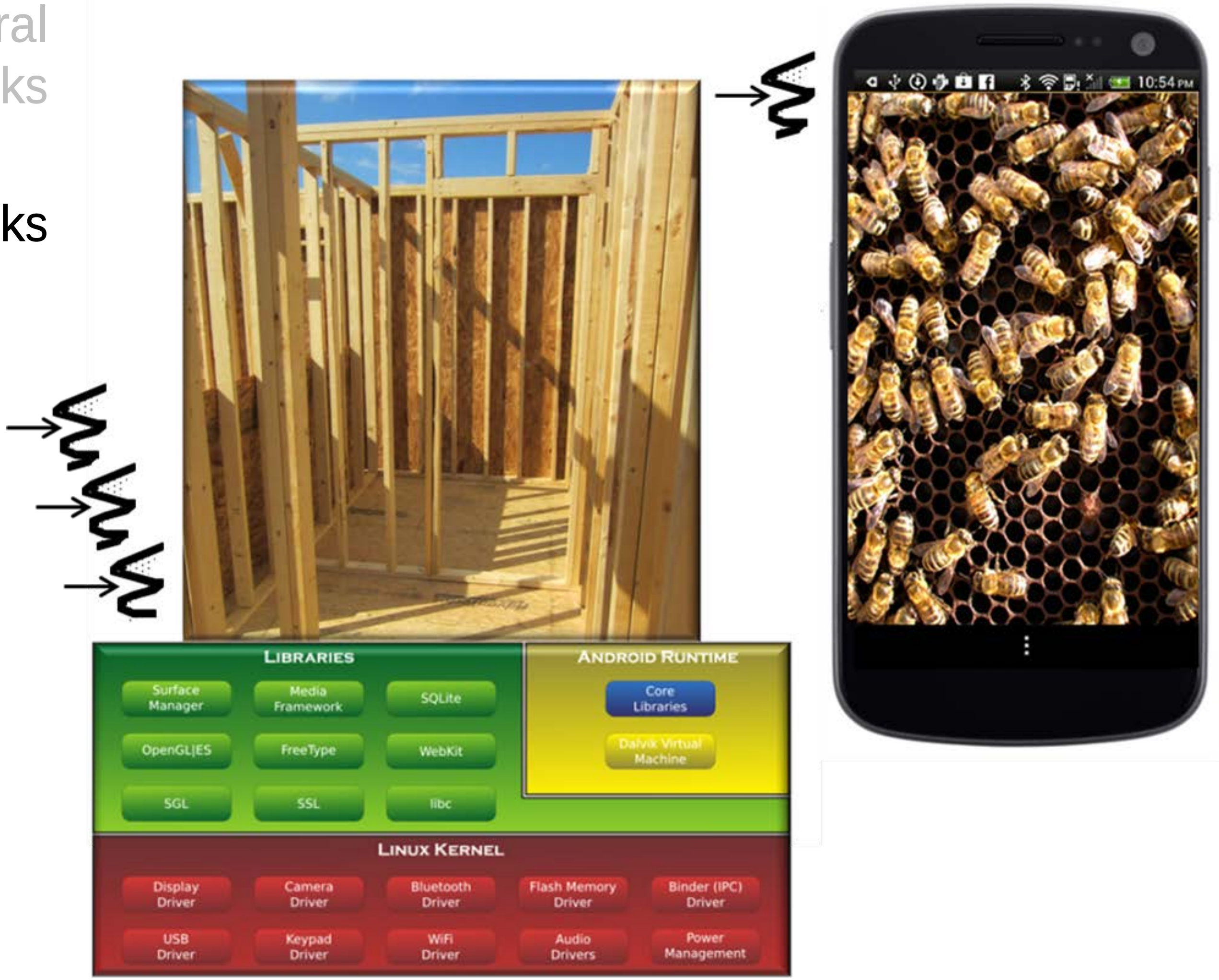
5: Activity displays image

*Image Files*

*Image Metadata Content Provider*

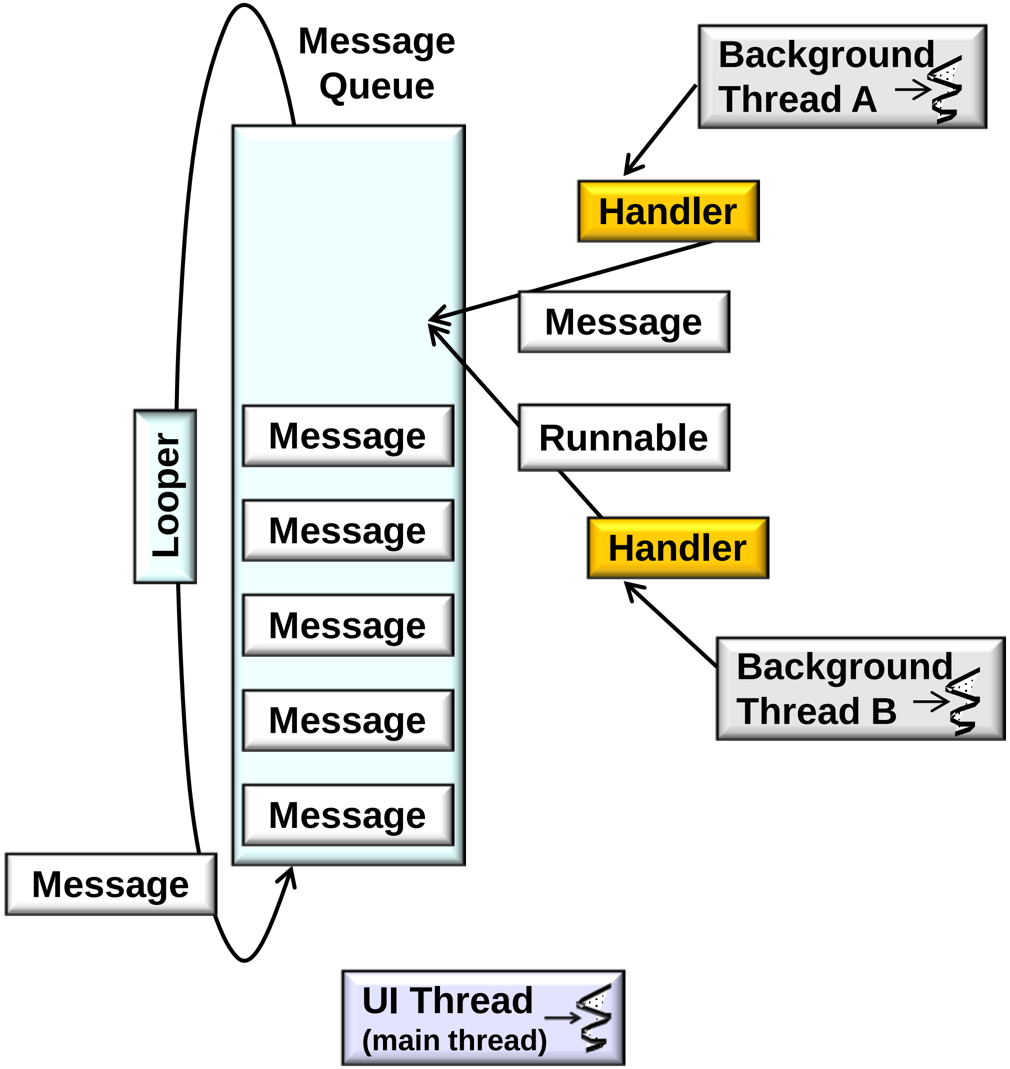
# Overview of Android Concurrency Frameworks

- Android provides several concurrency frameworks
- Android's two primary concurrency frameworks are



# Overview of Android Concurrency Frameworks

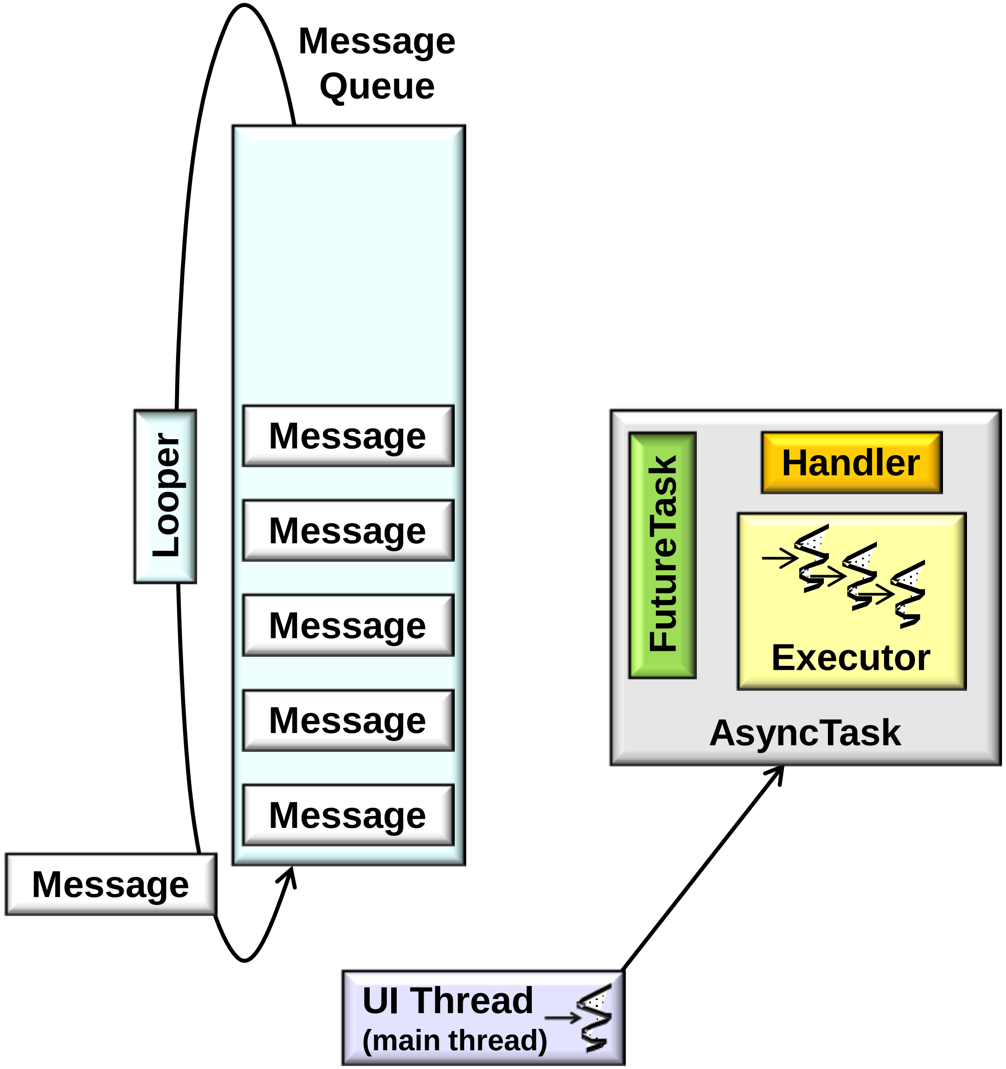
- Android provides several concurrency frameworks
- Android's two primary concurrency frameworks are
  - **Handlers, Messages, & Runnables (HaMeR)**
    - Allows operations to run in one or more background threads that publish their results to the UI thread



See developer.android.com/training/multiple-threads/communicate-ui.html

# Overview of Android Concurrency Frameworks

- Android provides several concurrency frameworks
- Android's two primary concurrency frameworks are
  - **Handlers, Messages, & Runnables (HaMeR)**
  - **AsyncTask**
    - Allows operations to run in one or more background threads & publish results to the UI thread without manipulating threads or handlers



developer.android.com/reference/android/os/AsyncTask.html has more info

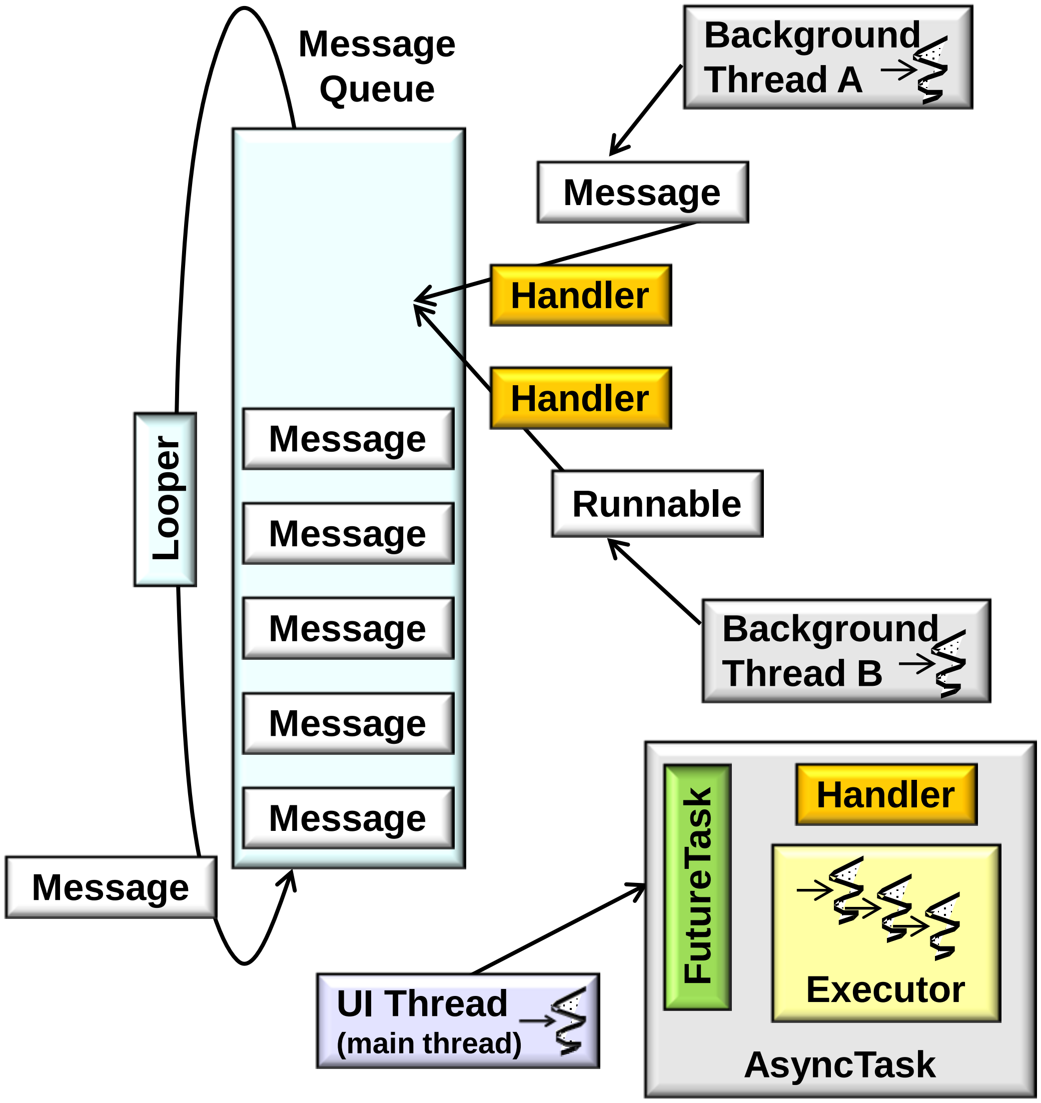
# Overview of Android Concurrency Frameworks

- Android provides several concurrency frameworks
- Android's two primary concurrency frameworks are
- Each frameworks has pros & cons & both are used extensively throughout Android

# Elements of Android Concurrency Frameworks

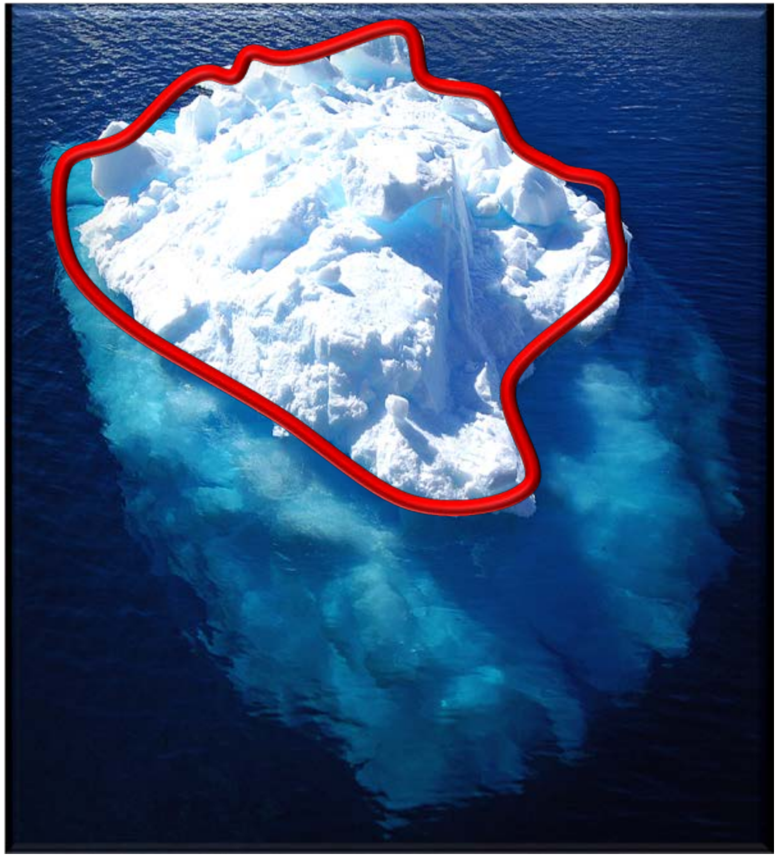
# Elements of Android Concurrency Frameworks

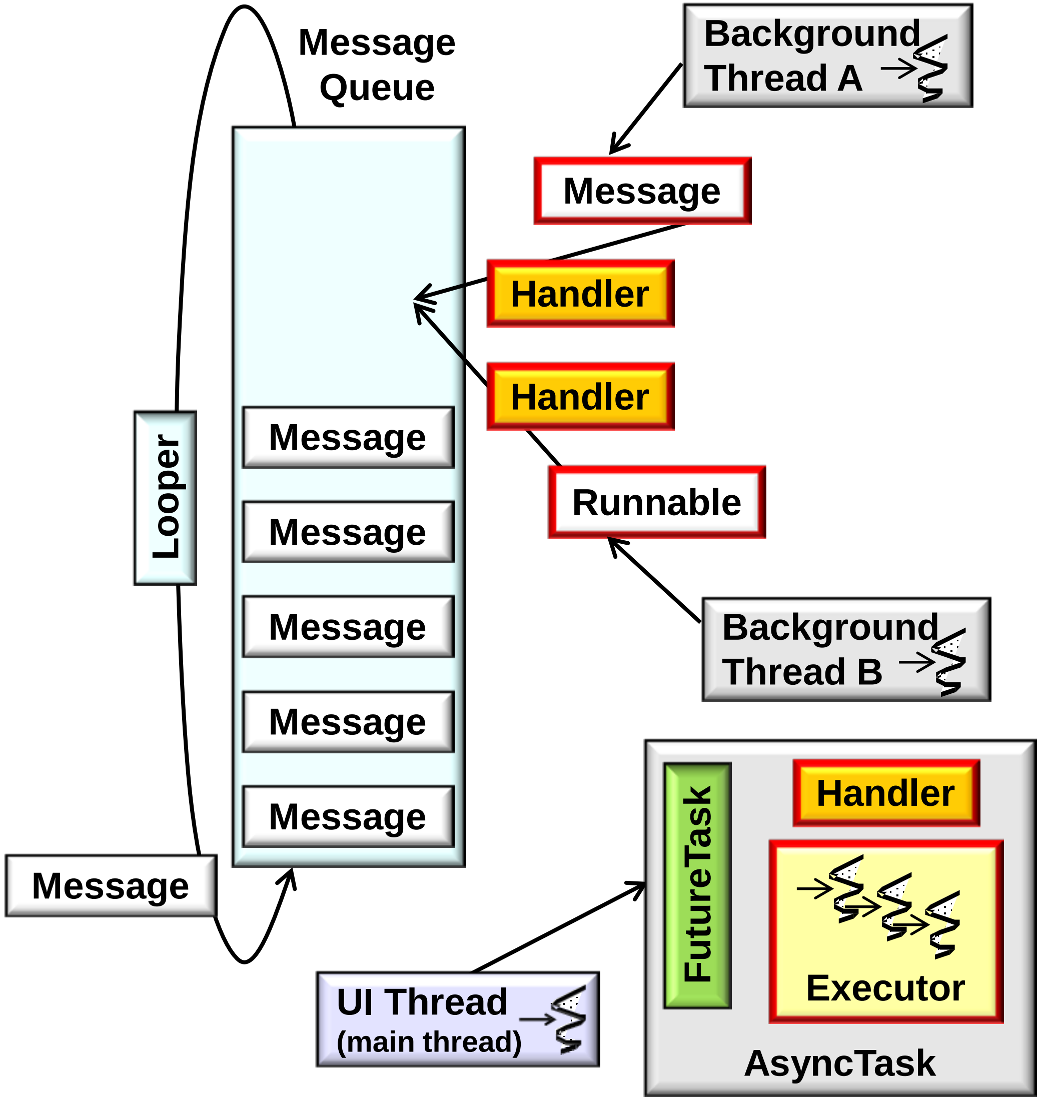
- Android's concurrency frameworks are built using reusable classes



# Elements of Android Concurrency Frameworks

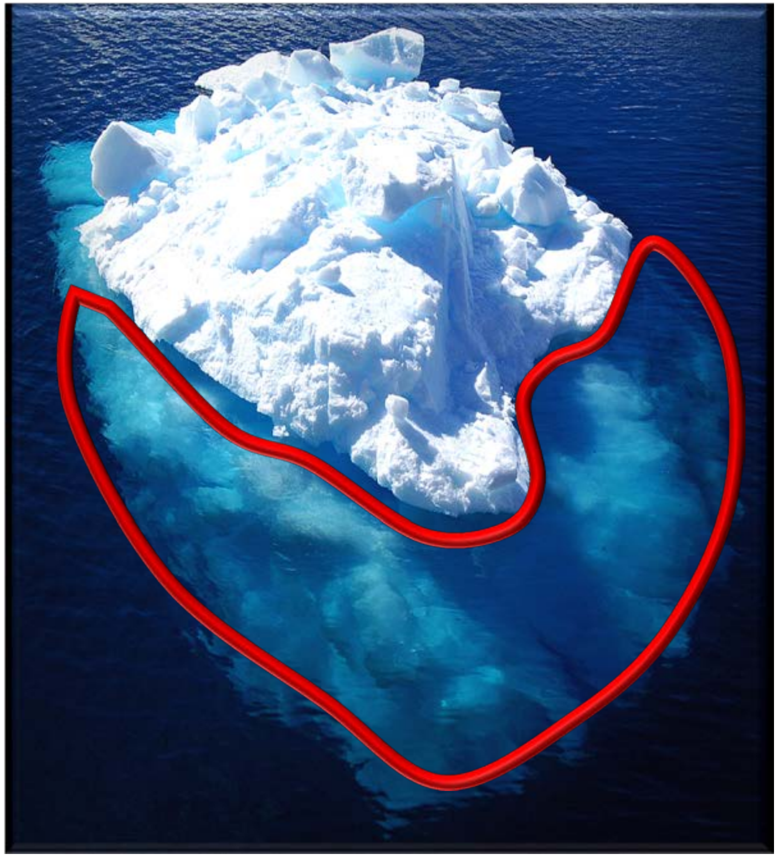
- Android's concurrency frameworks are built using reusable classes

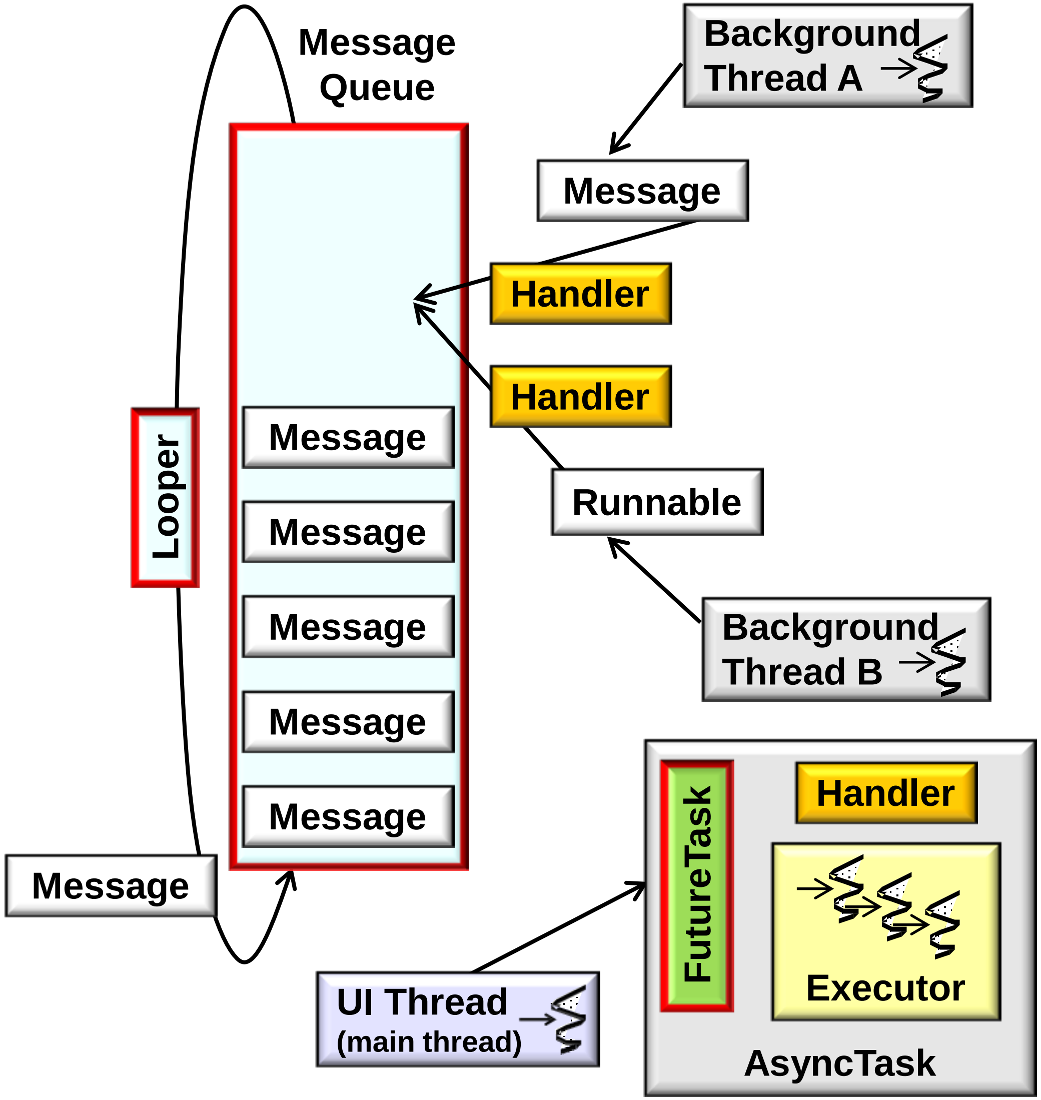




# Elements of Android Concurrency Frameworks

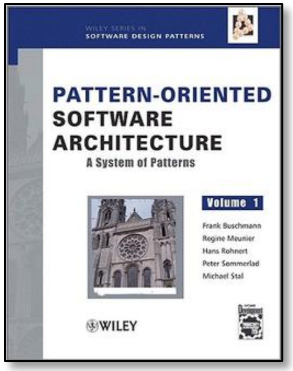
- Android's concurrency frameworks are built using reusable classes





# Elements of Android Concurrency Frameworks

- Android’s concurrency frameworks are built using reusable classes


WILEY SERIES IN SOFTWARE DESIGN PATTERNS

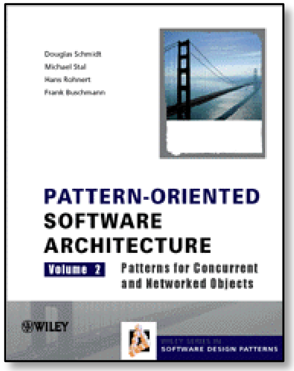
PATTERN-ORIENTED SOFTWARE ARCHITECTURE

A System of Patterns

Volume 1

Frank Buschmann
Regine Meunier
Hans Rohnert
Peter Sommerlad
Michael Stal

WILEY


Douglas Schmidt
Michael Stal
Hans Rohnert
Frank Buschmann

PATTERN-ORIENTED SOFTWARE ARCHITECTURE

Volume 2 Patterns for Concurrent and Networked Objects

WILEY

WILEY SERIES IN SOFTWARE DESIGN PATTERNS

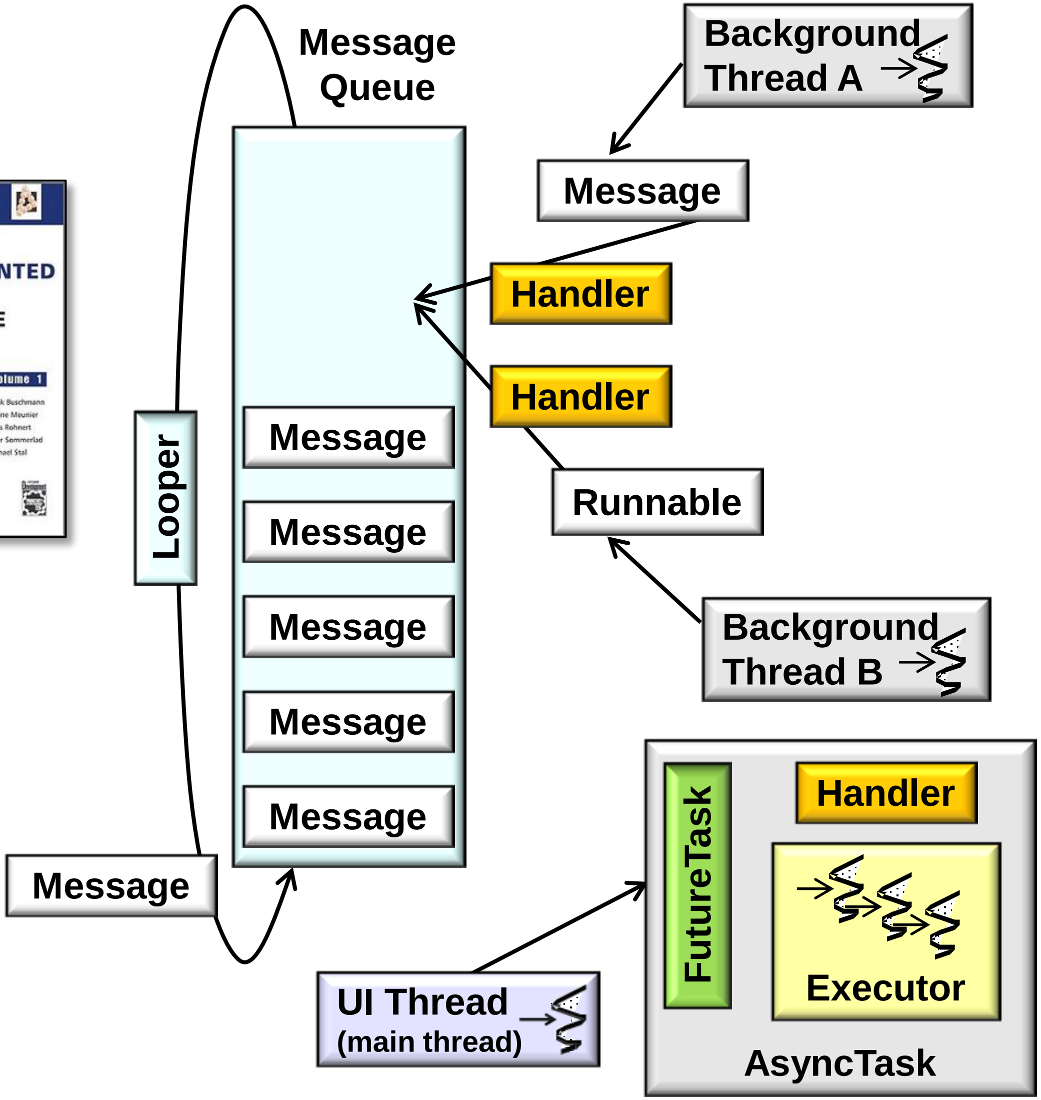
Design Patterns

Elements of Reusable Object-Oriented Software

Erich Gamma
Richard Helm
Ralph Johnson
John Vlissides

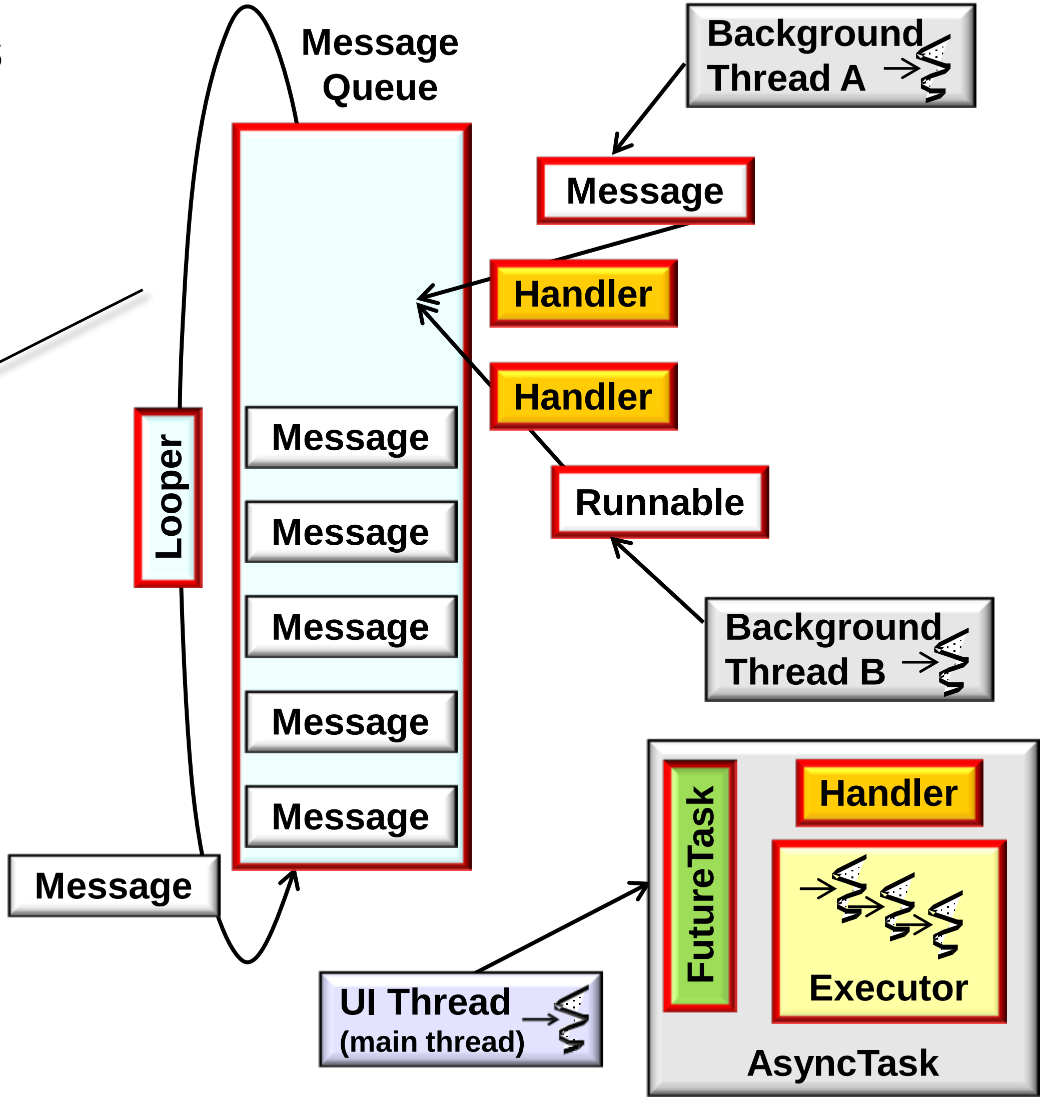
Foreword by Grady Booch

ADDISON-WESLEY PROFESSIONAL COMPUTING SERIES



# Elements of Android Concurrency Frameworks

- Android's concurrency frameworks are built using reusable classes

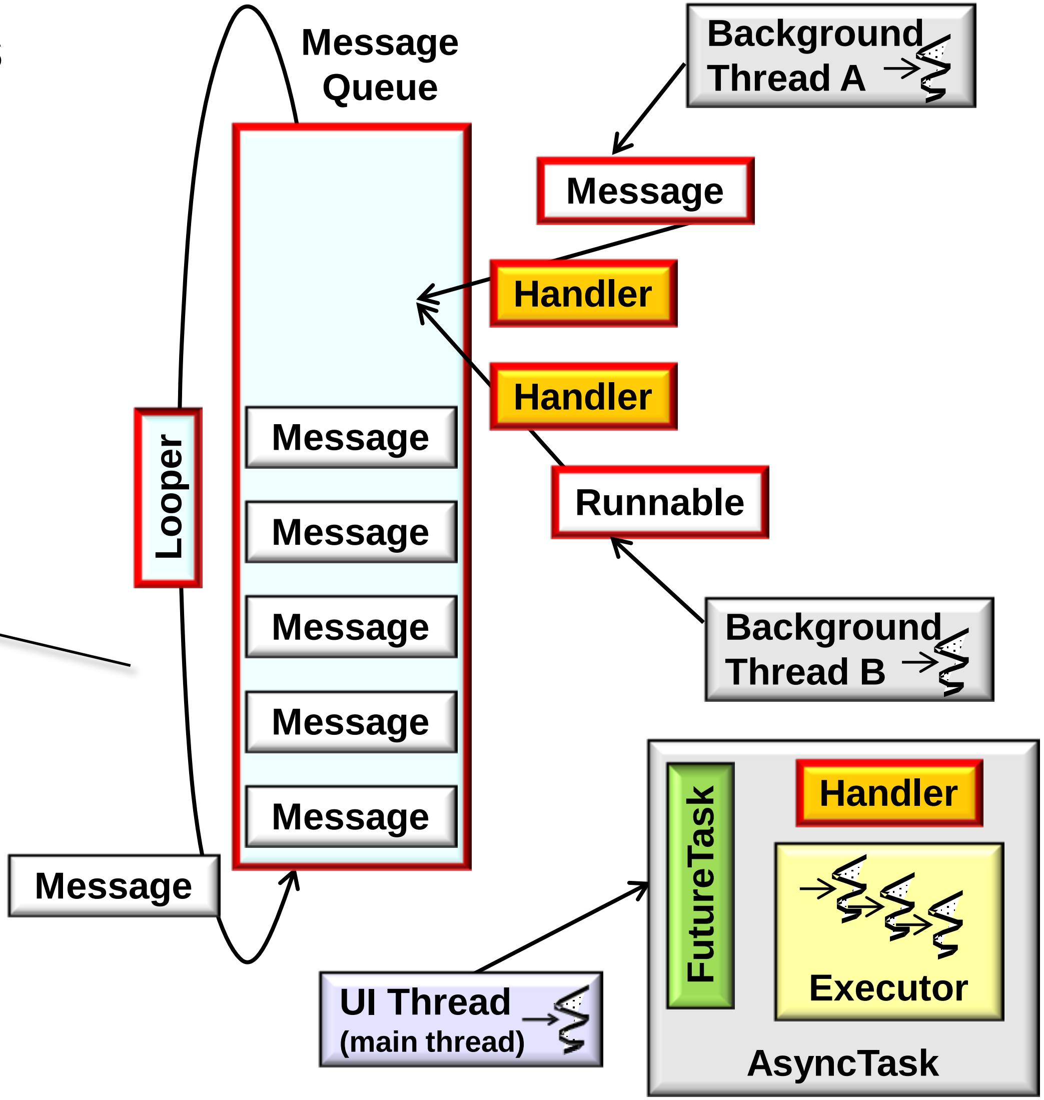


*We cover the most important classes in the Android concurrency frameworks*

# Elements of Android Concurrency Frameworks

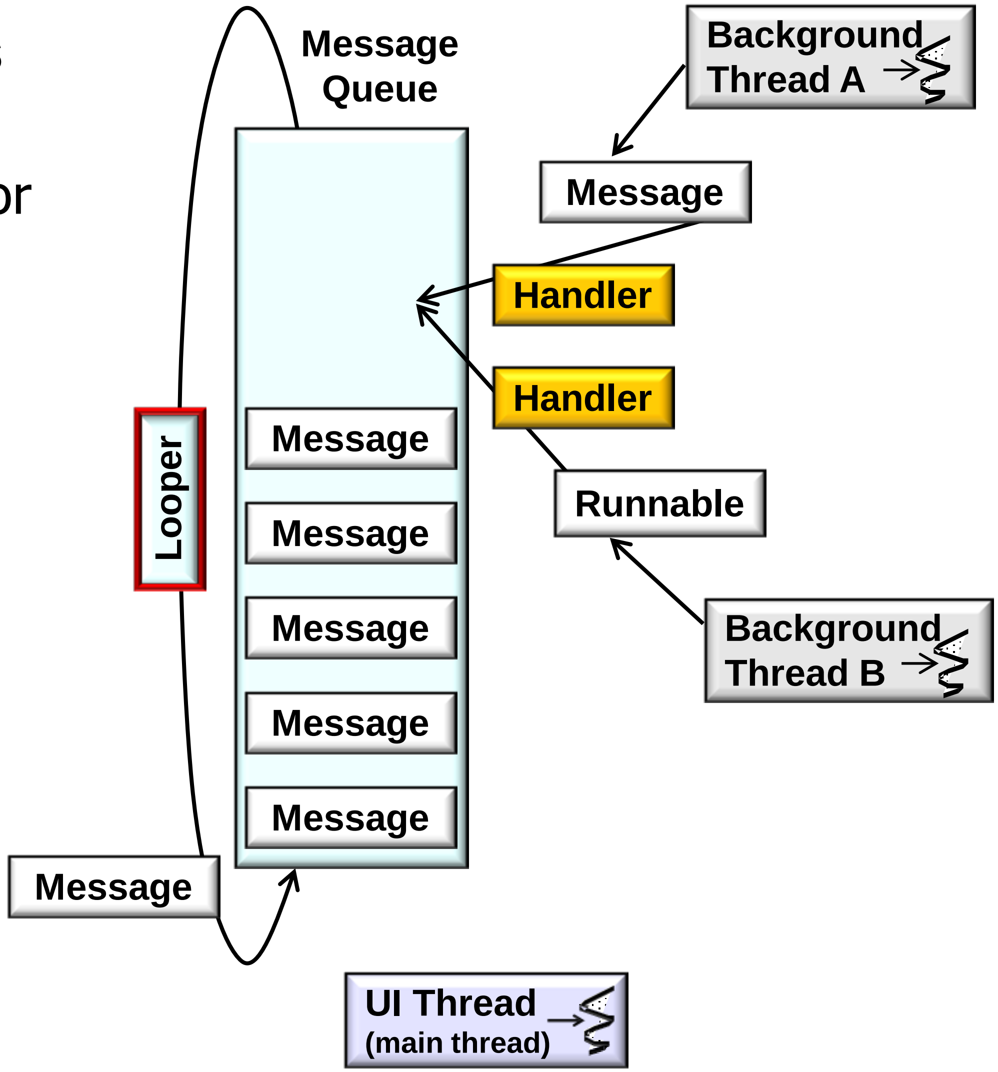
- Android's concurrency frameworks are built using reusable classes

*Some classes are used by both the HaMeR & AsyncTask concurrency frameworks*



# Elements of Android Concurrency Frameworks

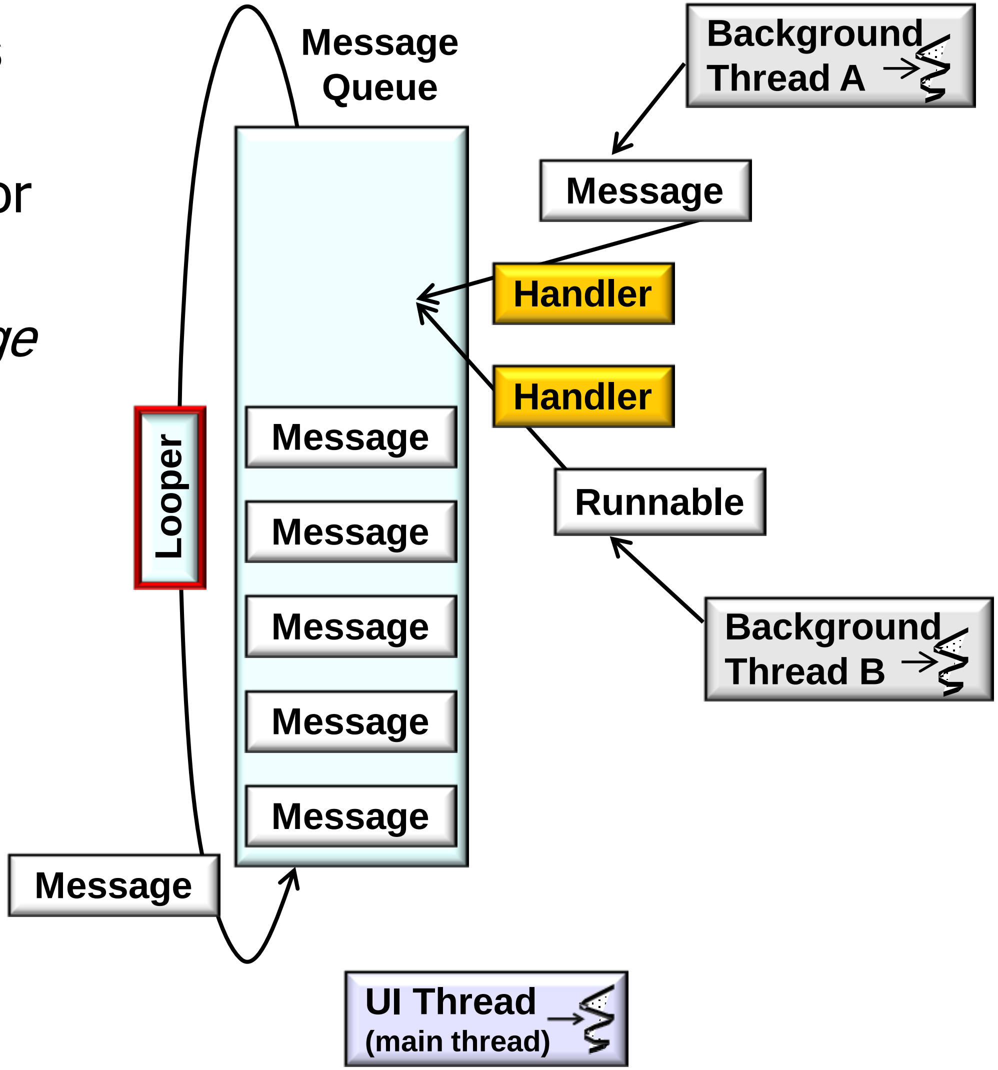
- Android's concurrency frameworks are built using reusable classes
  - Looper – Run a message loop for a thread



developer.android.com/reference/android/os/Looper.html

# Elements of Android Concurrency Frameworks

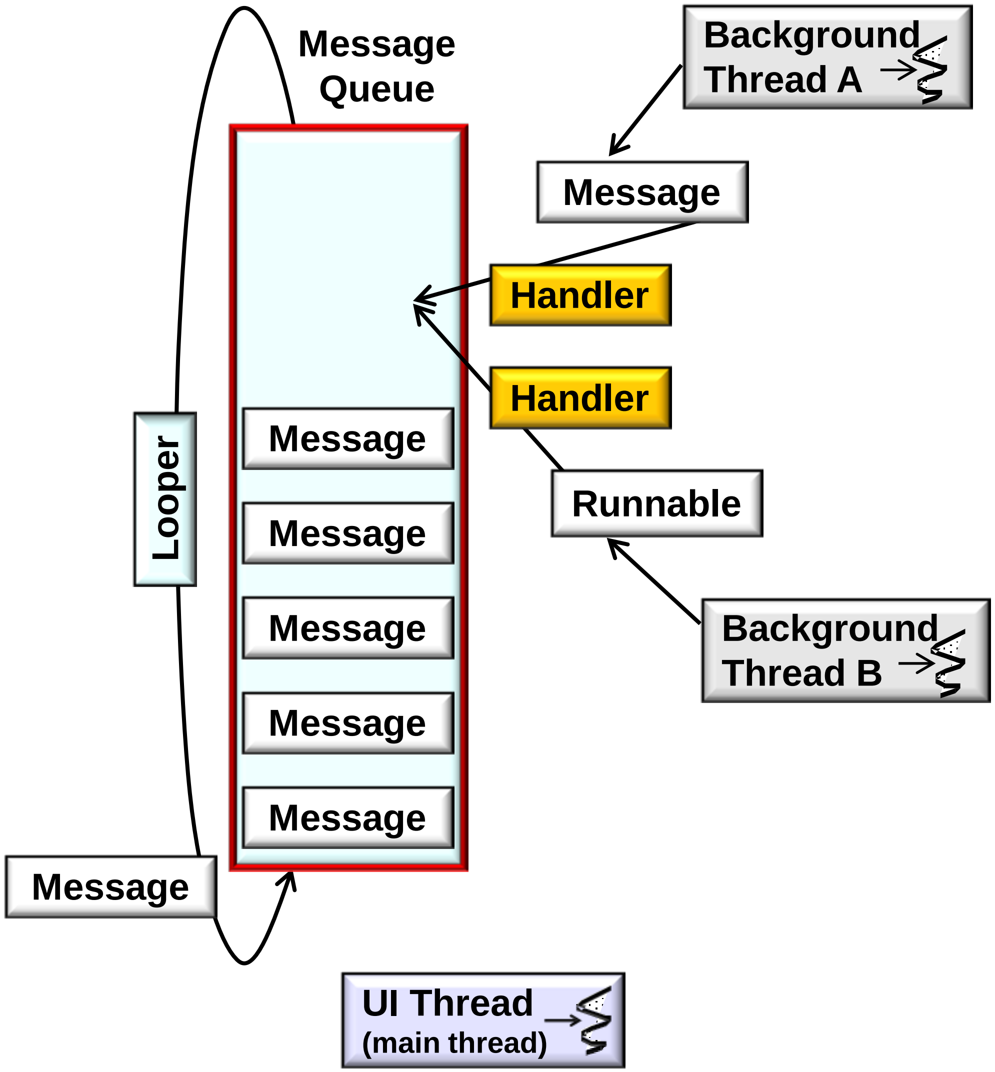
- Android's concurrency frameworks are built using reusable classes
  - Looper – Run a message loop for a thread
    - Applies *Thread-Specific Storage* pattern to ensure only one Looper is allowed per Thread



See [www.dre.vanderbilt.edu/~schmidt/PDF/TSS-pattern.pdf](http://www.dre.vanderbilt.edu/~schmidt/PDF/TSS-pattern.pdf)

# Elements of Android Concurrency Frameworks

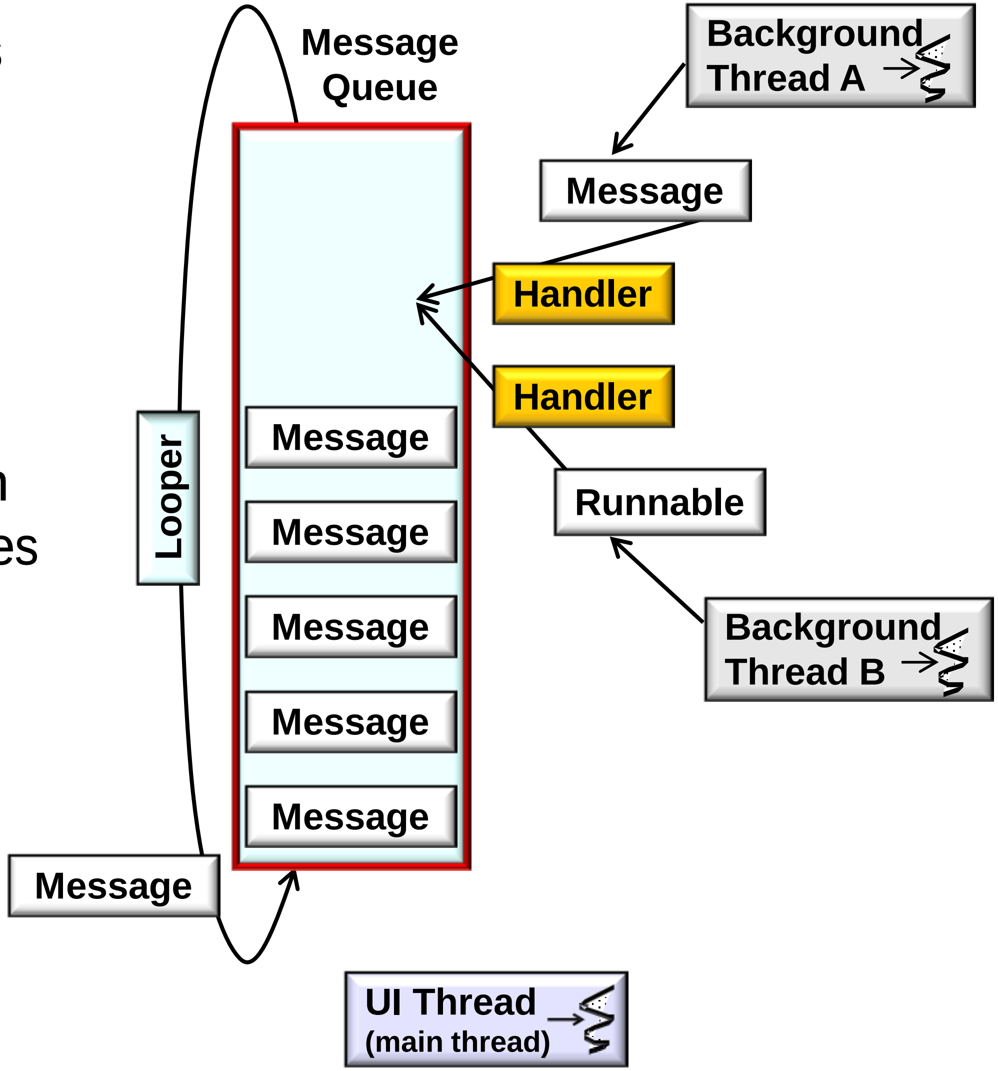
- Android’s concurrency frameworks are built using reusable classes
  - Looper
  - MessageQueue – Holds the list of messages to be dispatched by a Looper



developer.android.com/reference/android/os/MessageQueue.html

# Elements of Android Concurrency Frameworks

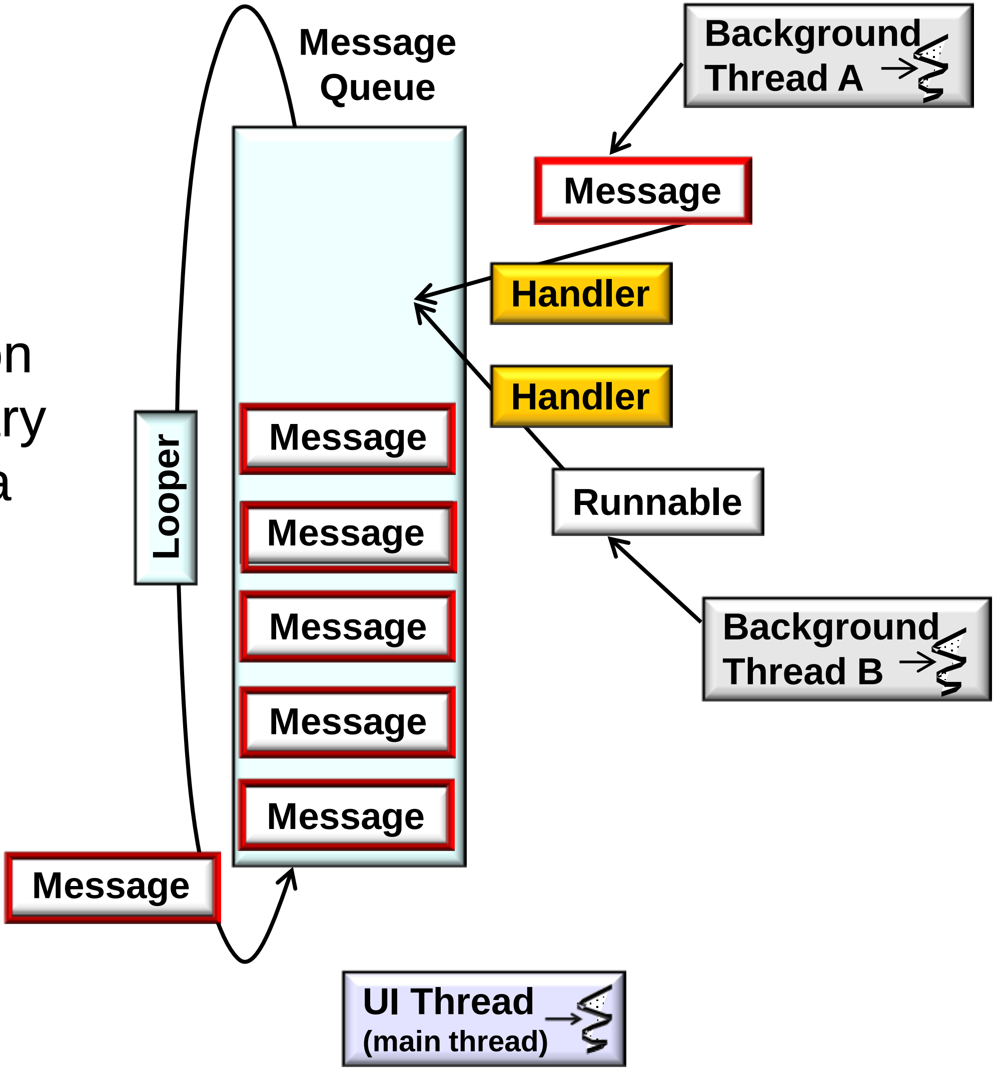
- Android's concurrency frameworks are built using reusable classes
  - Looper
  - MessageQueue – Holds the list of messages to be dispatched by a Looper
    - Applies *Monitor Object* pattern to enqueue /dequeue Messages concurrently & efficiently



See www.dre.vanderbilt.edu/~schmidt/PDF/monitor.pdf

# Elements of Android Concurrency Frameworks

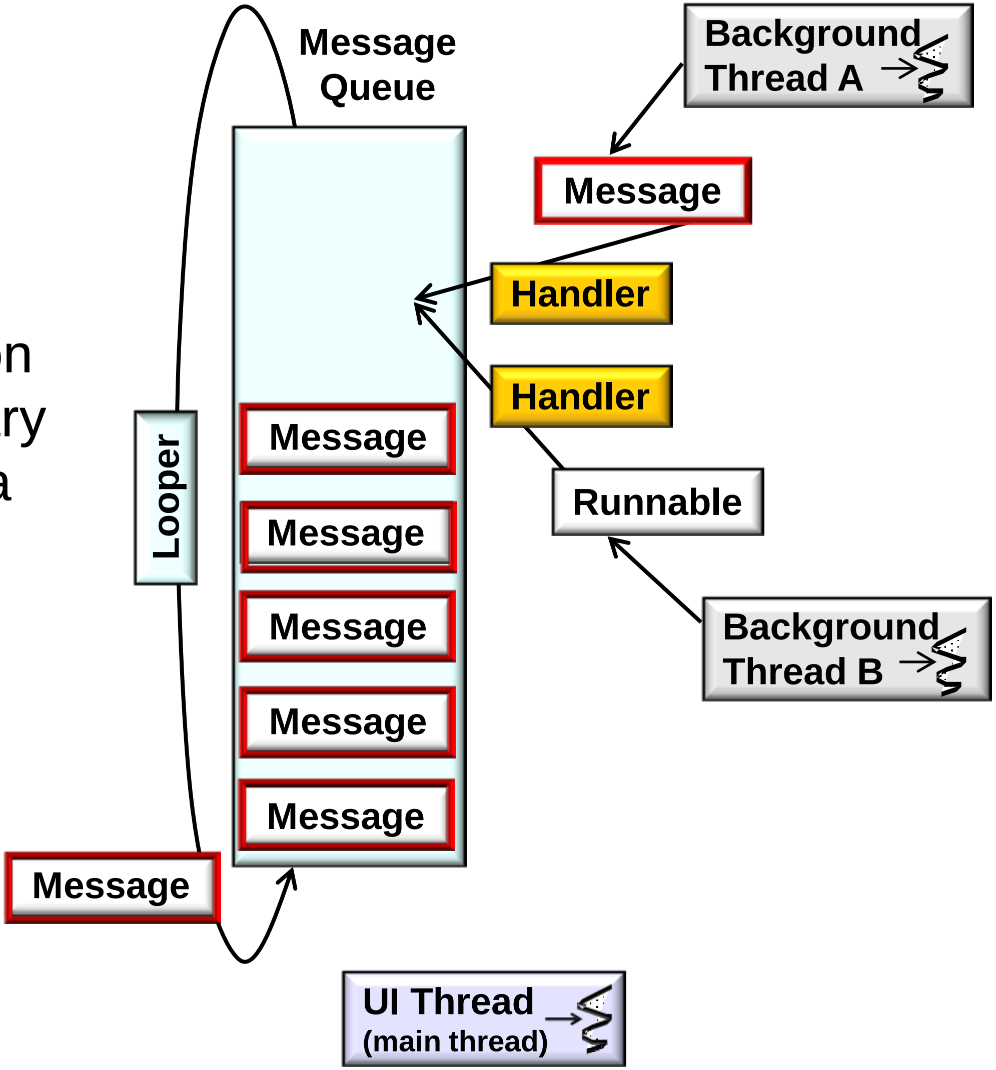
- Android's concurrency frameworks are built using reusable classes
  - Looper
  - MessageQueue
  - Message – Contains a description of a message's type & an arbitrary data object that can be sent to a Handler via a MessageQueue



developer.android.com/reference/android/os/Message.html

# Elements of Android Concurrency Frameworks

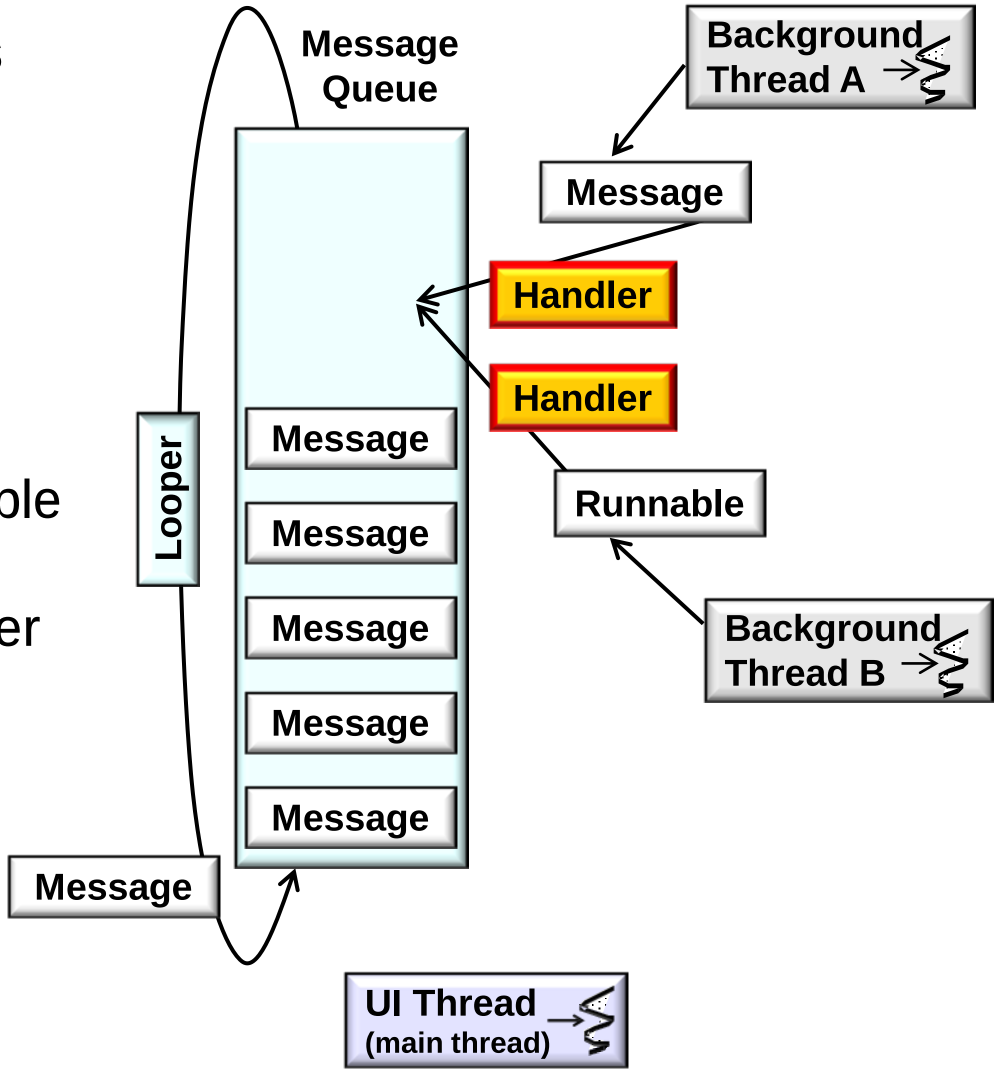
- Android's concurrency frameworks are built using reusable classes
  - Looper
  - MessageQueue
  - Message – Contains a description of a message's type & an arbitrary data object that can be sent to a Handler via a MessageQueue
    - Messages are created via *Factory Method* pattern



See en.wikipedia.org/wiki/Factory_method_pattern

# Elements of Android Concurrency Frameworks

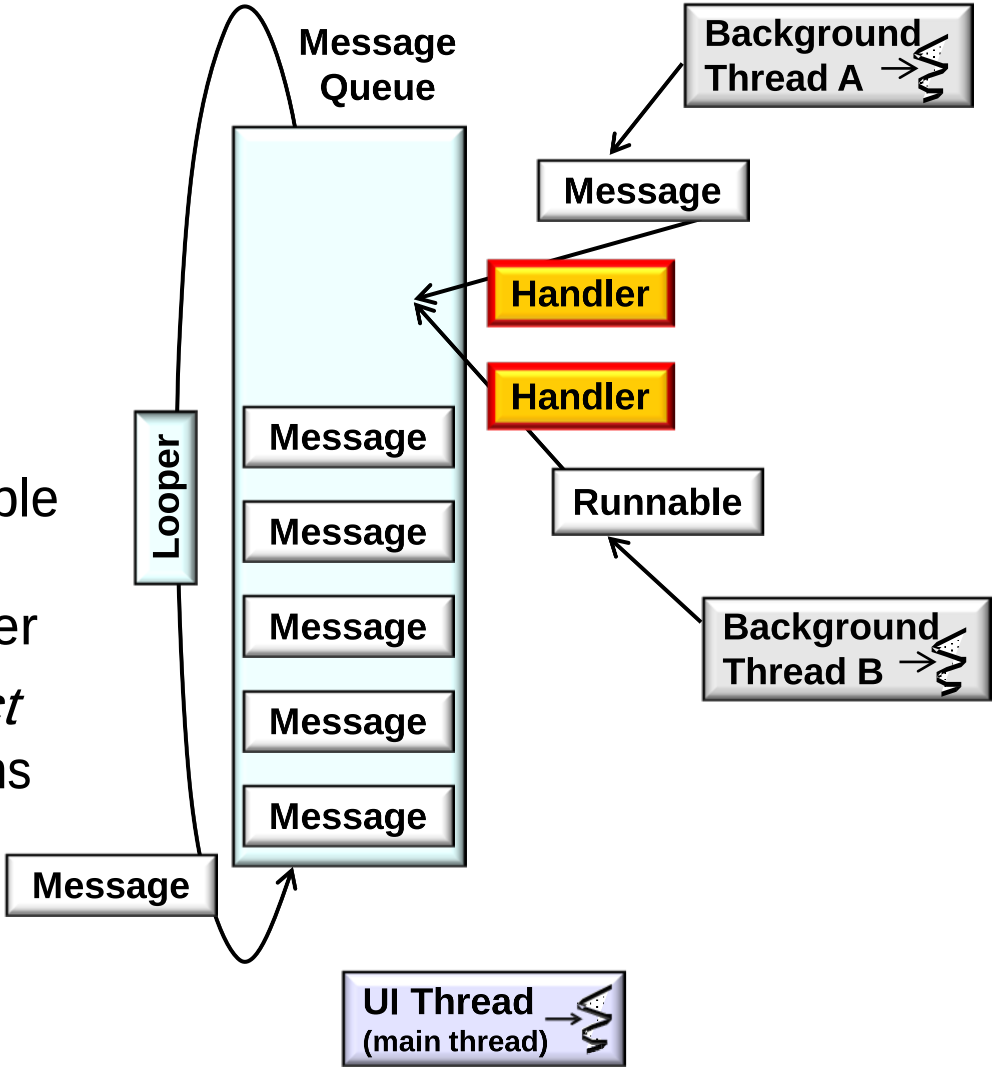
- Android's concurrency frameworks are built using reusable classes
  - Looper
  - MessageQueue
  - Message
  - Handler – Allows the sending & processing of Message & Runnable objects in the MessageQueue associated with a Thread's Looper



developer.android.com/reference/android/os/Handler.html

# Elements of Android Concurrency Frameworks

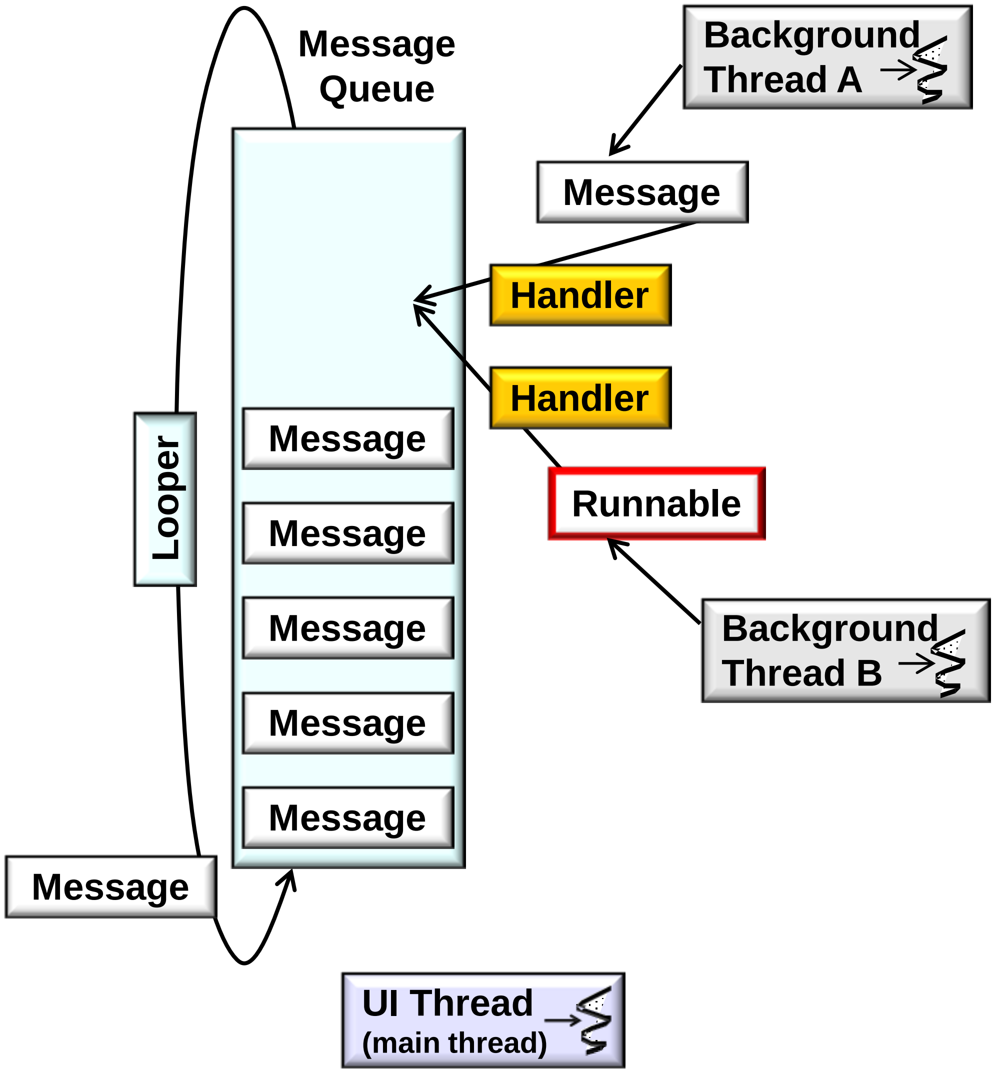
- Android's concurrency frameworks are built using reusable classes
  - Looper
  - MessageQueue
  - Message
  - Handler – Allows the sending & processing of Message & Runnable objects in the MessageQueue associated with a Thread's Looper
    - Handlers support *Active Object* & *Command Processor* patterns to allow sender & receiver Threads to run concurrently



See www.dre.vanderbilt.edu/~schmidt/PDF/{Act-Obj,CommandProcessor}.pdf

# Elements of Android Concurrency Frameworks

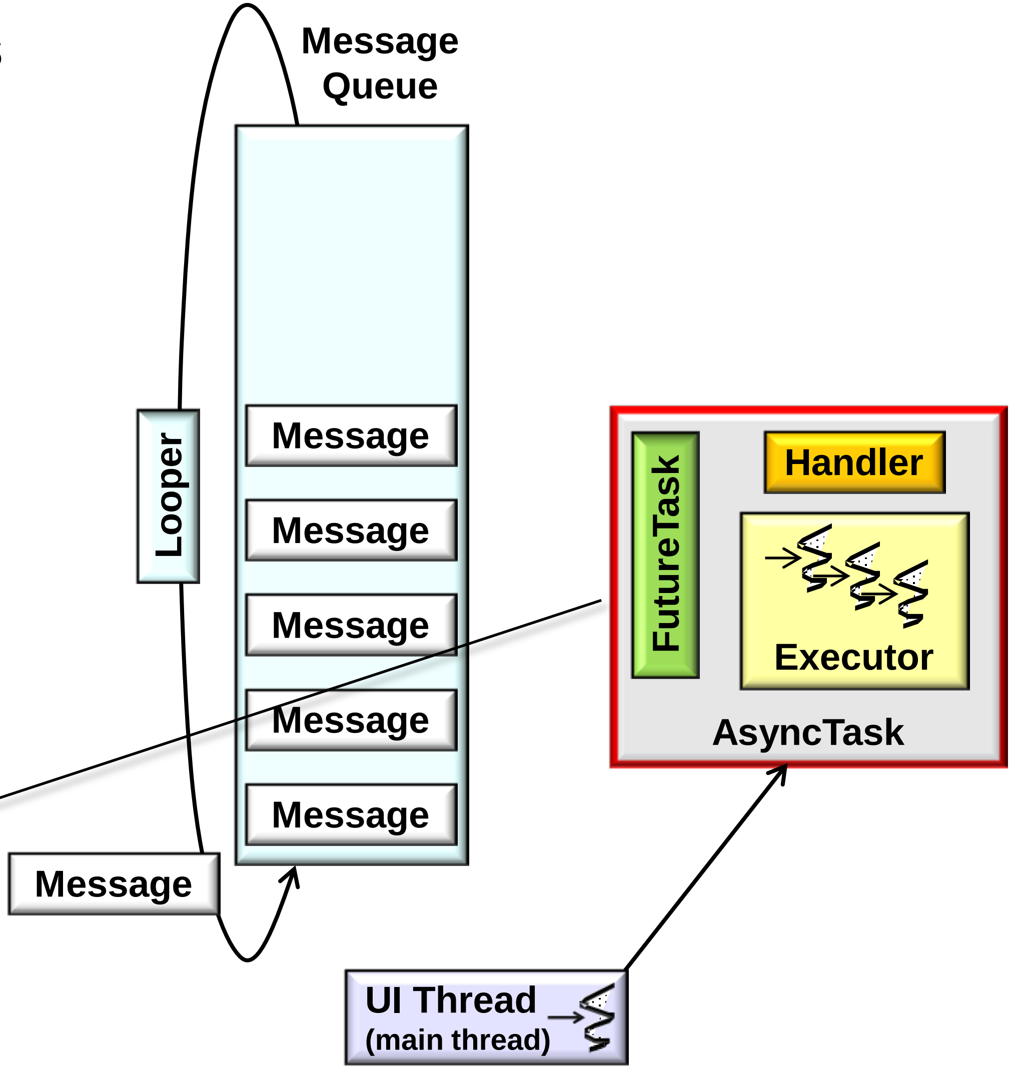
- Android's concurrency frameworks are built using reusable classes
  - Looper
  - MessageQueue
  - Message
  - Handler
  - Runnable – Represents a command that can be executed



developer.android.com/reference/java/lang/Runnable.html

# Elements of Android Concurrency Frameworks

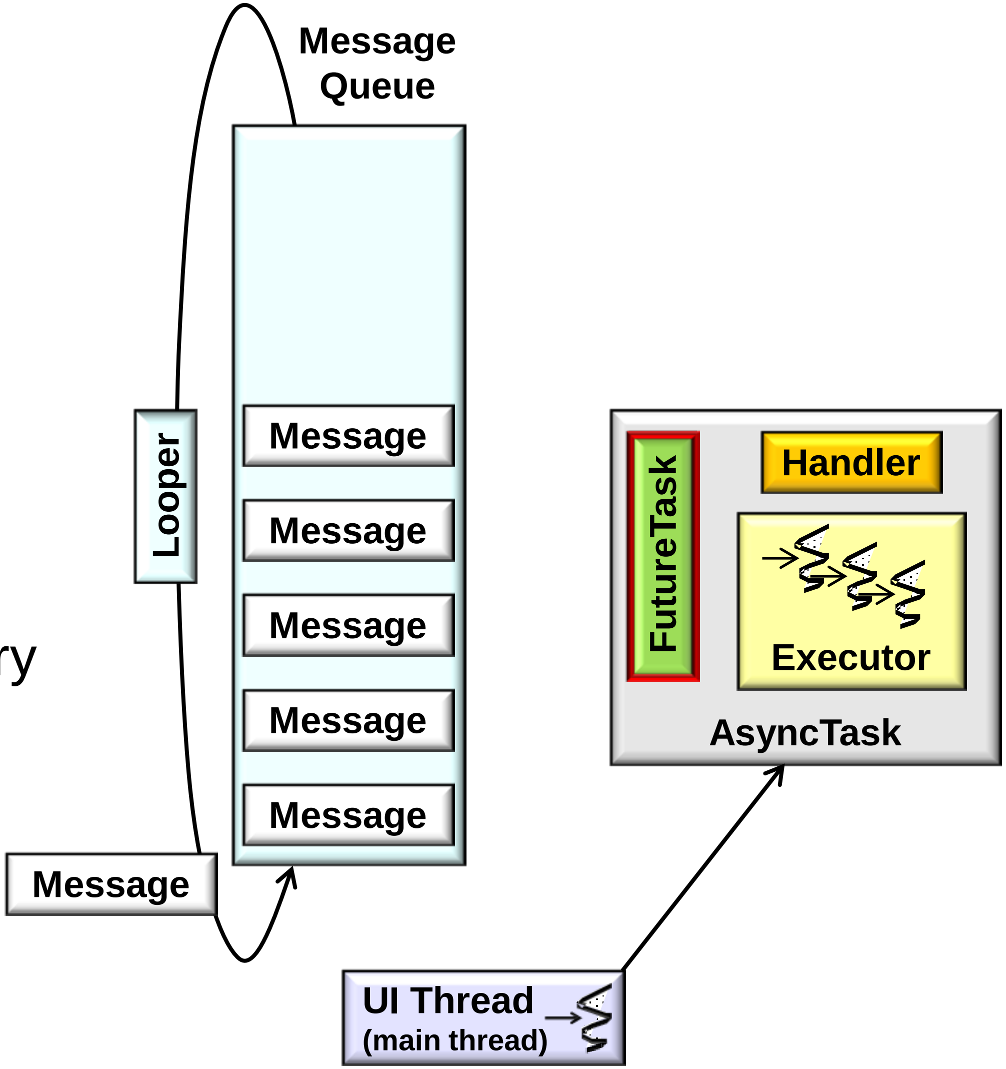
- Android's concurrency frameworks are built using reusable classes
  - Looper
  - MessageQueue
  - Message
  - Handler
  - Runnable



*Some classes are just used by the AsyncTask concurrency framework*

# Elements of Android Concurrency Frameworks

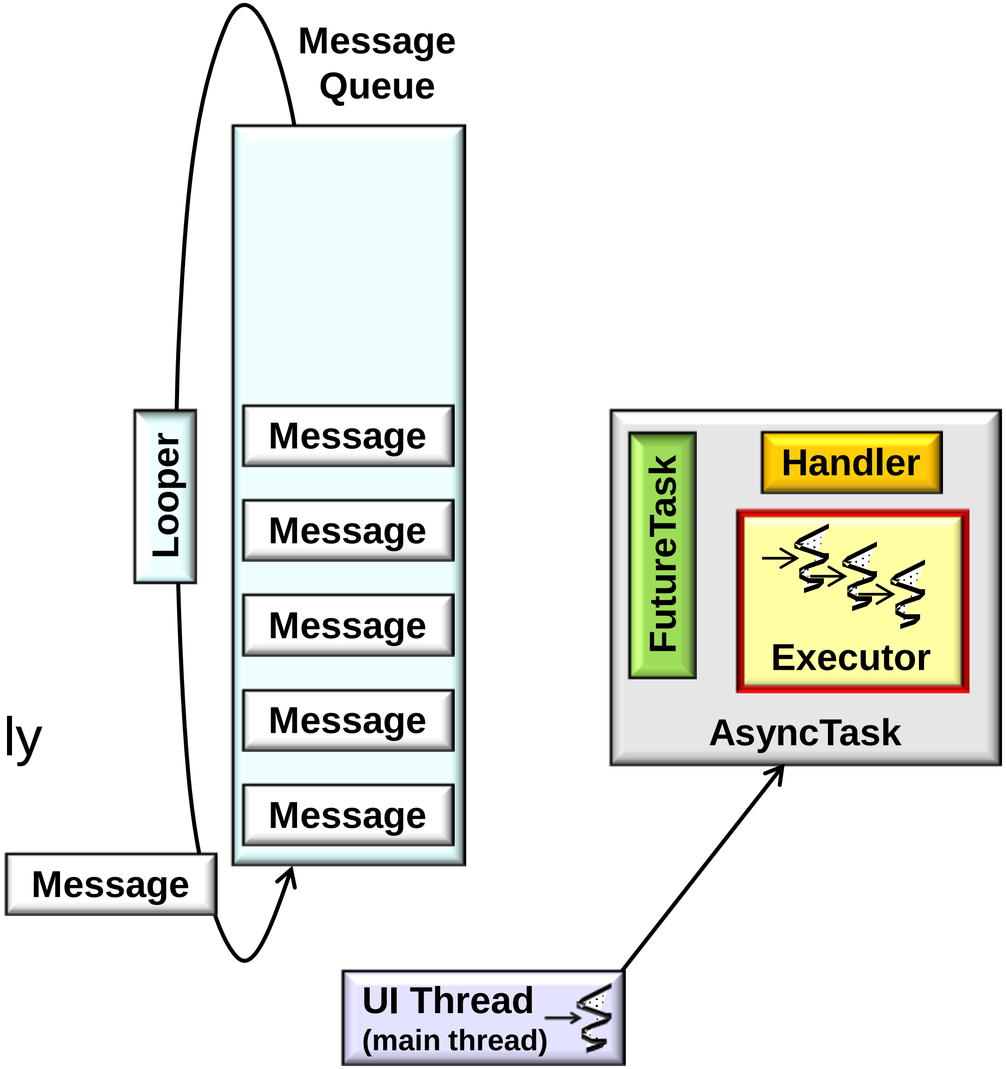
- Android's concurrency frameworks are built using reusable classes
  - Looper
  - MessageQueue
  - Message
  - Handler
  - Runnable
  - FutureTask – Start & cancel an asynchronous computation, query to see if the computation is complete, & retrieve the result of the computation



developer.android.com/reference/java/util/concurrent/FutureTask.html

# Elements of Android Concurrency Frameworks

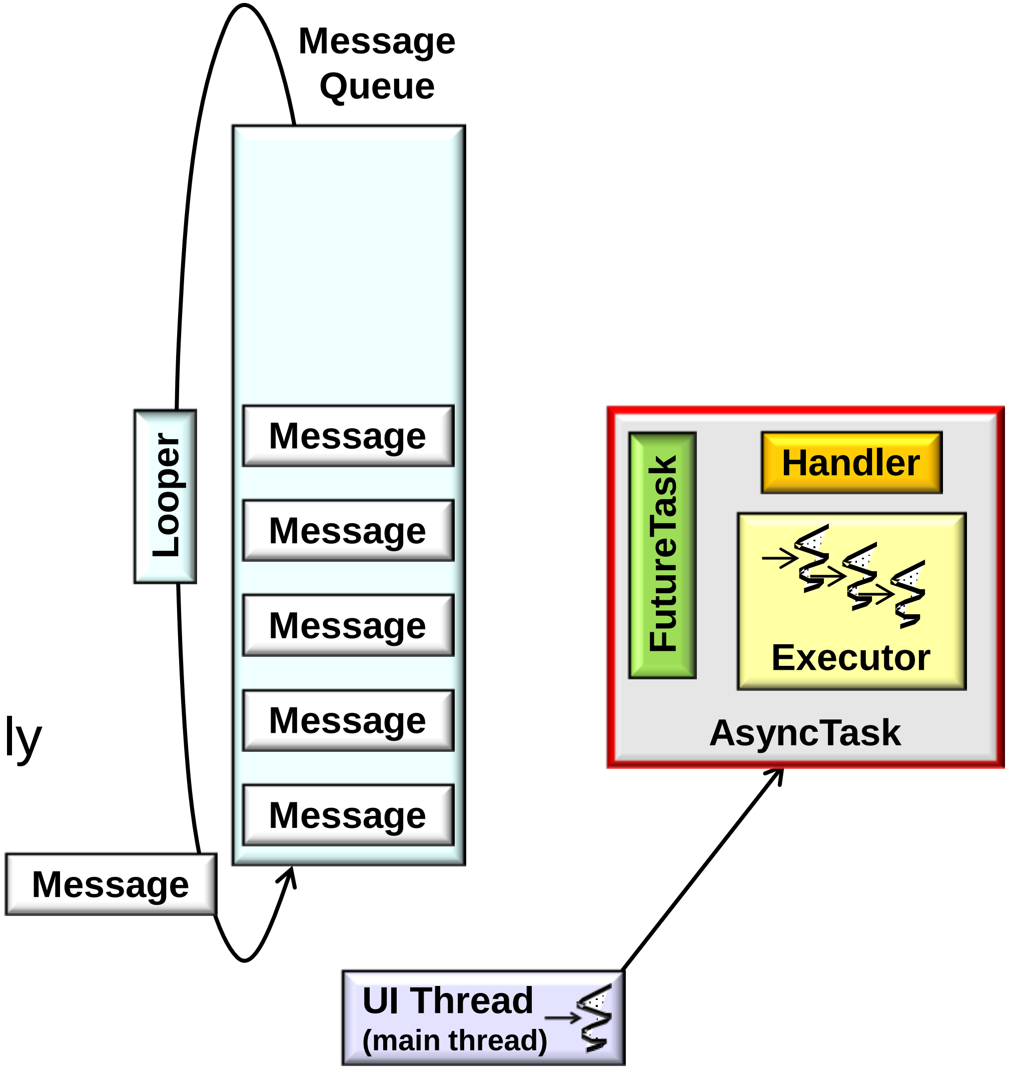
- Android's concurrency frameworks are built using reusable classes
  - Looper
  - MessageQueue
  - Message
  - Handler
  - Runnable
  - FutureTask
  - Executor – Execute submitted Runnable tasks either sequentially or in a pool of threads



developer.android.com/reference/java/util/concurrent/Executor.html

# Elements of Android Concurrency Frameworks

- Android's concurrency frameworks are built using reusable classes
  - Looper
  - MessageQueue
  - Message
  - Handler
  - Runnable
  - FutureTask
  - Executor – Execute submitted Runnable tasks either sequentially or in a pool of threads

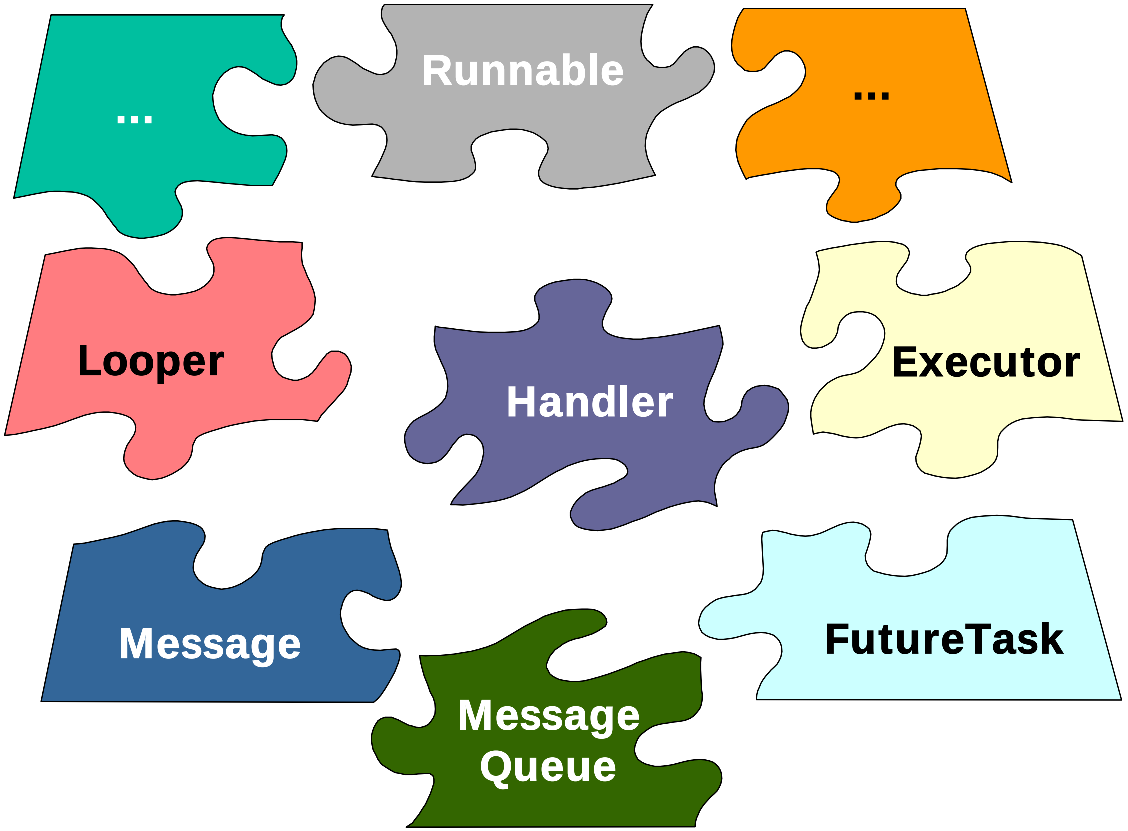


See www.dre.vanderbilt.edu/~schmidt/PDF/HS-HA.pdf

# Mapping Android Concurrency Frameworks to Key Framework Characteristics

# Mapping Android Concurrency Frameworks

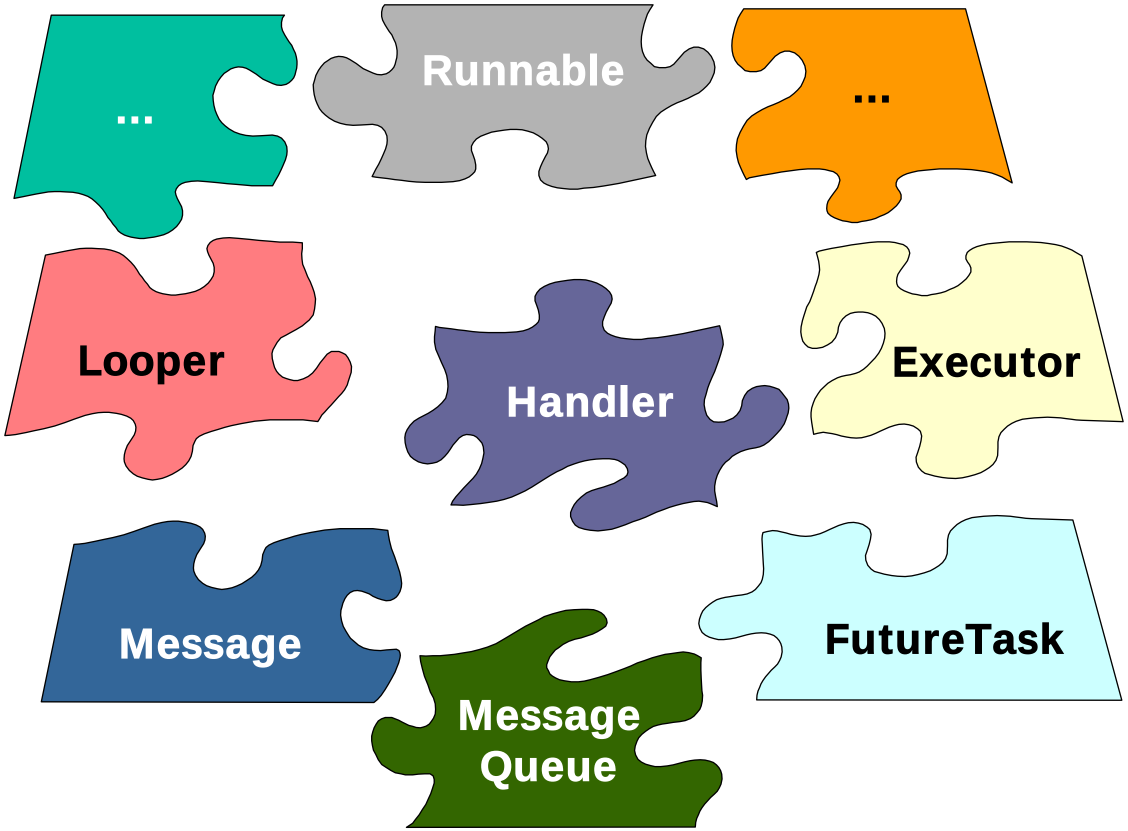
- These classes work together to embody key characteristics of frameworks



See earlier part on "Overview of Patterns & Frameworks (Part 1)"

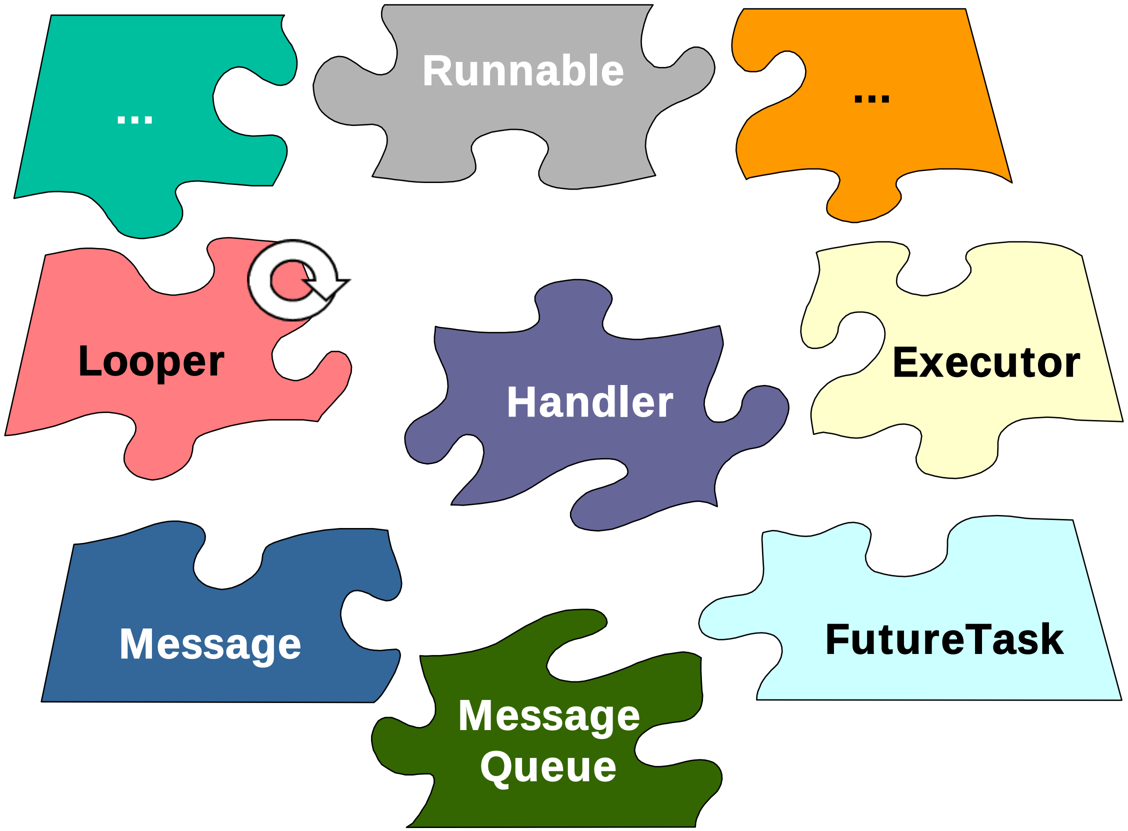
# Mapping Android Concurrency Frameworks

- These classes work together to embody key characteristics of frameworks
  - Inversion of control



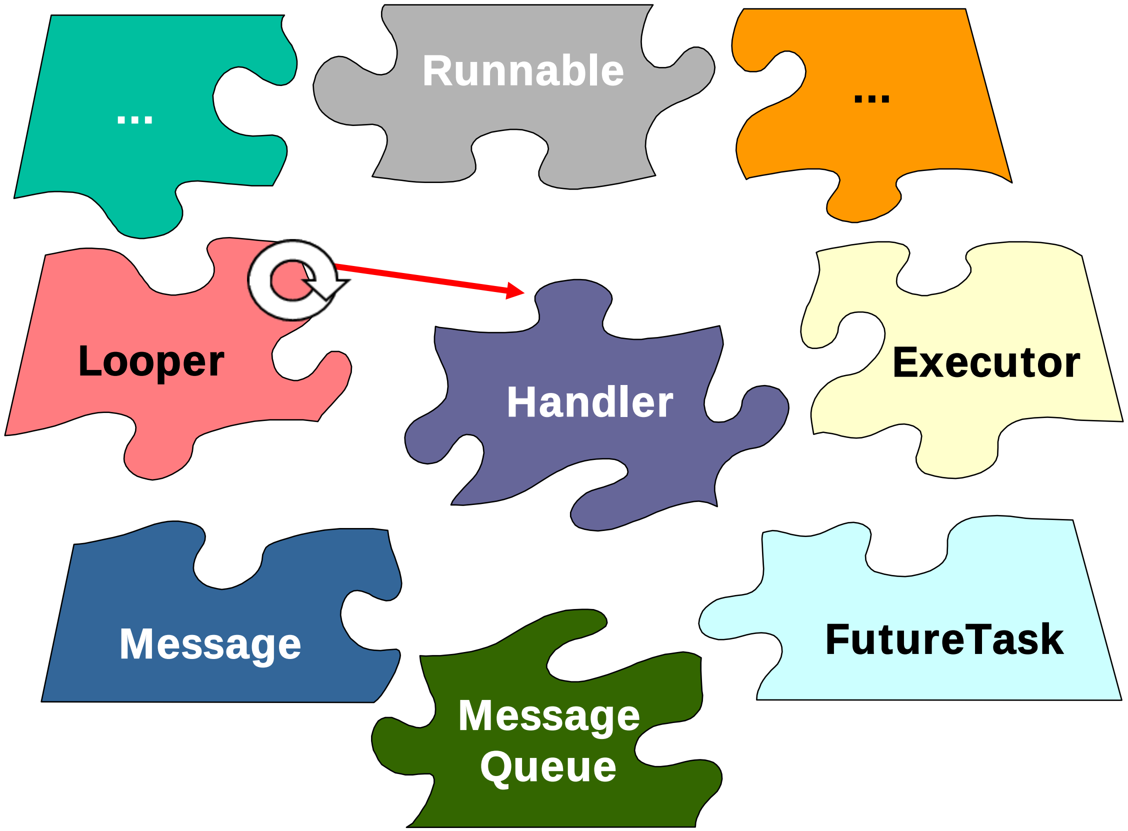
# Mapping Android Concurrency Frameworks

- These classes work together to embody key characteristics of frameworks
  - Inversion of control



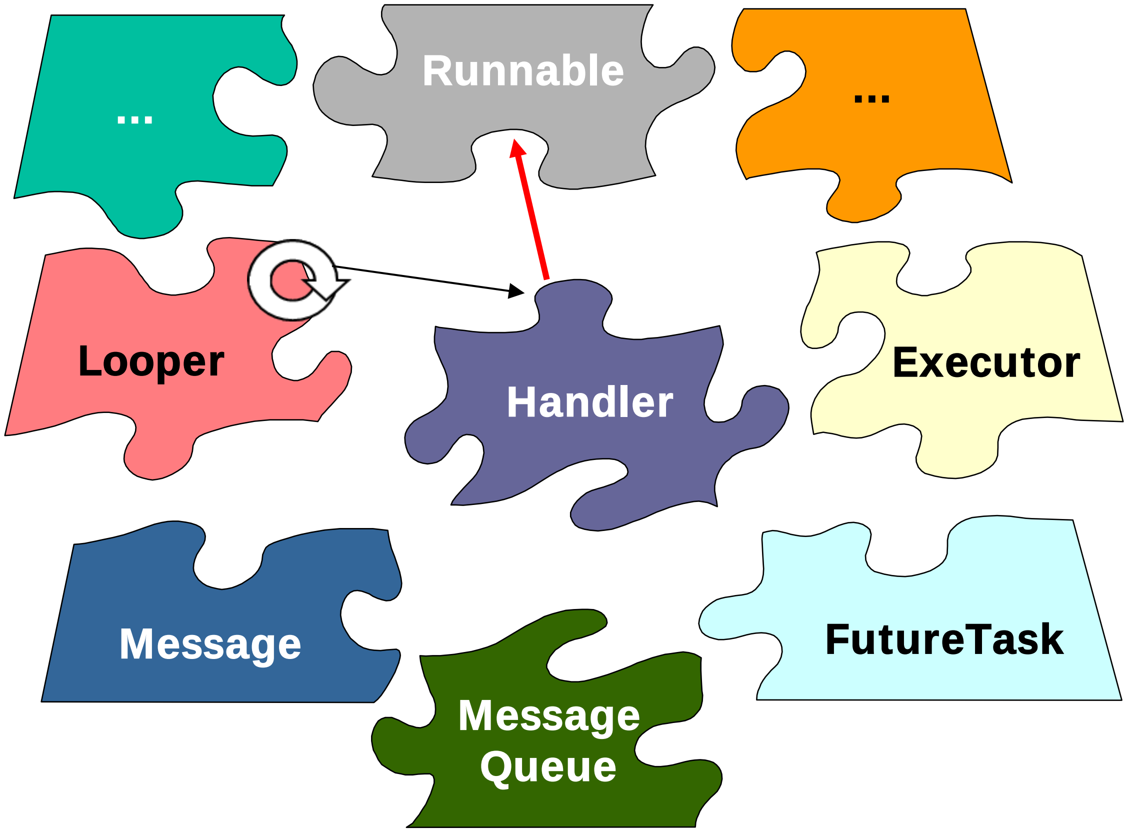
# Mapping Android Concurrency Frameworks

- These classes work together to embody key characteristics of frameworks
  - Inversion of control



# Mapping Android Concurrency Frameworks

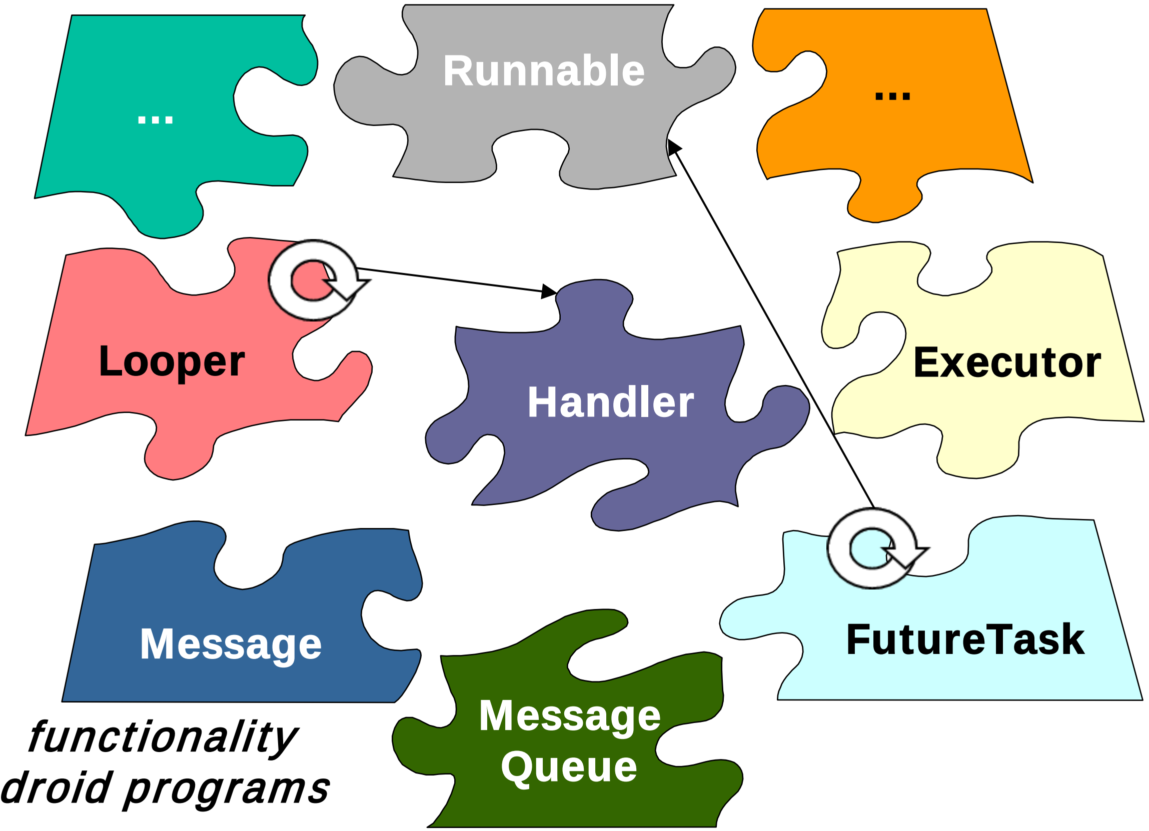
- These classes work together to embody key characteristics of frameworks
  - Inversion of control



# Mapping Android Concurrency Frameworks

- These classes work together to embody key characteristics of frameworks
  - Inversion of control
  - Domain-specific structure & functionality

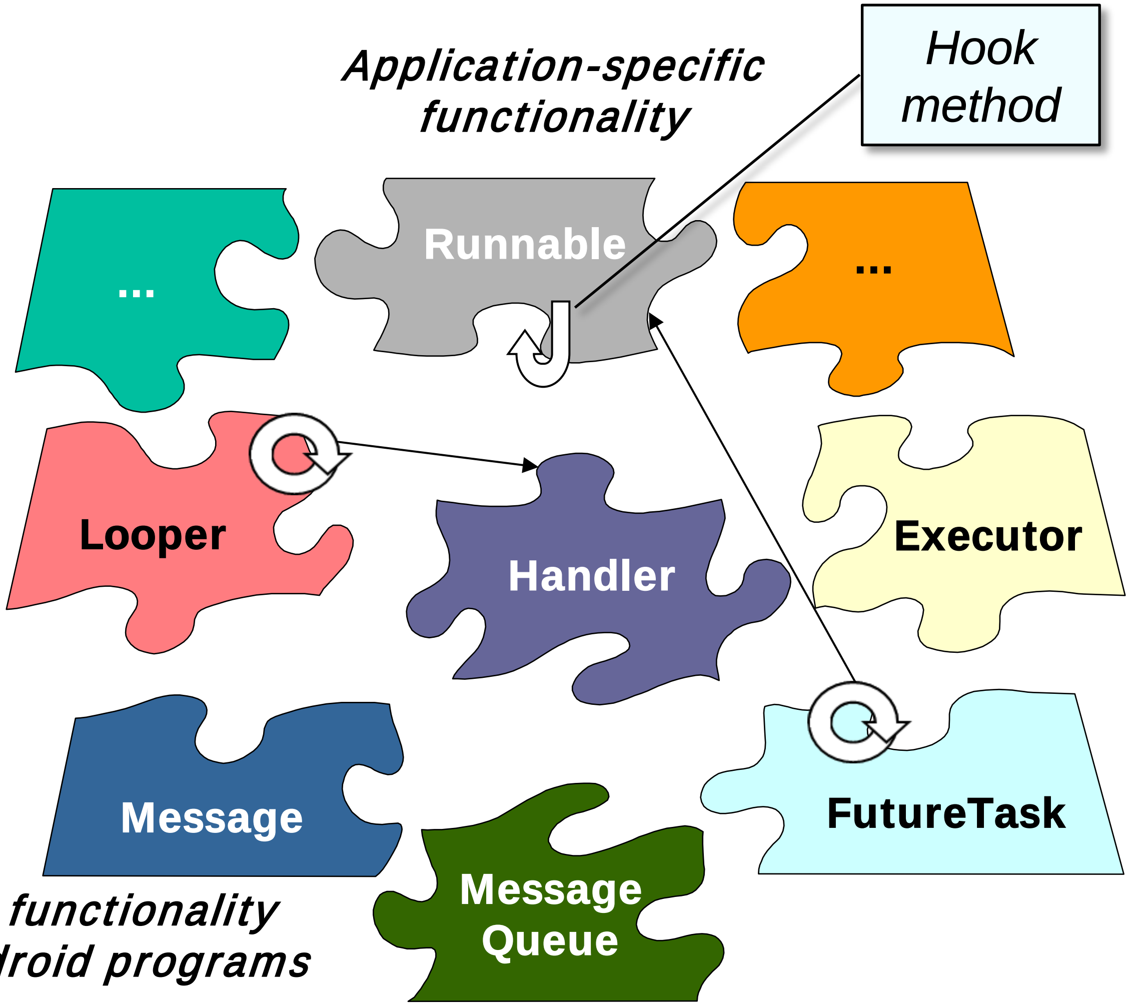
*Application-specific functionality*



*Domain-specific functionality for concurrent Android programs*

# Mapping Android Concurrency Frameworks

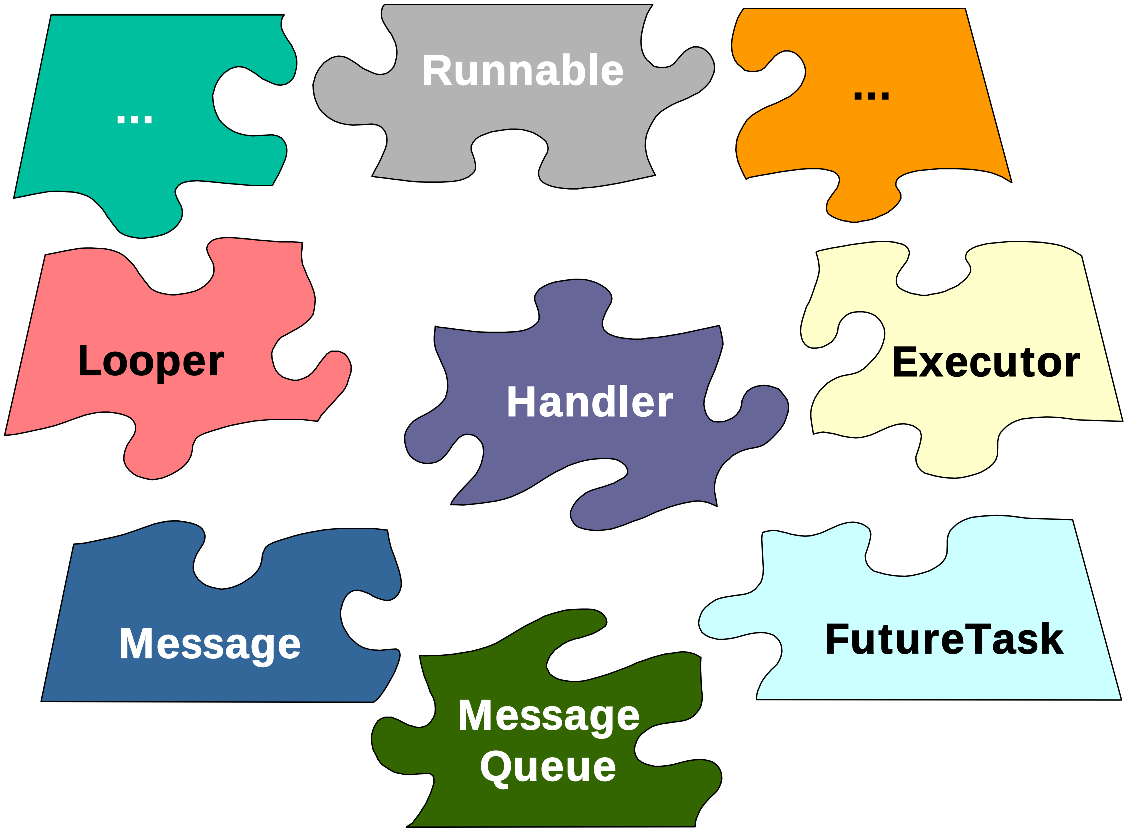
- These classes work together to embody key characteristics of frameworks
  - Inversion of control
  - Domain-specific structure & functionality
  - Semi-complete (portions of) applications



*Domain-specific functionality for concurrent Android programs*

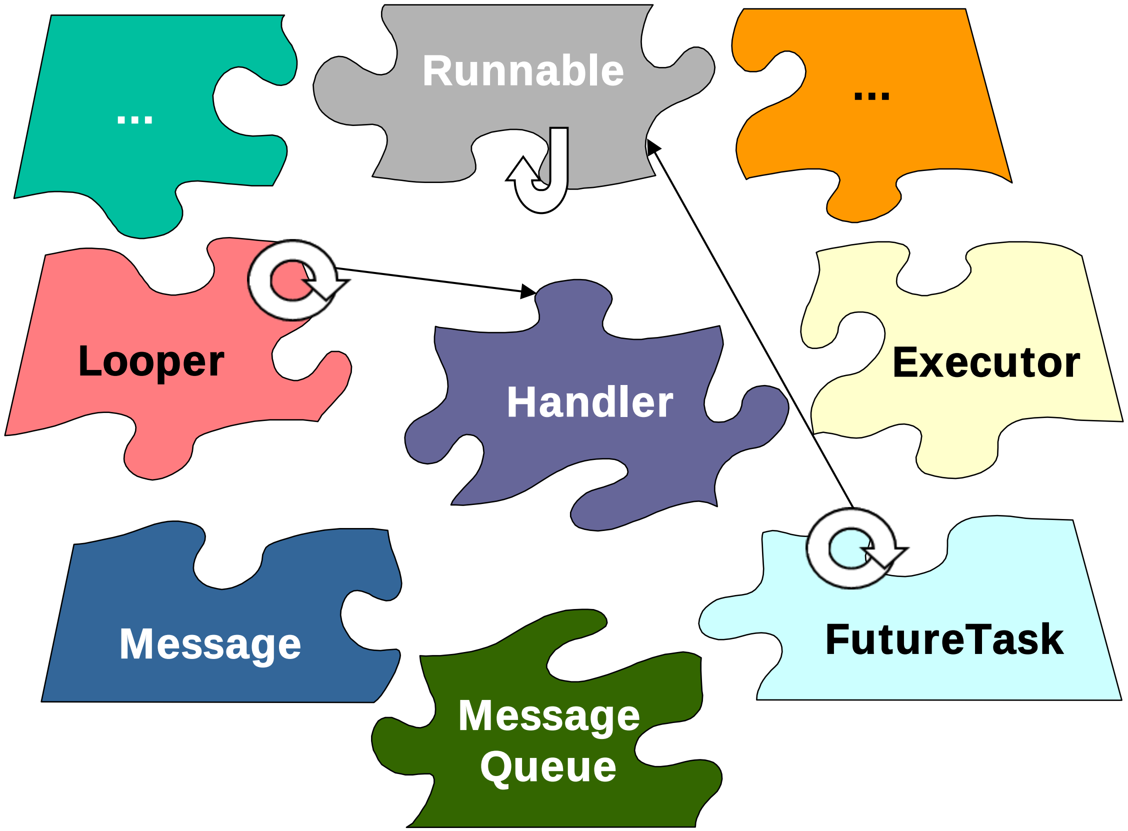
# Mapping Android Concurrency Frameworks

- These classes work together to embody key characteristics of frameworks
- We'll analyze all these classes throughout this Module



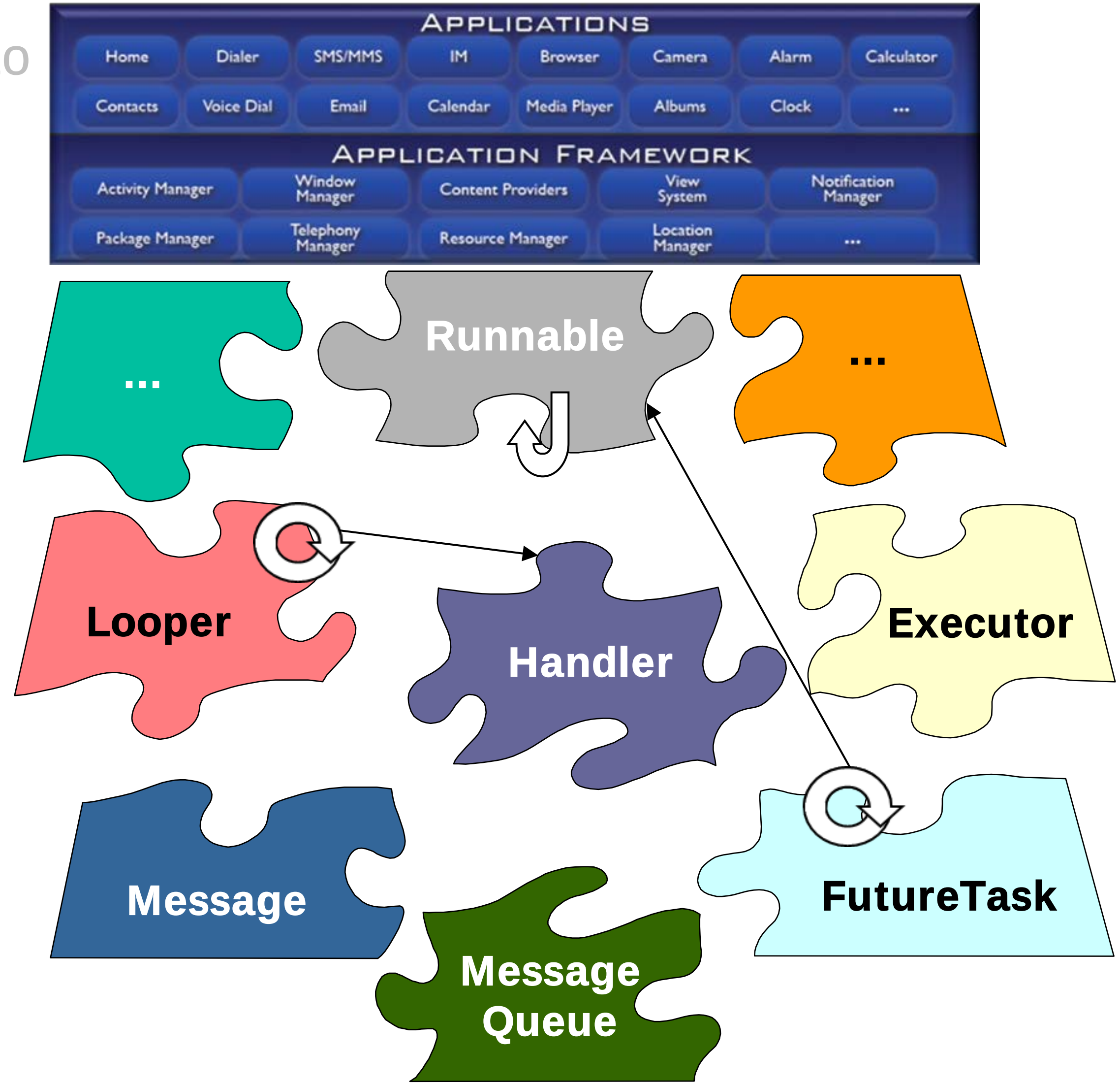
# Mapping Android Concurrency Frameworks

- These classes work together to embody key characteristics of frameworks
- We'll analyze all these classes throughout this Module



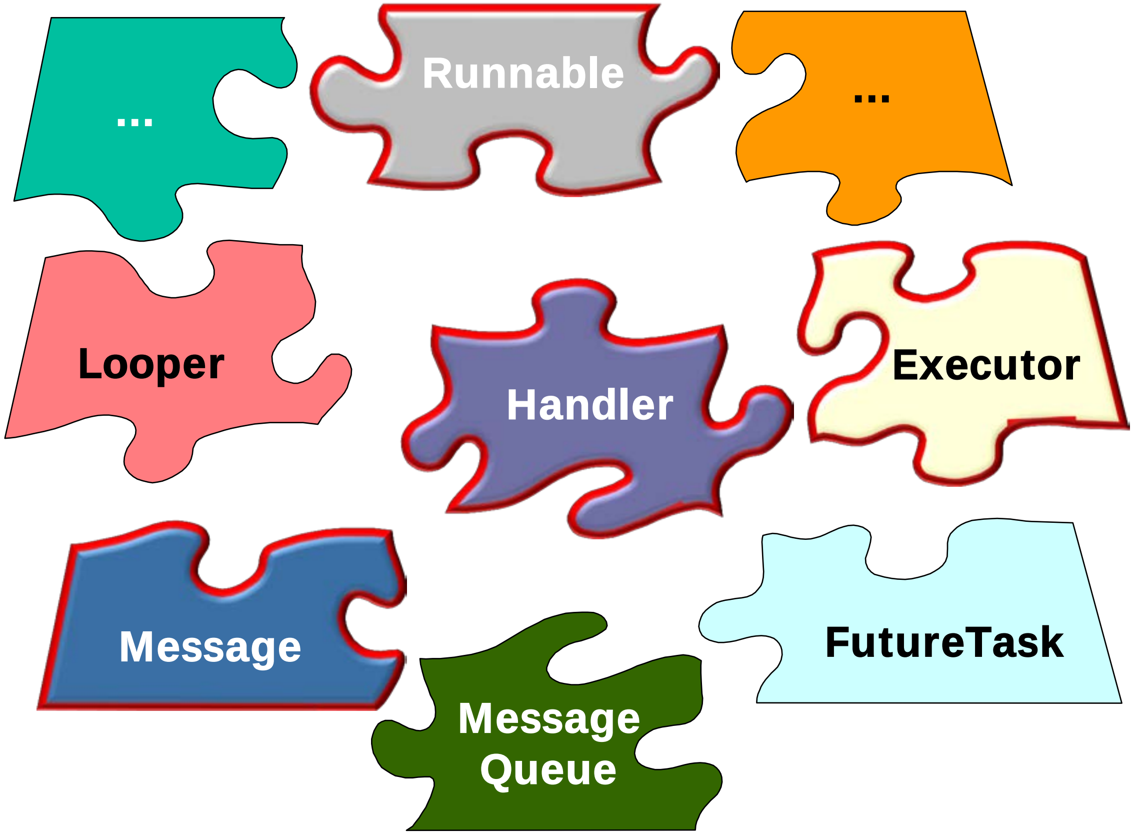
# Mapping Android Concurrency Frameworks

- These classes work together to embody key characteristics of frameworks
- We'll analyze all these classes throughout this Module



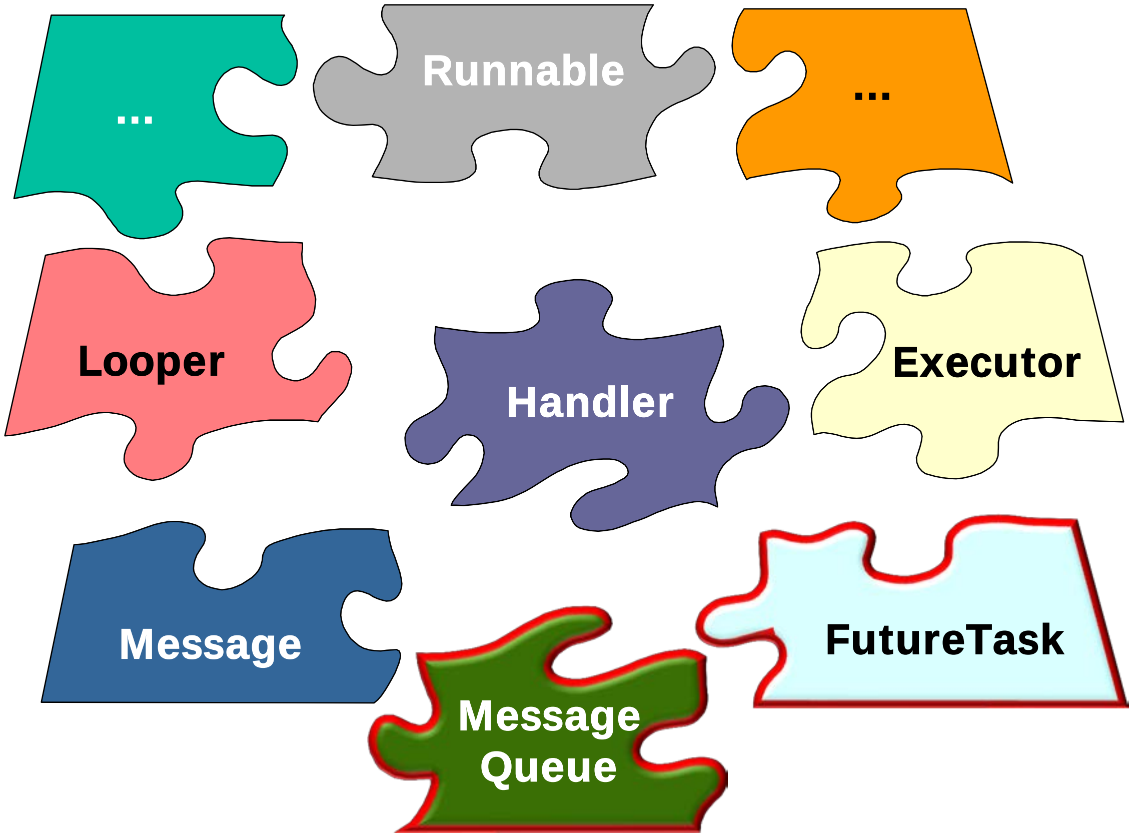
# Mapping Android Concurrency Frameworks

- These classes work together to embody key characteristics of frameworks
- We'll analyze all these classes throughout this Module
  - Interface classes visible to application developers



# Mapping Android Concurrency Frameworks

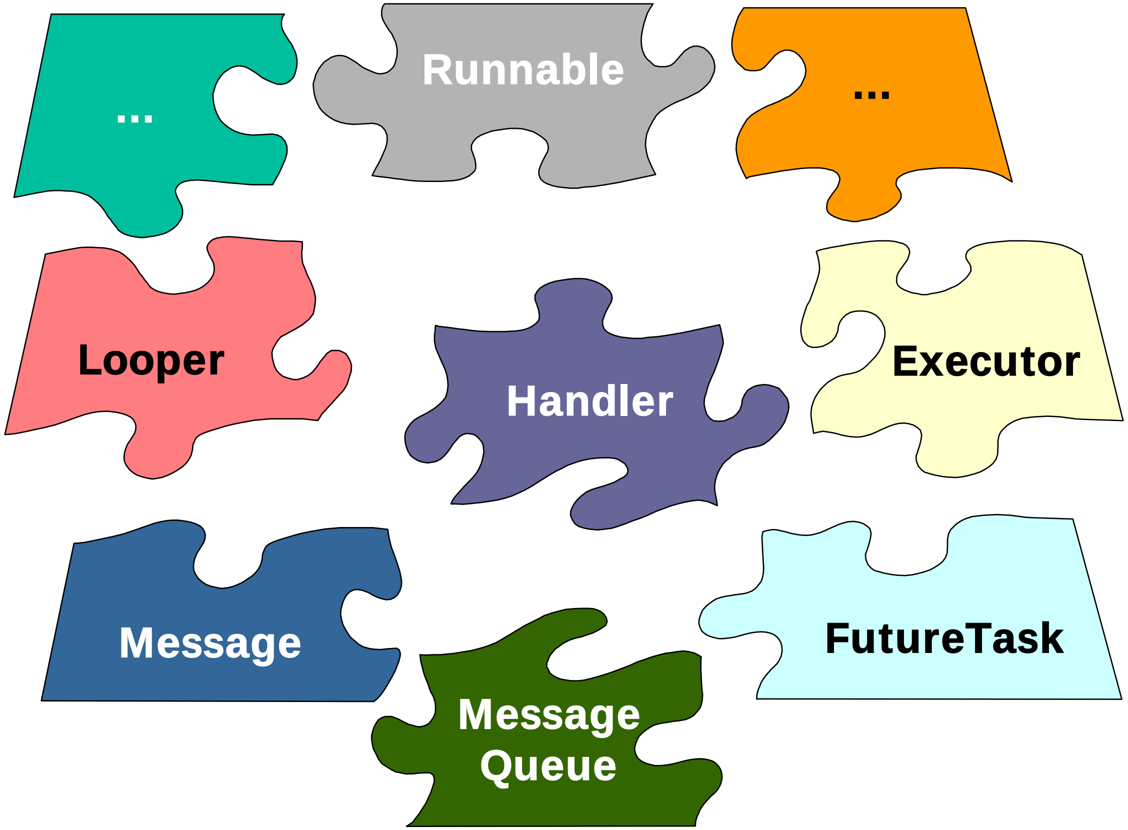
- These classes work together to embody key characteristics of frameworks
- We'll analyze all these classes throughout this Module
  - Interface classes visible to application developers
  - Implementation classes less visible to application developers



# Mapping Android Concurrency Frameworks

- These classes work together to embody key characteristics of frameworks
- We'll analyze all these classes throughout this Module

PATTERN-ORIENTED SOFTWARE ARCHITECTURE
A System of Patterns
Volume 1
WILEY
PATTERN-ORIENTED SOFTWARE ARCHITECTURE
Volume 2
Patterns for Concurrent and Networked Objects
WILEY
Design Patterns
Elements of Reusable Object-Oriented Software
Erich Gamma
Richard Helm
Ralph Johnson
John Vlissides

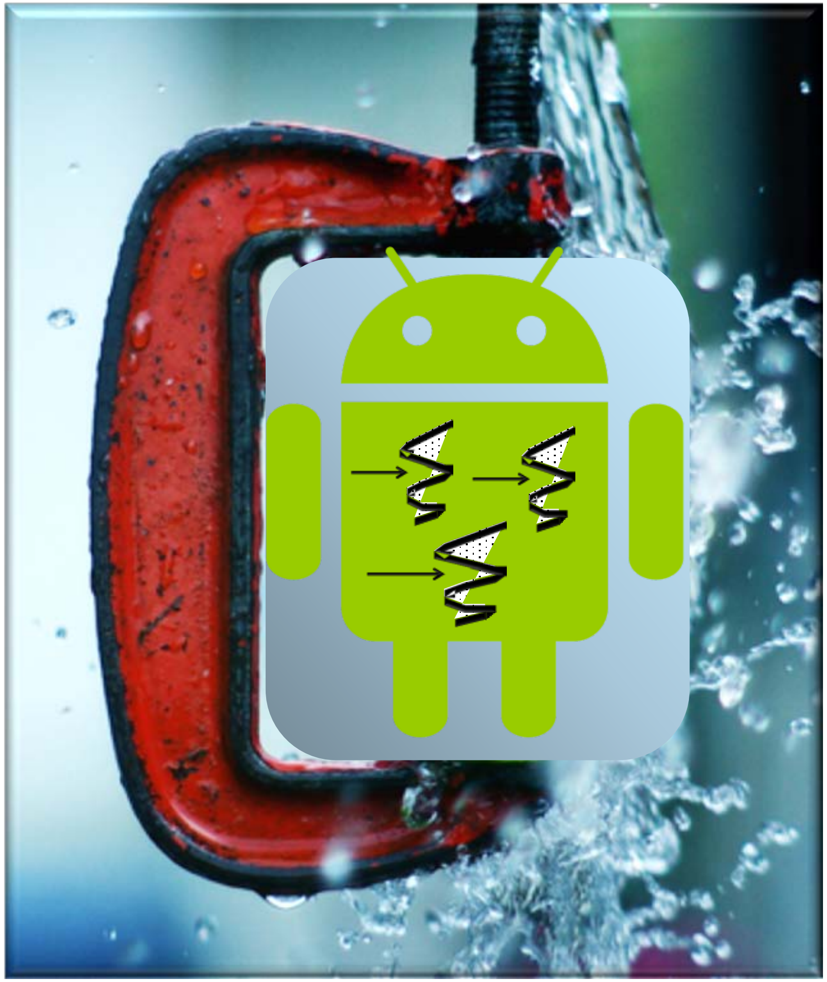


See upcoming section on "Android Concurrency & Communication Patterns"

# Summary

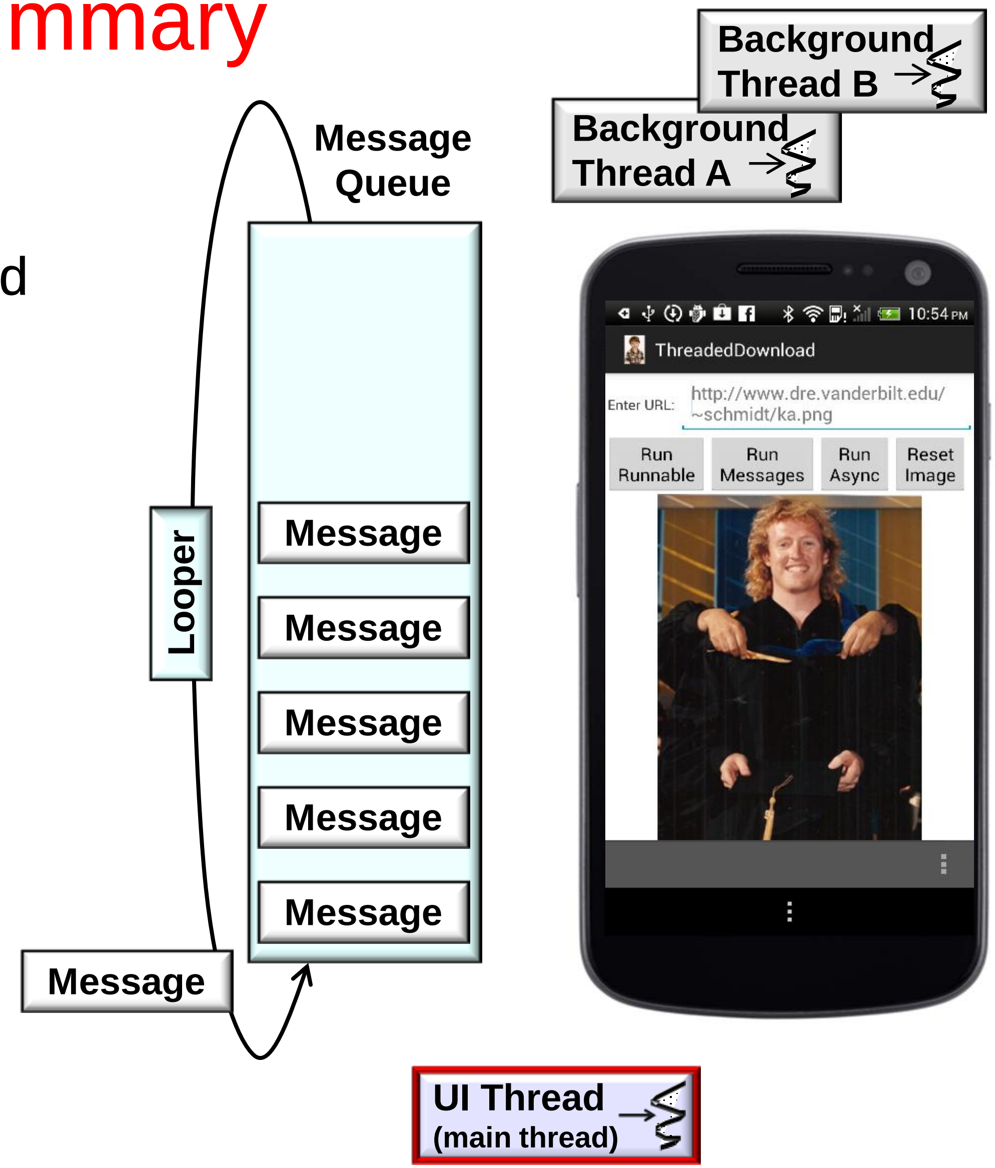
# Summary

- Android design constraints affect concurrent program development



# Summary

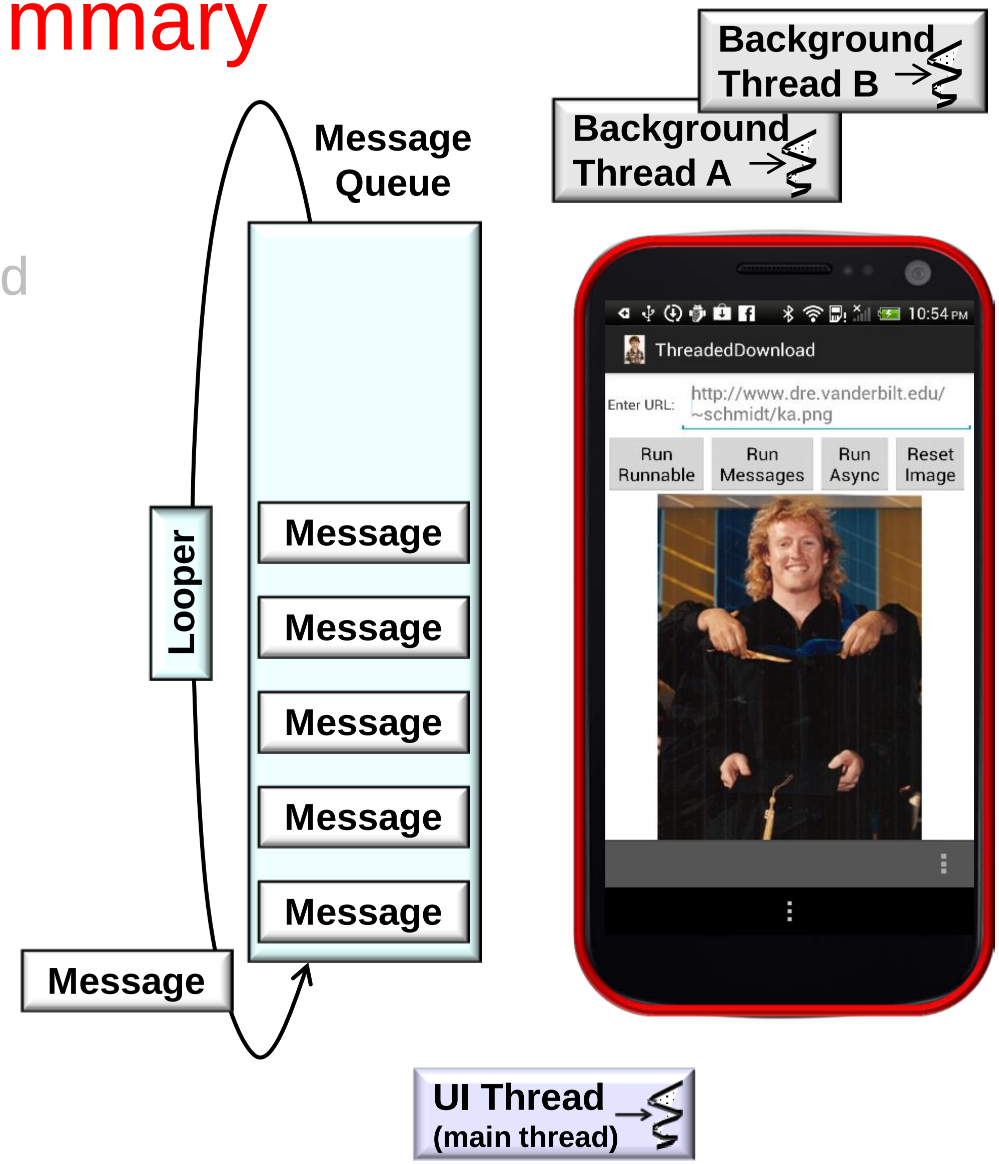
- Android design constraints affect concurrent program development
  - Android apps have one UI Thread



Enter URL: http://www.dre.vanderbilt.edu/~schmidt/ka.png

# Summary

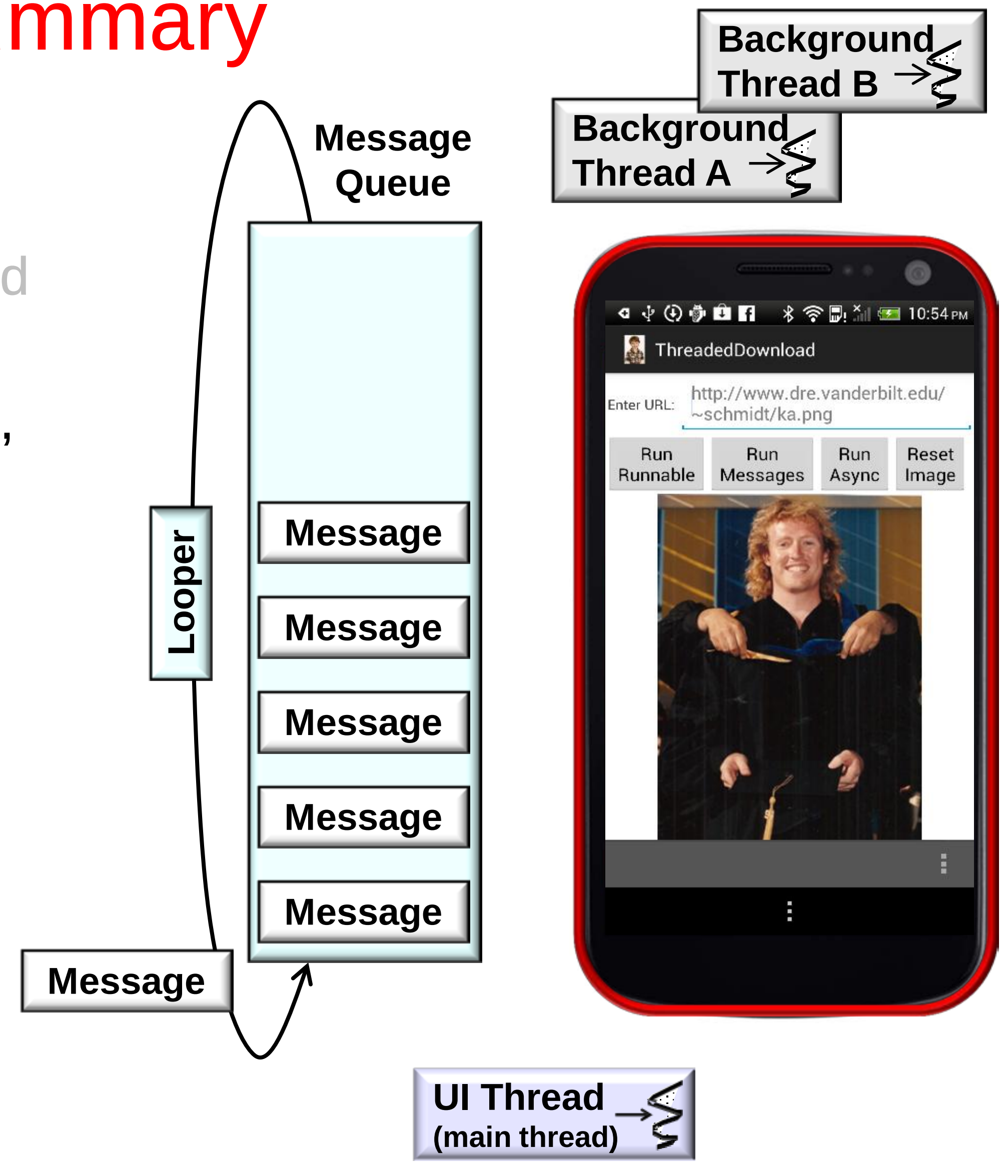
- Android design constraints affect concurrent program development
  - Android apps have one UI Thread
  - All components in the same process use the same UI Thread



Enter URL: http://www.dre.vanderbilt.edu/~schmidt/ka.png

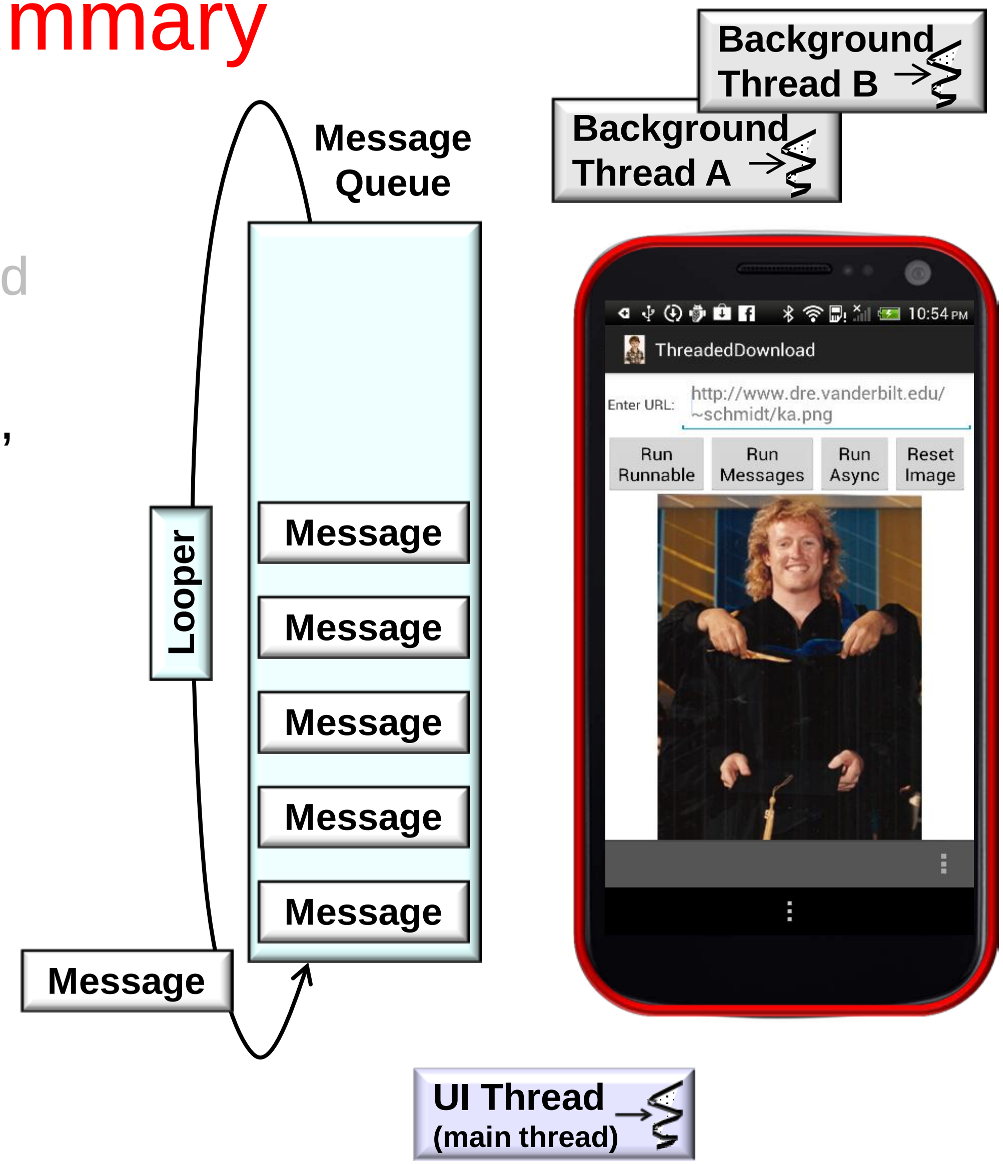
# Summary

- Android design constraints affect concurrent program development
  - Android apps have one UI Thread
  - All components in the same process use the same UI Thread, e.g.
    - Receive system notifications & broadcasts



# Summary

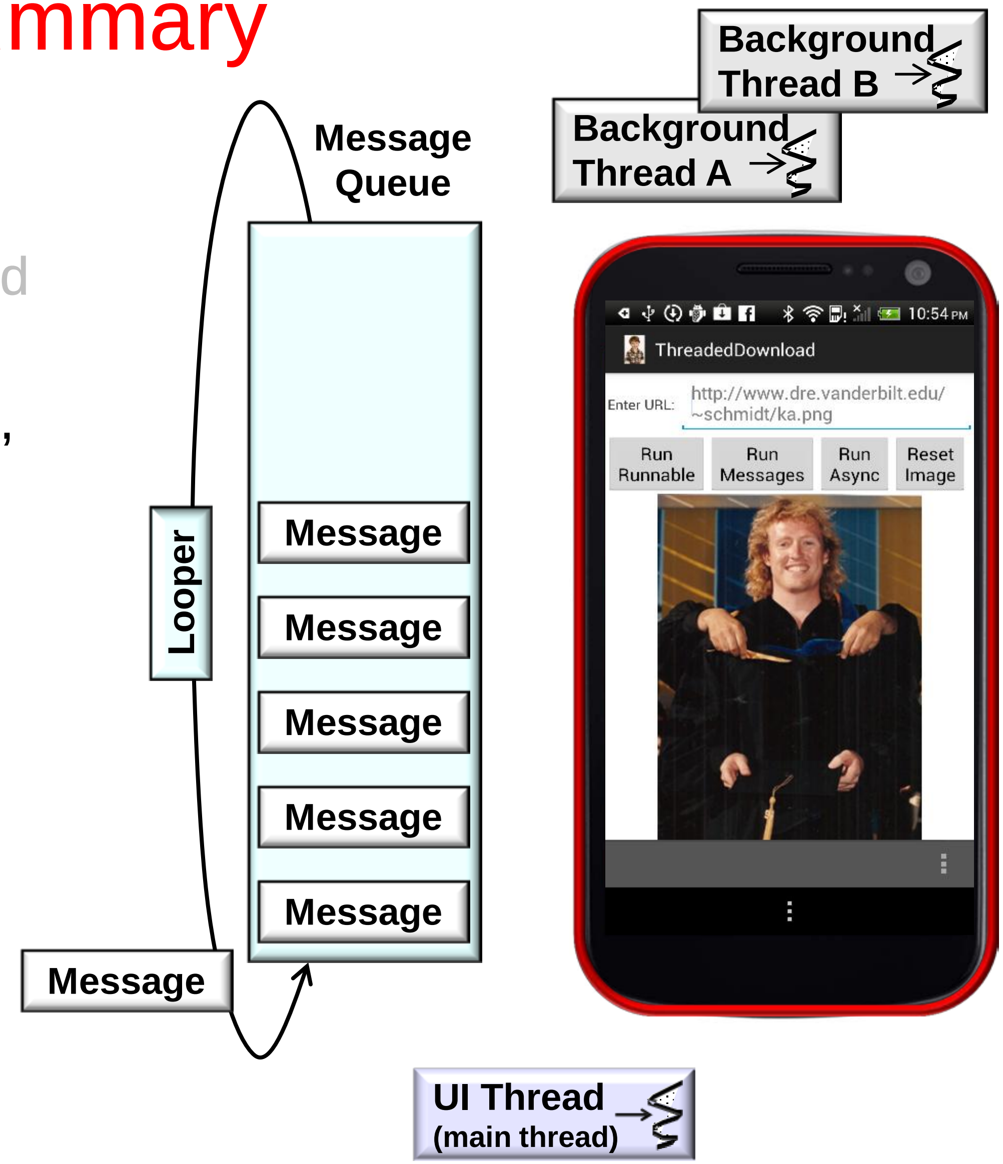
- Android design constraints affect concurrent program development
  - Android apps have one UI Thread
  - All components in the same process use the same UI Thread, e.g.
    - Receive system notifications & broadcasts
    - Interact with users



Enter URL: http://www.dre.vanderbilt.edu/~schmidt/ka.png

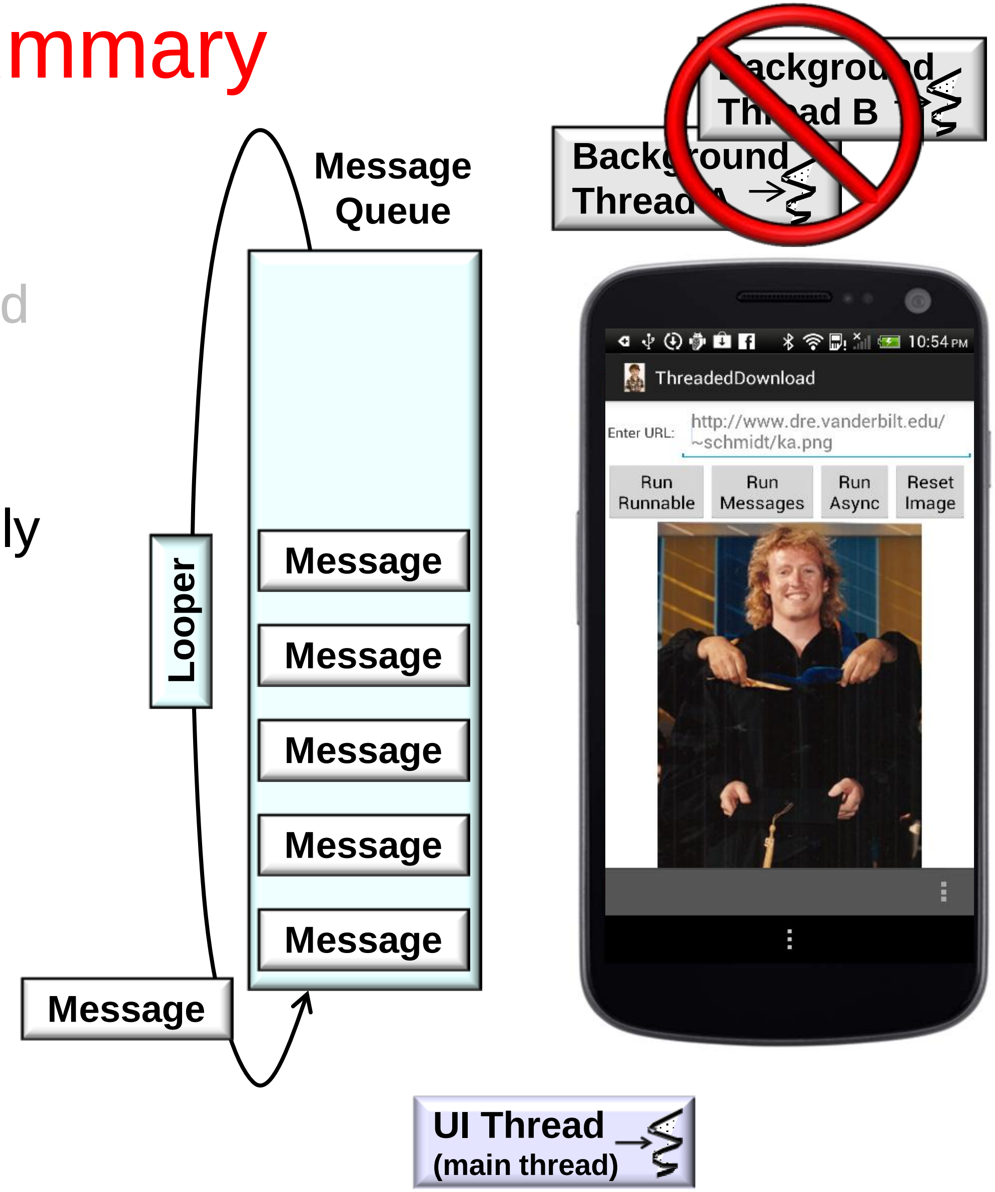
# Summary

- Android design constraints affect concurrent program development
  - Android apps have one UI Thread
  - All components in the same process use the same UI Thread, e.g.
    - Receive system notifications & broadcasts
    - Interact with users
    - Perform Activity lifecycle methods



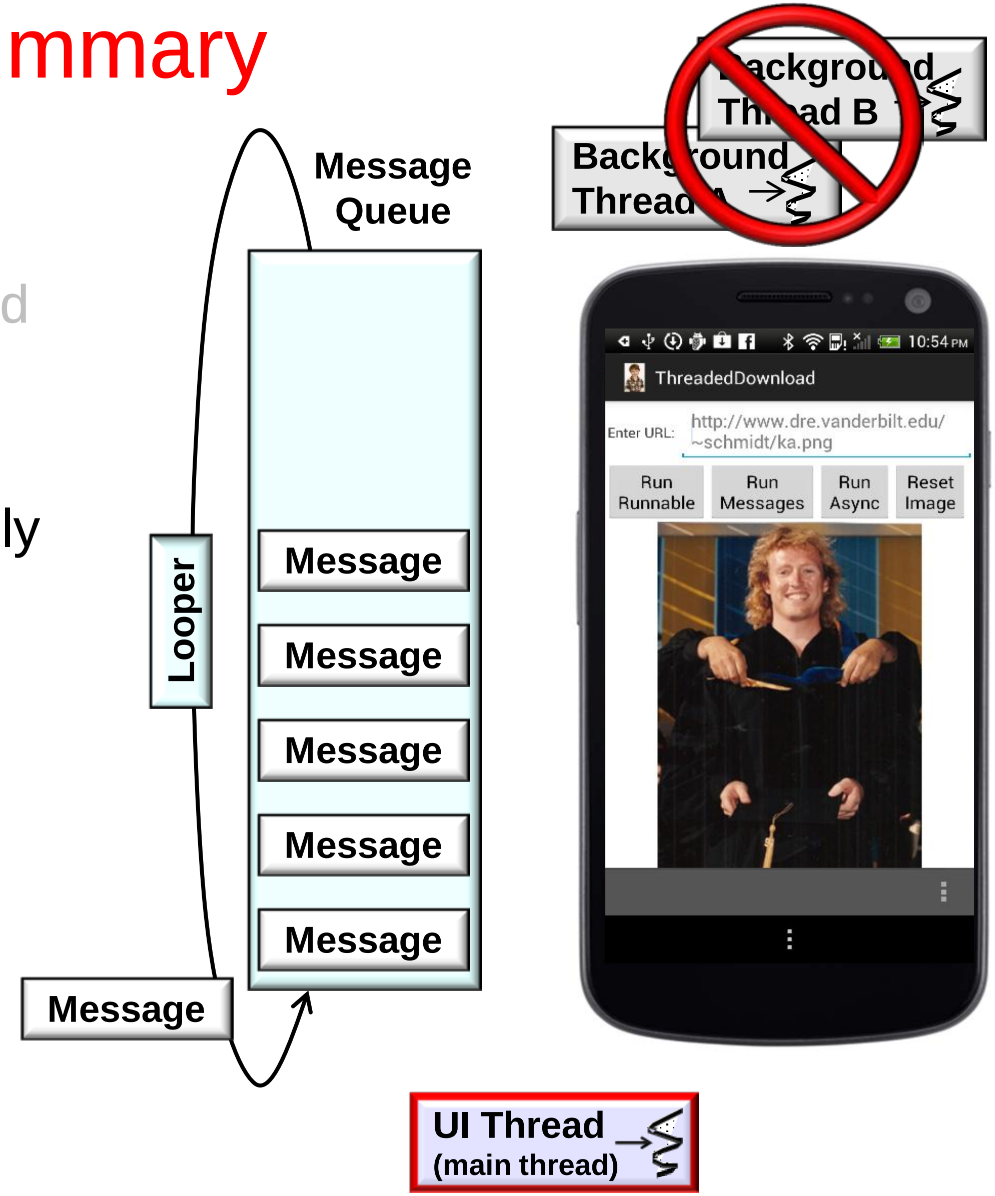
# Summary

- Android design constraints affect concurrent program development
  - Android apps have one UI Thread
  - All components in the same process use the same UI Thread
  - UI toolkit components should only be accessed by the UI Thread



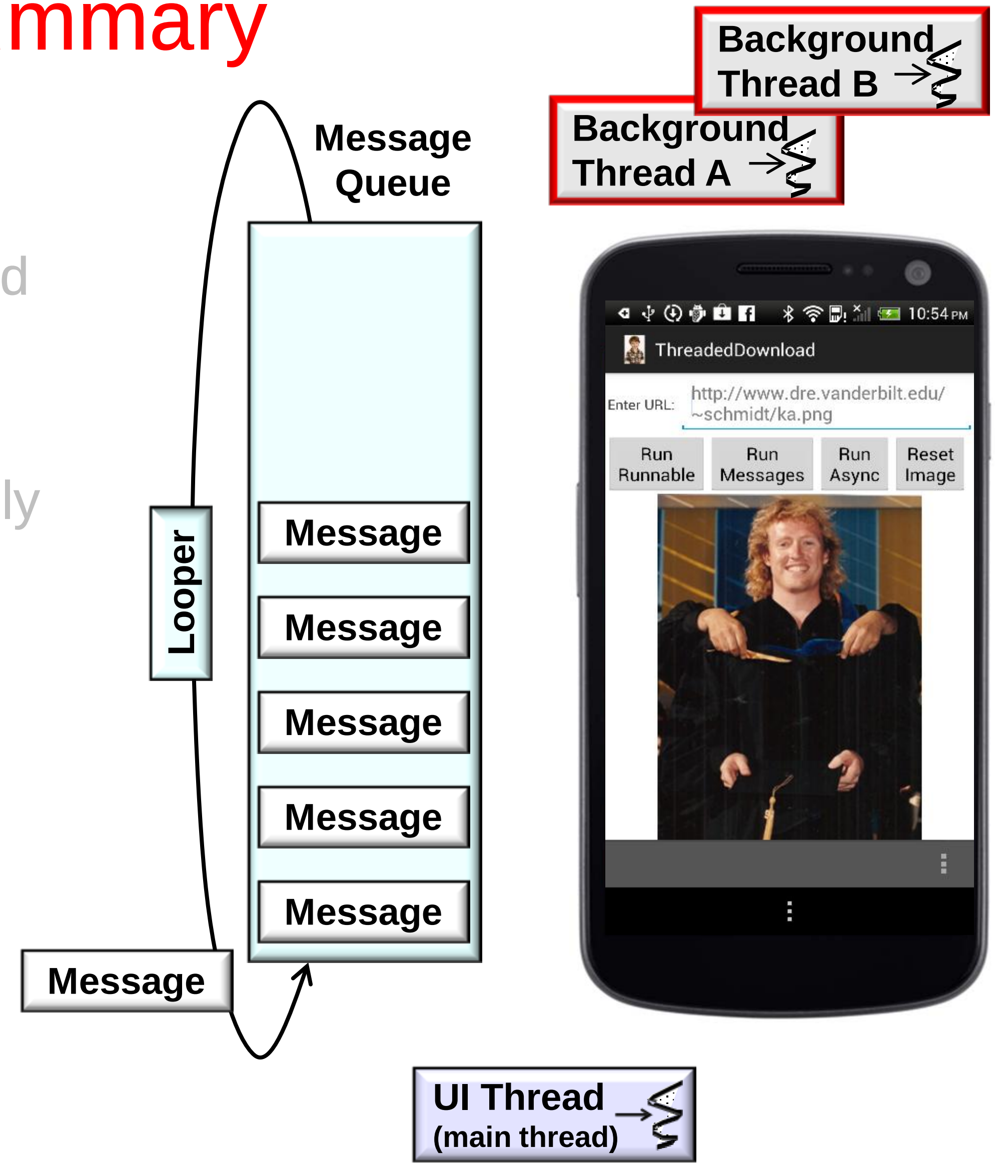
# Summary

- Android design constraints affect concurrent program development
  - Android apps have one UI Thread
  - All components in the same process use the same UI Thread
  - UI toolkit components should only be accessed by the UI Thread



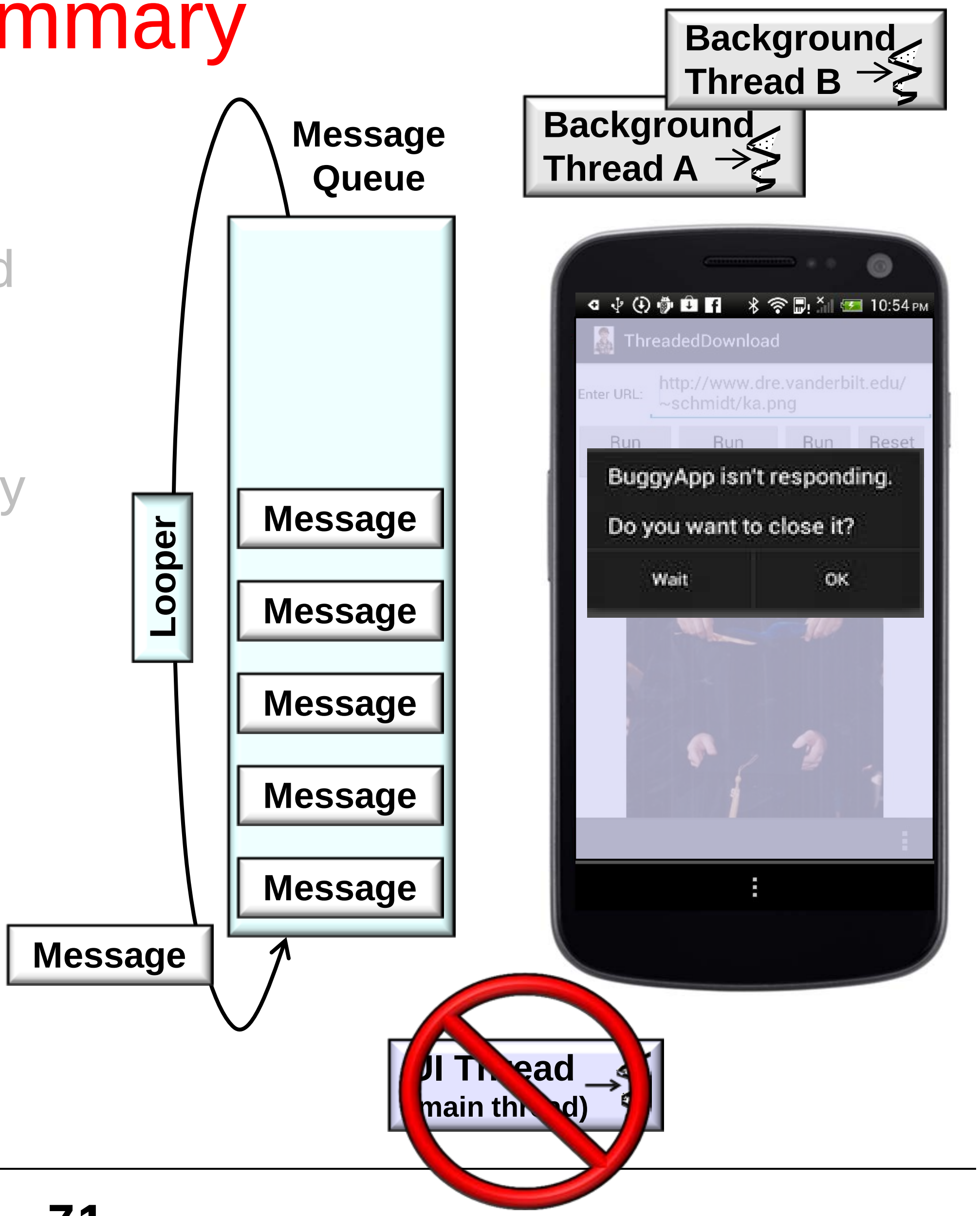
# Summary

- Android design constraints affect concurrent program development
  - Android apps have one UI Thread
  - All components in the same process use the same UI Thread
  - UI toolkit components should only be accessed by the UI Thread
  - Long-duration operations should run in background Thread(s) to avoid generating “ANRs”



# Summary

- Android design constraints affect concurrent program development
  - Android apps have one UI Thread
  - All components in the same process use the same UI Thread
  - UI toolkit components should only be accessed by the UI Thread
  - Long-duration operations should run in background Thread(s) to avoid generating "ANRs"

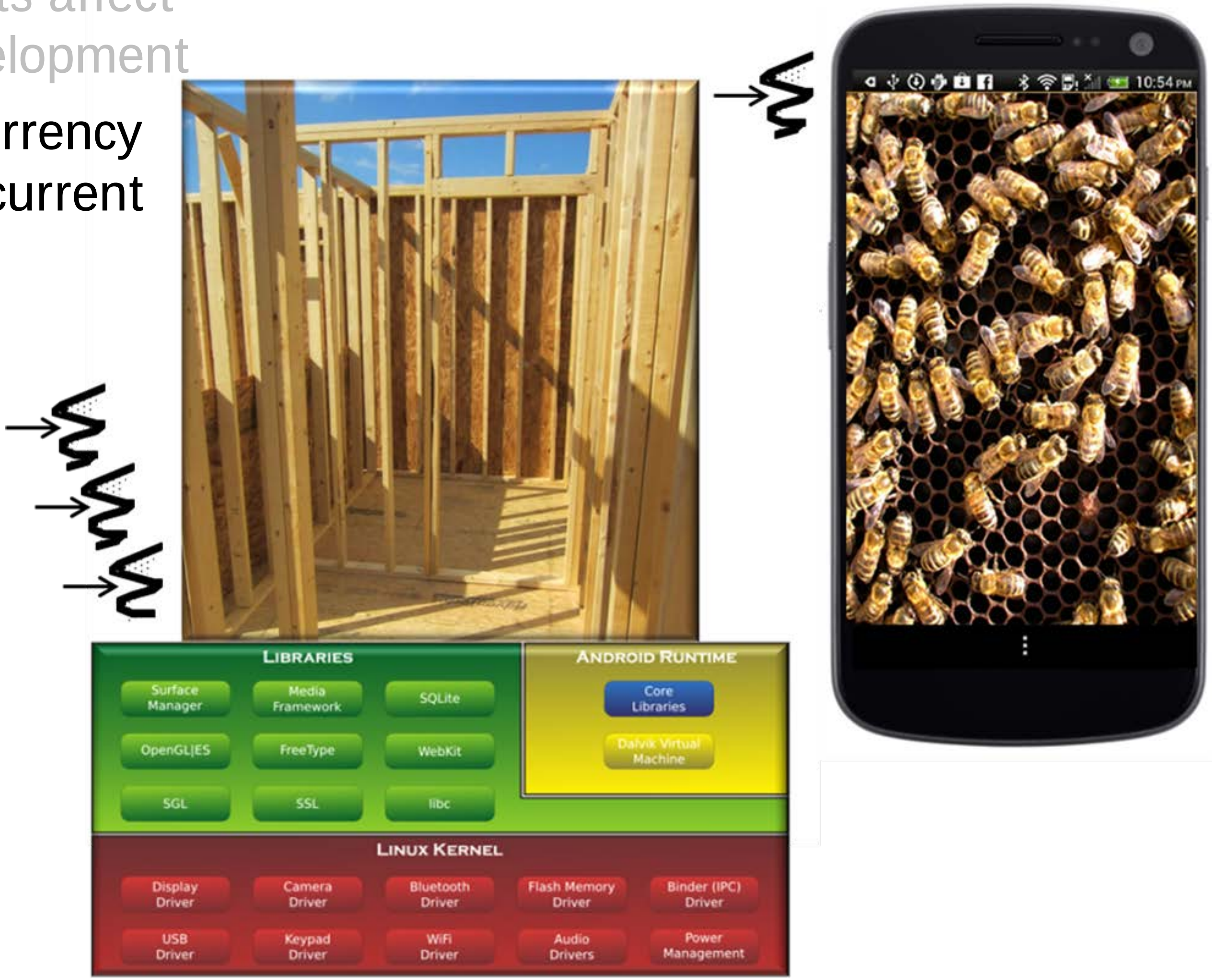


BuggyApp isn't responding.

Do you want to close it?

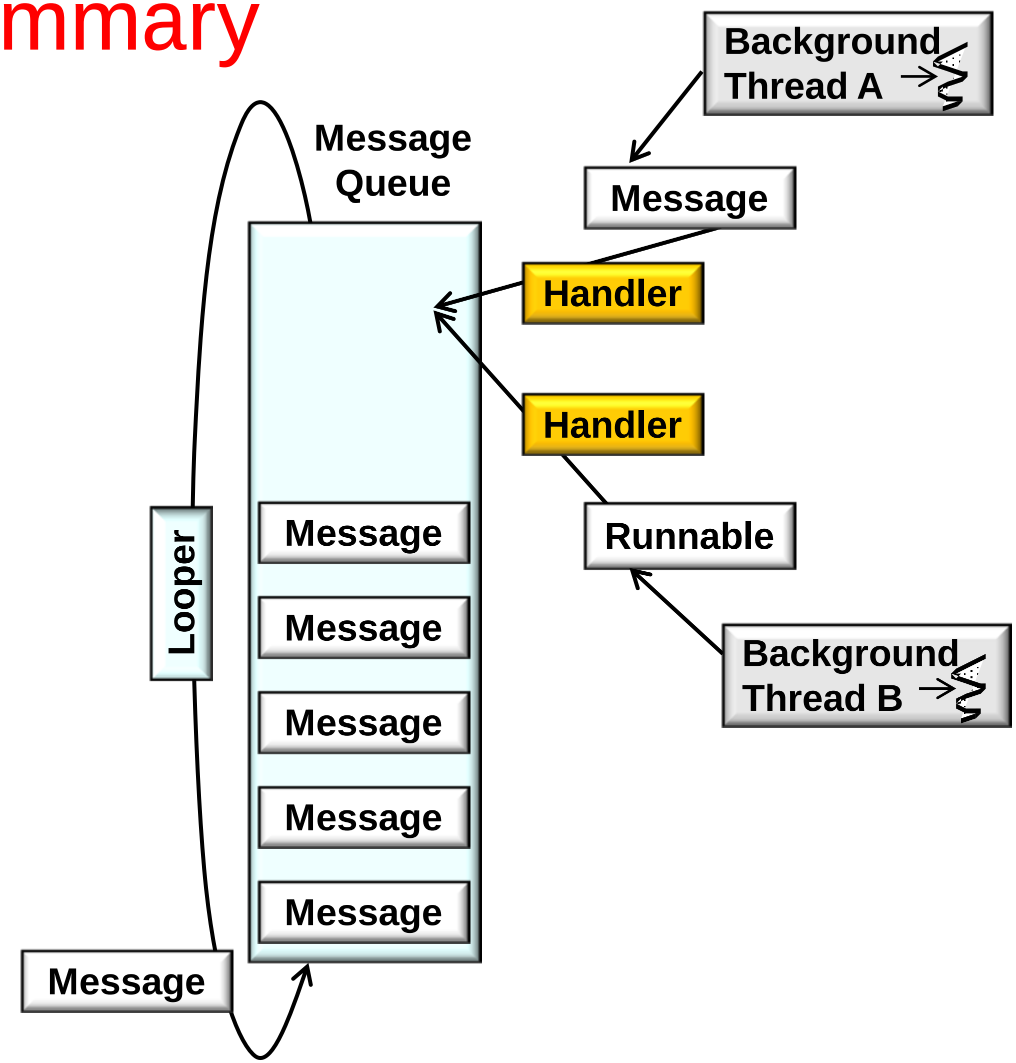
# Summary

- Android design constraints affect concurrent program development
- Android's common concurrency frameworks simplify concurrent programs



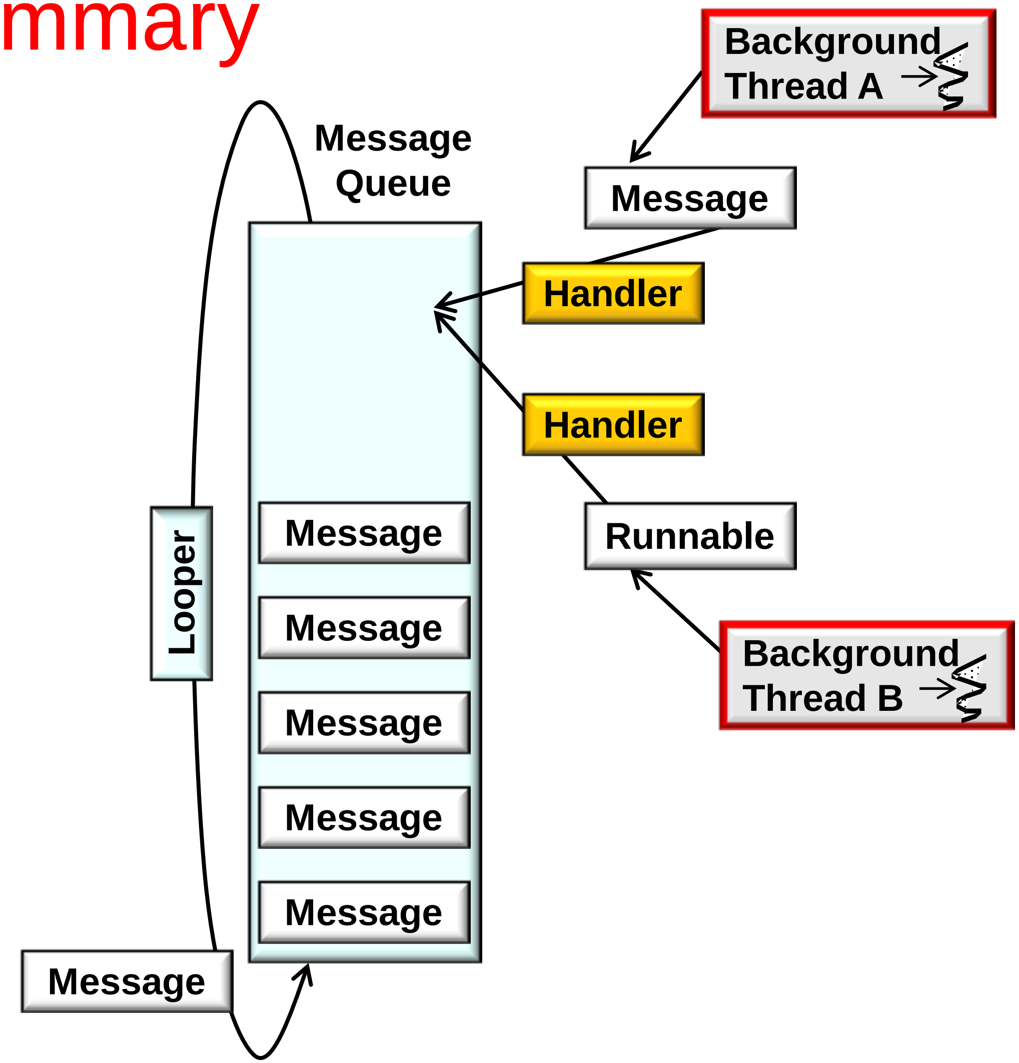
# Summary

- Android design constraints affect concurrent program development
- Android's common concurrency frameworks simplify concurrent programs, e.g.
    - **Handlers, Messages, & Runnables (HaMeR)**



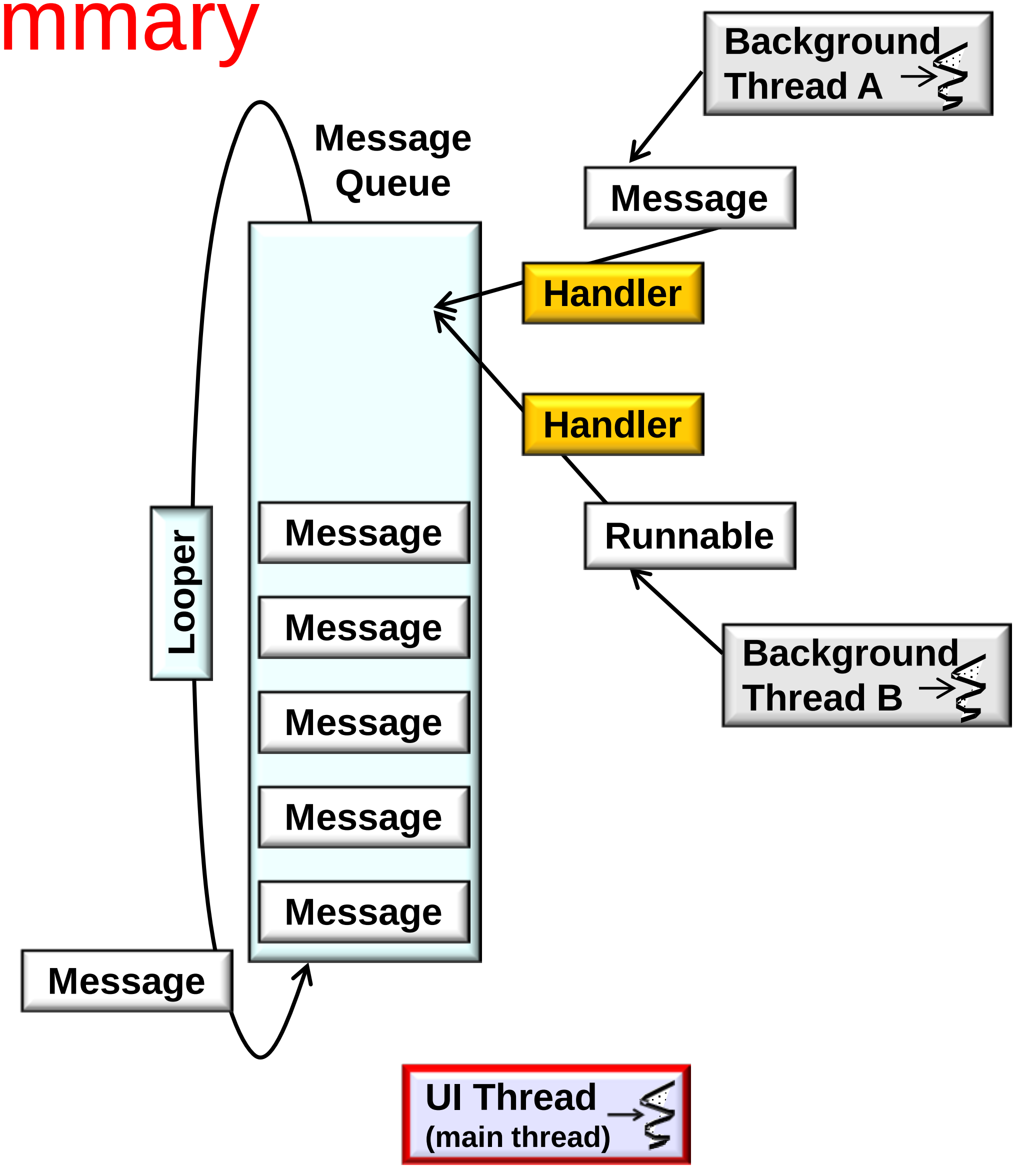
# Summary

- Android design constraints affect concurrent program development
- Android's common concurrency frameworks simplify concurrent programs, e.g.
  - **Handlers, Messages, & Runnables (HaMeR)**



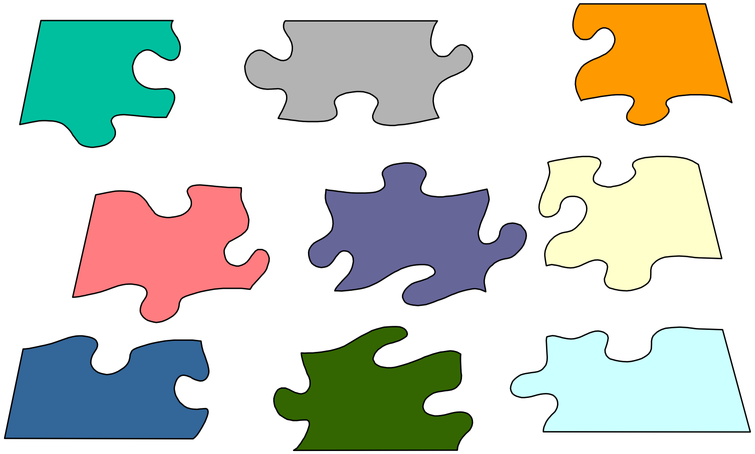
# Summary

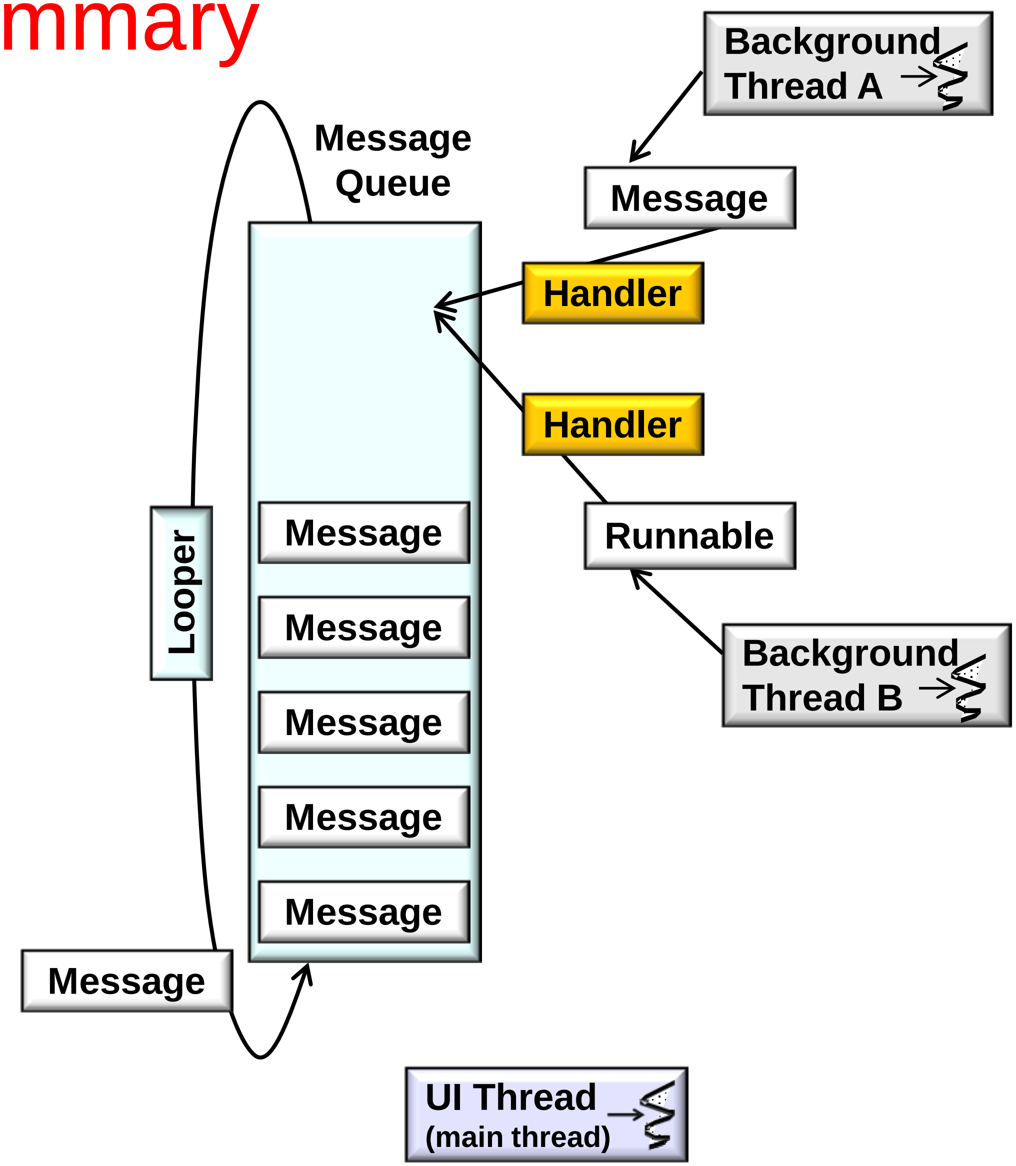
- Android design constraints affect concurrent program development
- Android's common concurrency frameworks simplify concurrent programs, e.g.
  - **Handlers, Messages, & Runnables (HaMeR)**



# Summary

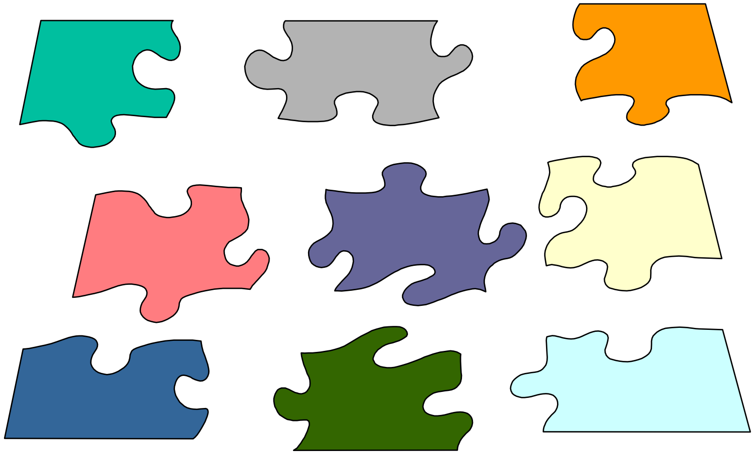
- Android design constraints affect concurrent program development
- Android's common concurrency frameworks simplify concurrent programs, e.g.
  - **Handlers, Messages, & Runnables (HaMeR)**
    - Classes in HaMeR are loosely connected

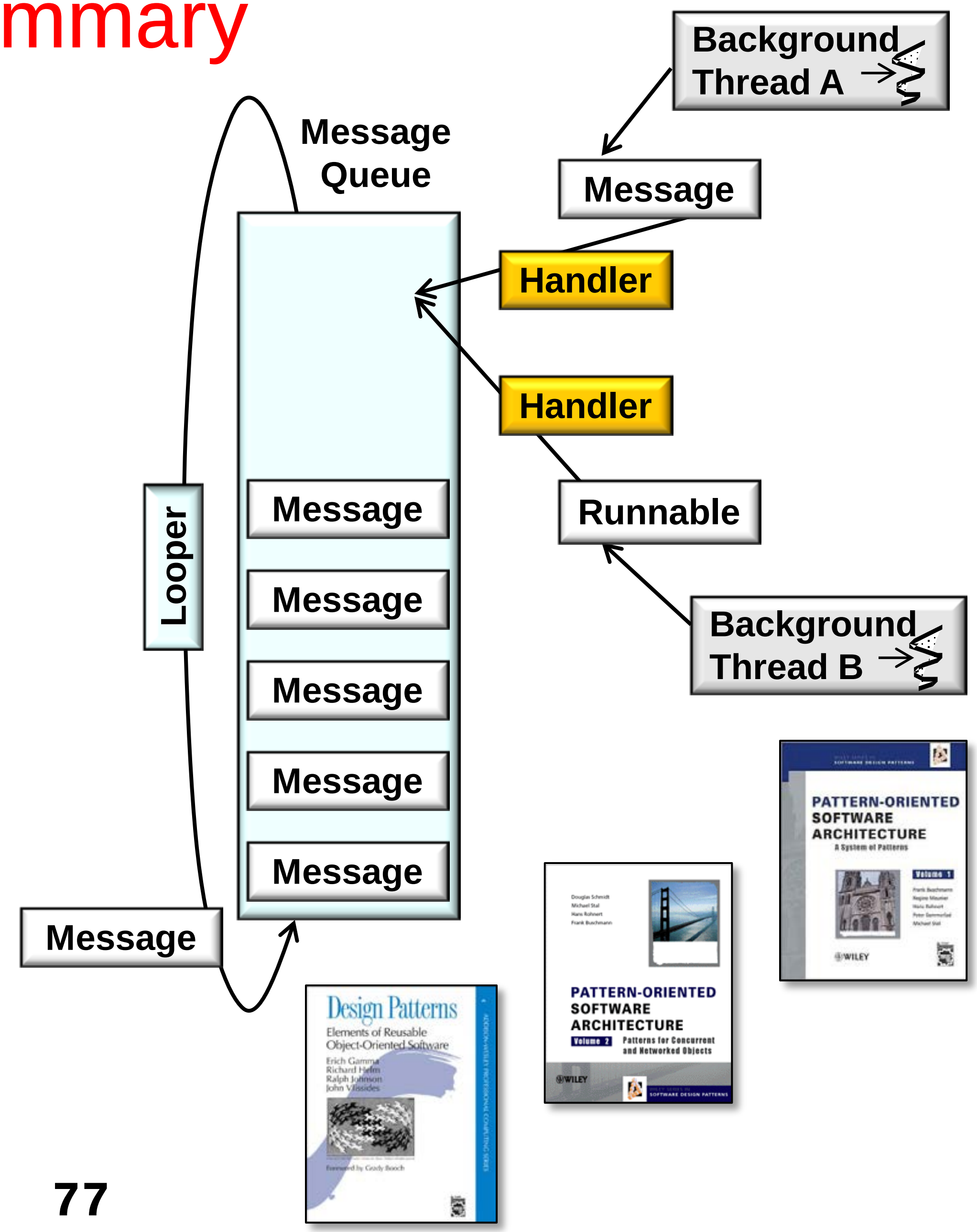




# Summary

- Android design constraints affect concurrent program development
- Android’s common concurrency frameworks simplify concurrent programs, e.g.
  - **Handlers, Messages, & Runnables (HaMeR)**
    - Classes in HaMeR are loosely connected





Background Thread A

Message

Message Queue

Handler

Handler

Looper

Message

Message

Message

Message

Message

Runnable

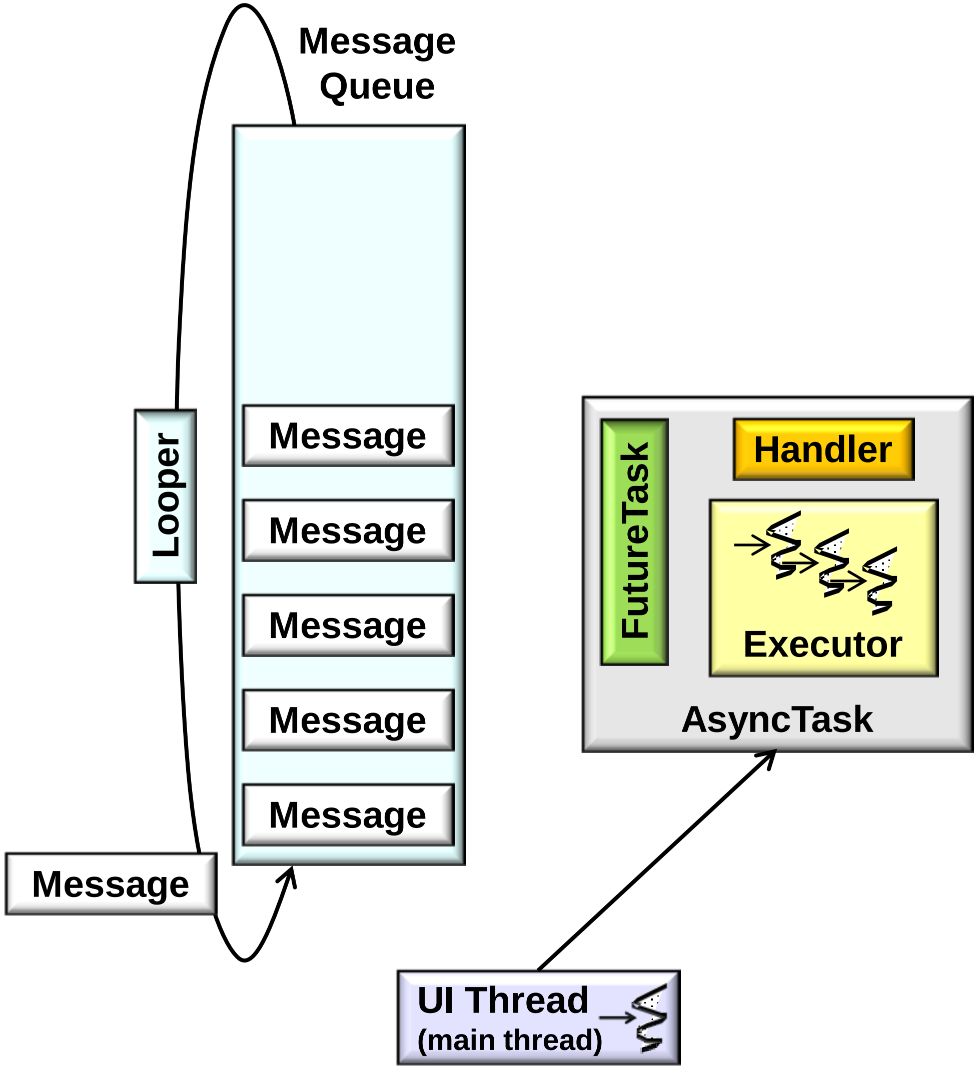
Background Thread B

Message

Design Patterns
Elements of Reusable Object-Oriented Software
Erich Gamma
Richard Helm
Ralph Johnson
John Vlissides
PATTERN-ORIENTED SOFTWARE ARCHITECTURE
Volume 2
Patterns for Concurrent and Networked Objects
WILEY
PATTERN-ORIENTED SOFTWARE ARCHITECTURE
A System of Patterns
Volume 1
WILEY

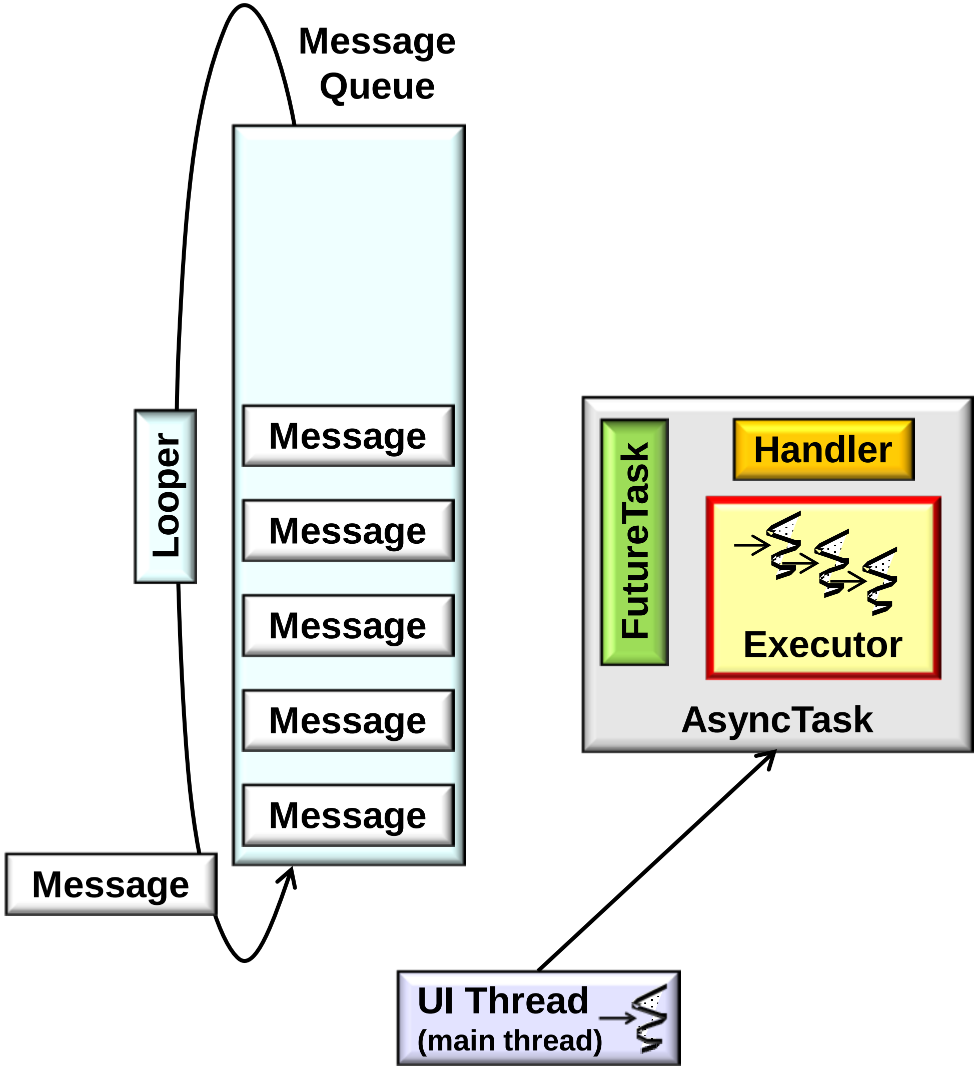
# Summary

- Android design constraints affect concurrent program development
- Android's common concurrency frameworks simplify concurrent programs, e.g.
  - **Handlers, Messages, & Runnables (HaMeR)**
  - **AsyncTask**



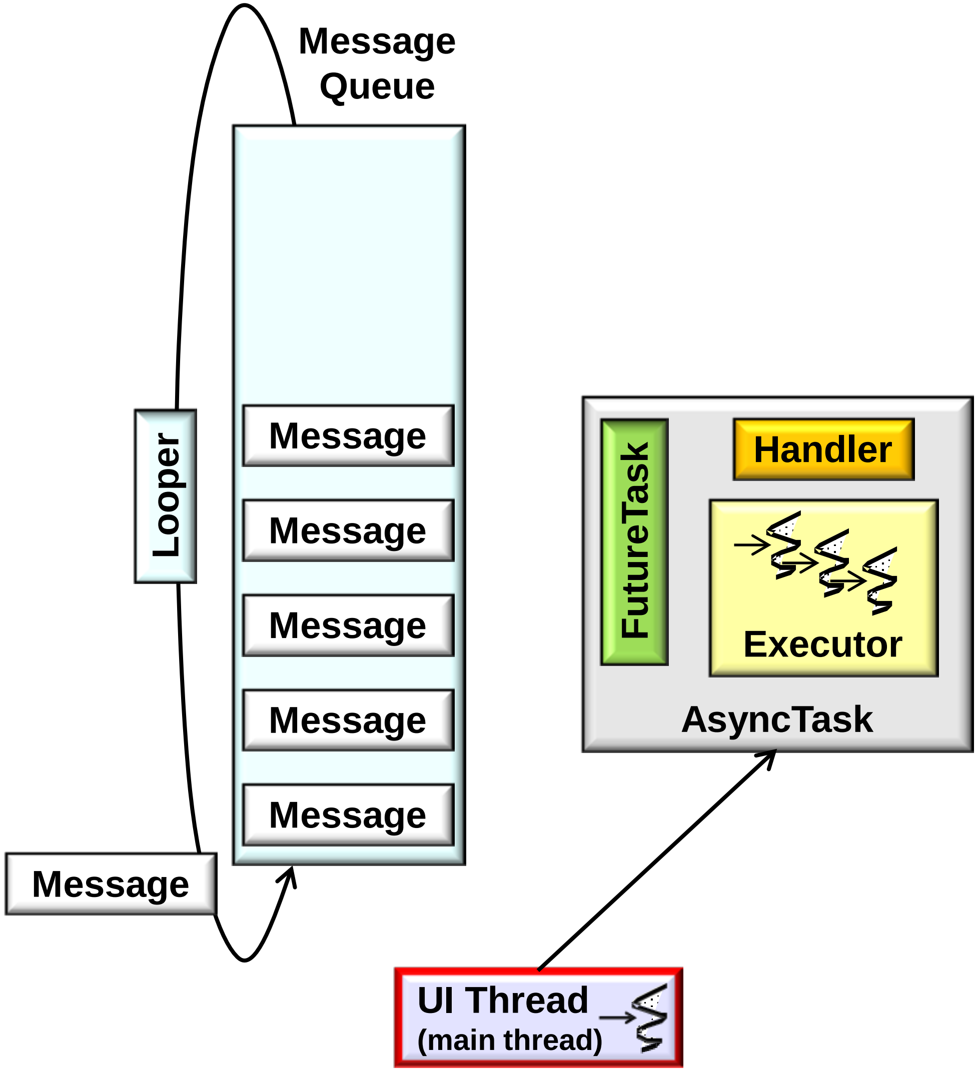
# Summary

- Android design constraints affect concurrent program development
- Android's common concurrency frameworks simplify concurrent programs, e.g.
  - **Handlers, Messages, & Runnables (HaMeR)**
  - **AsyncTask**



# Summary

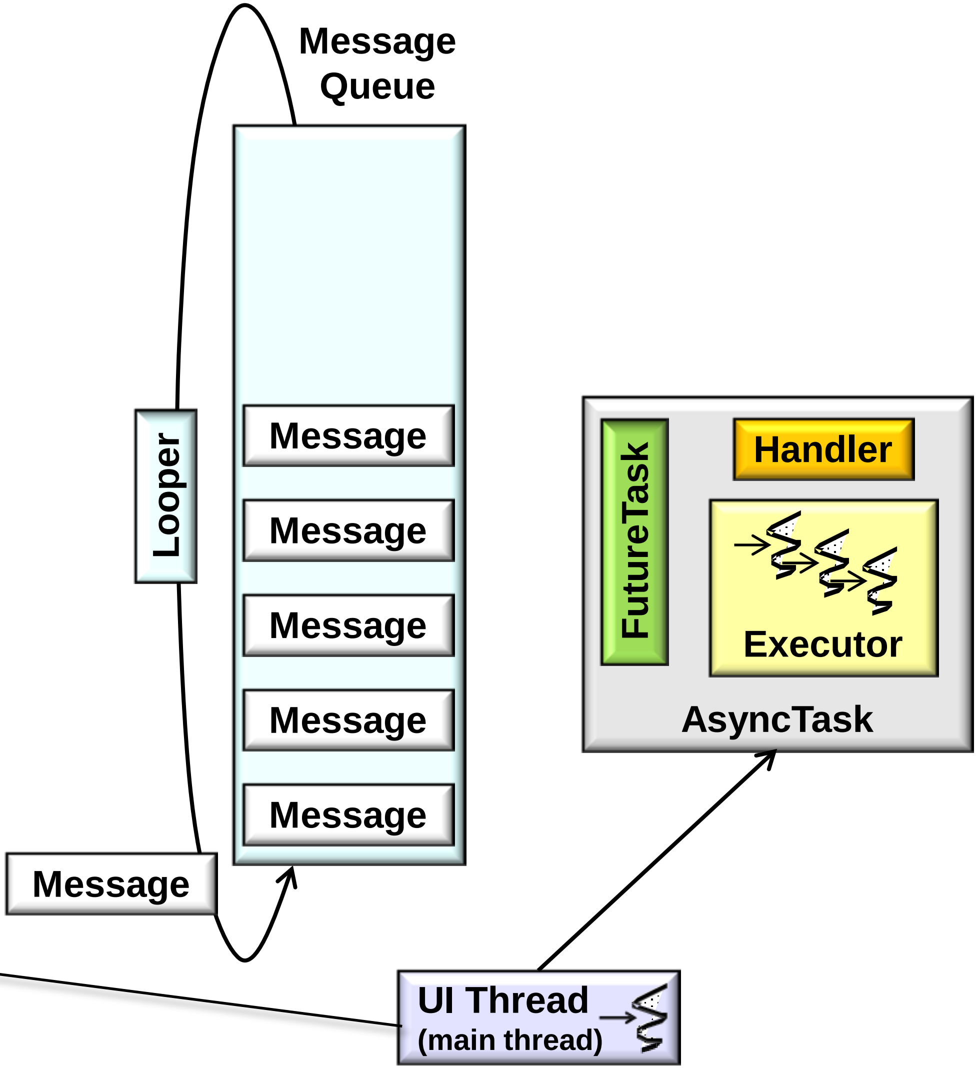
- Android design constraints affect concurrent program development
- Android's common concurrency frameworks simplify concurrent programs, e.g.
  - **Handlers, Messages, & Runnables (HaMeR)**
  - **AsyncTask**



# Summary

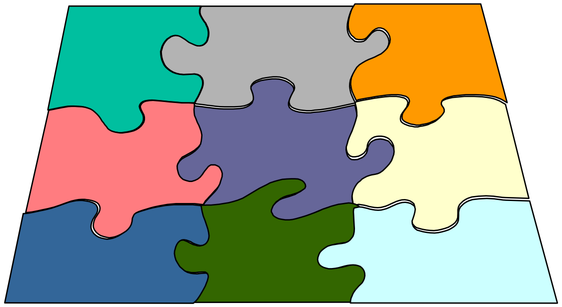
- Android design constraints affect concurrent program development
- Android's common concurrency frameworks simplify concurrent programs, e.g.
  - **Handlers, Messages, & Runnables (HaMeR)**
  - **AsyncTask**

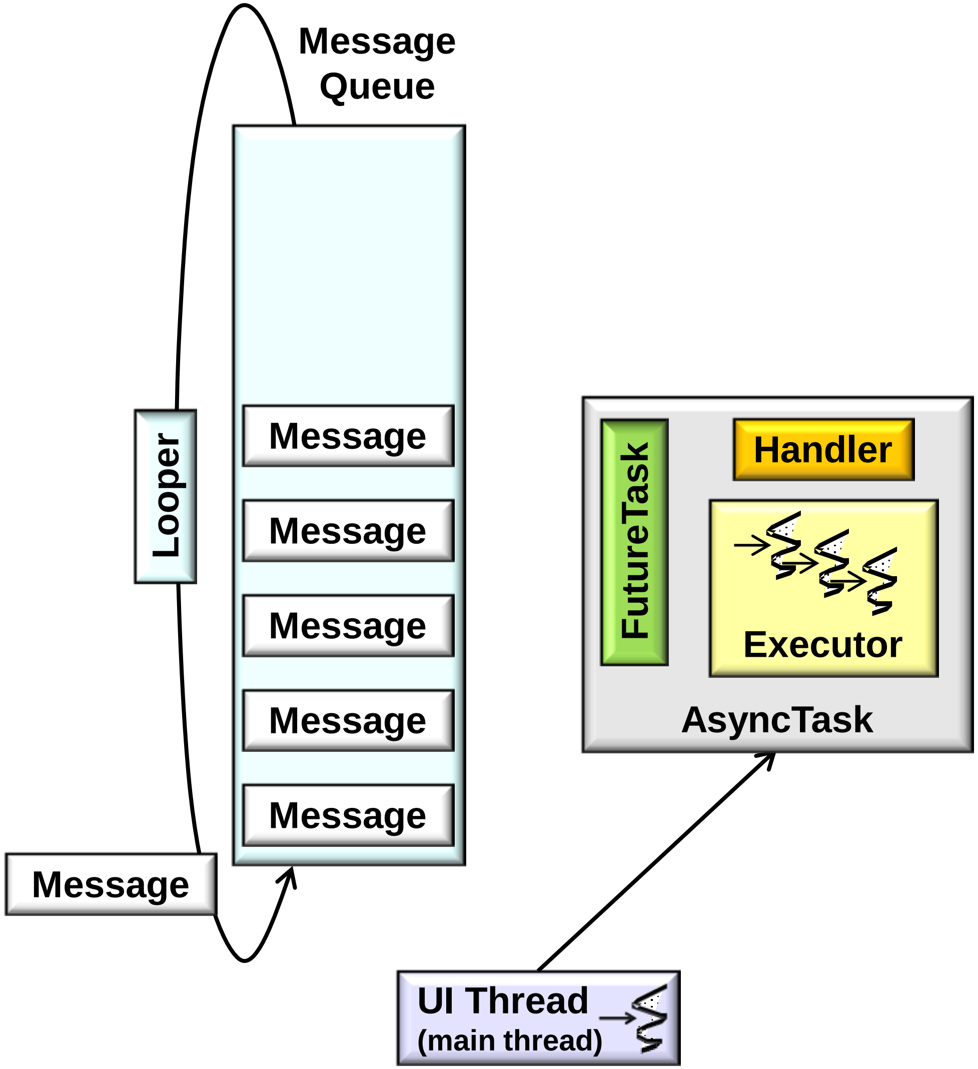
*No need for applications to manipulate Threads, Handlers, Messages, or Runnables directly*



# Summary

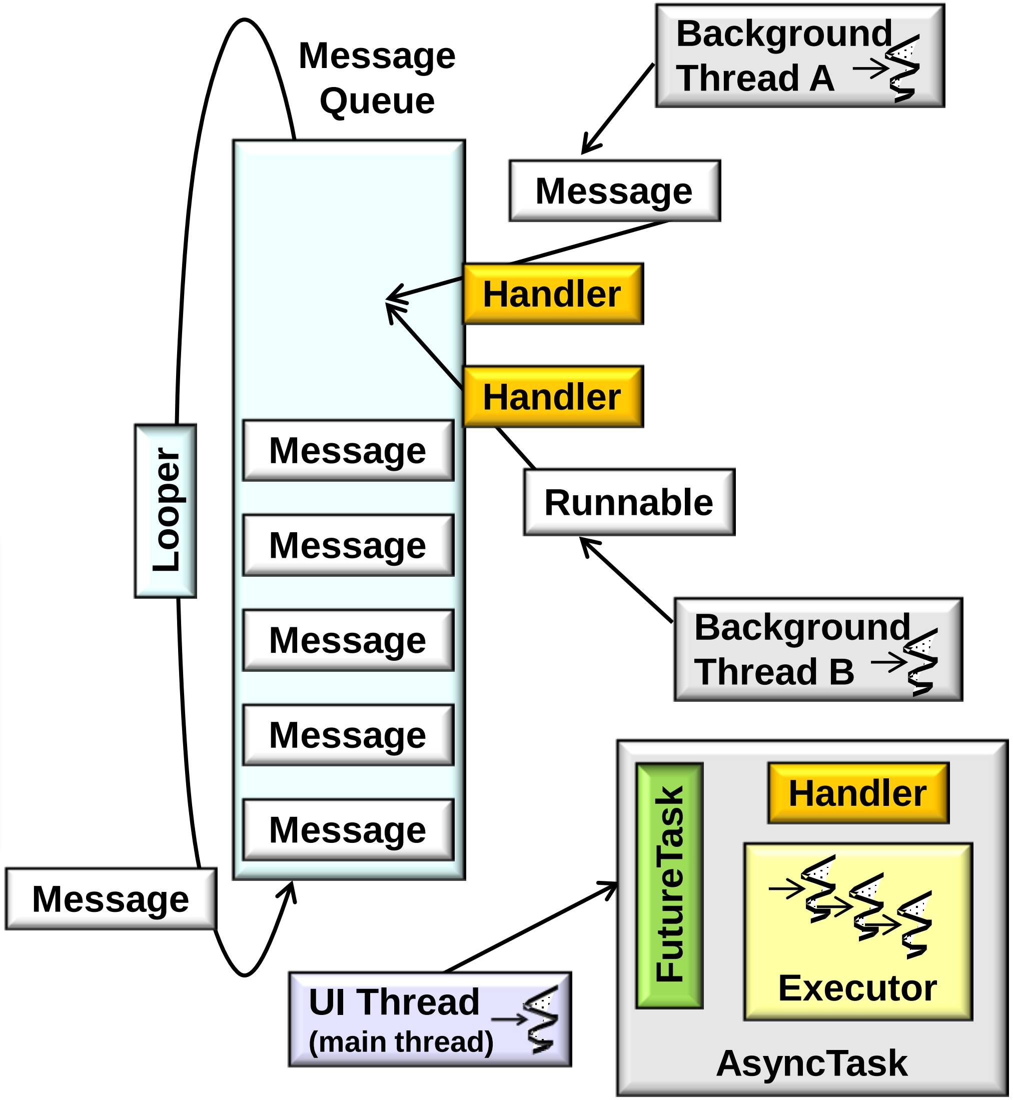
- Android design constraints affect concurrent program development
- Android's common concurrency frameworks simplify concurrent programs, e.g.
  - **Handlers, Messages, & Runnables (HaMeR)**
  - **AsyncTask**
    - Classes in AsyncTask are more strongly connected

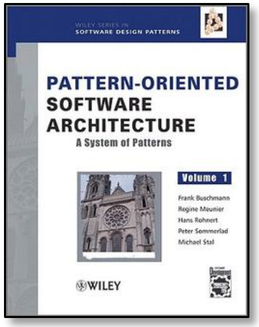




# Summary

- Android design constraints affect concurrent program development
- Android's common concurrency frameworks simplify concurrent programs
- Both frameworks are designed using *patterns*



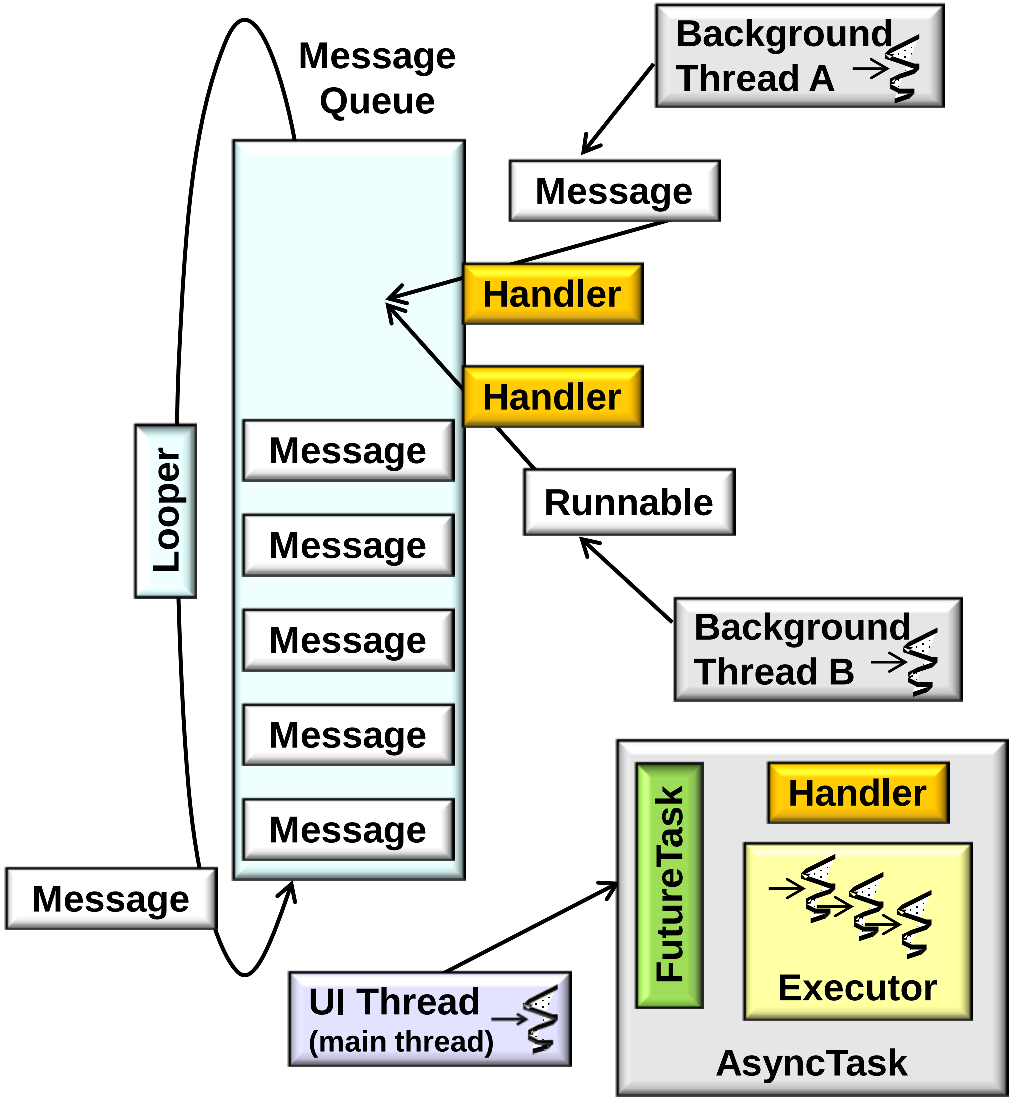


Douglas Schmidt
Michael Stal
Hans Rohnert
Frank Buschmann
PATTERN-ORIENTED SOFTWARE ARCHITECTURE
Volume 2
Patterns for Concurrent and Networked Objects
WILEY
WILEY SERIES IN SOFTWARE DESIGN PATTERNS

Design Patterns
Elements of Reusable Object-Oriented Software
Erich Gamma
Richard Helm
Ralph Johnson
John Vlissides
Foreword by Grady Booch
ADDISON-WESLEY PROFESSIONAL COMPUTING SERIES

# Summary

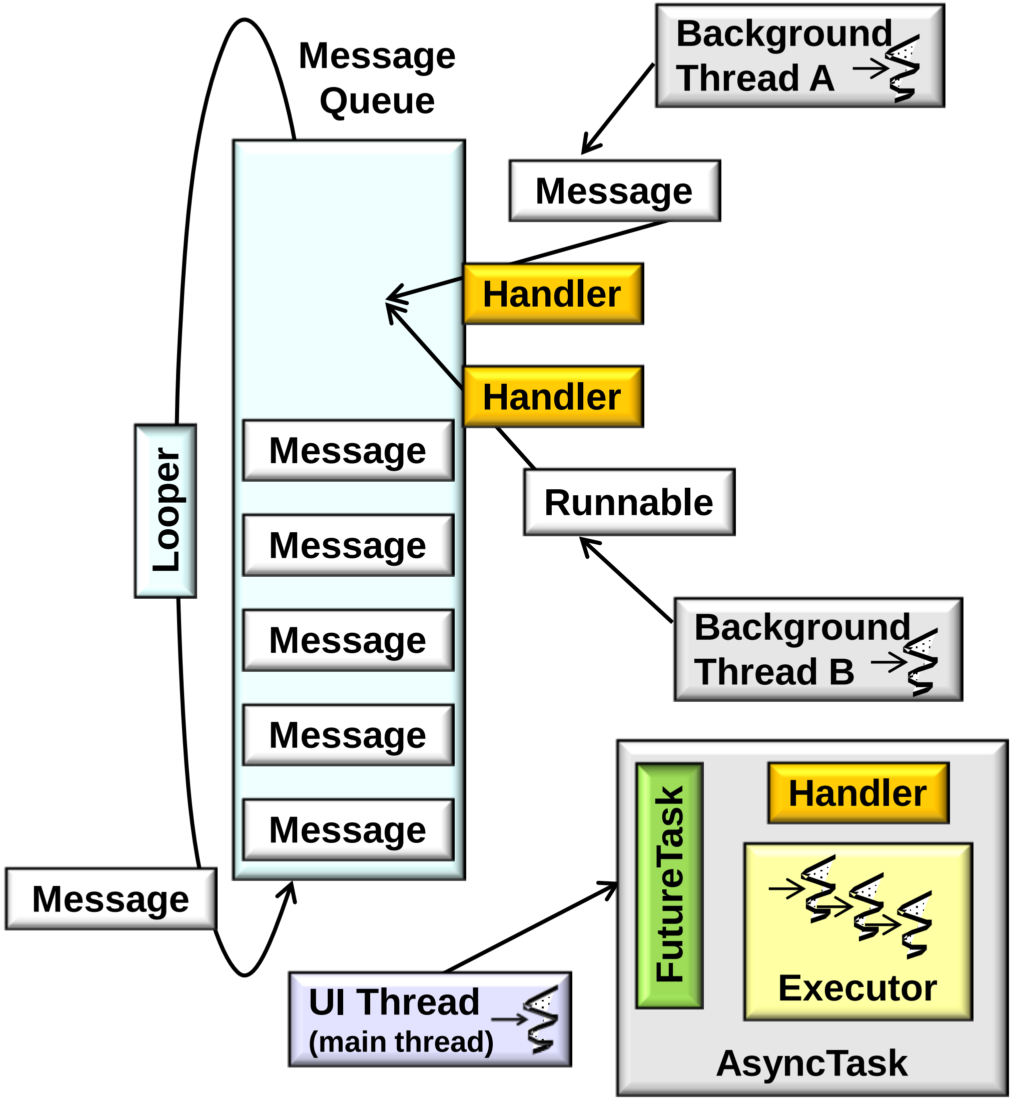
- Android design constraints affect concurrent program development
- Android's common concurrency frameworks simplify concurrent programs
- Both frameworks are designed using *patterns* & yield *idioms*



HaMeR & AsyncTask frameworks embody Android-specific concurrency idioms

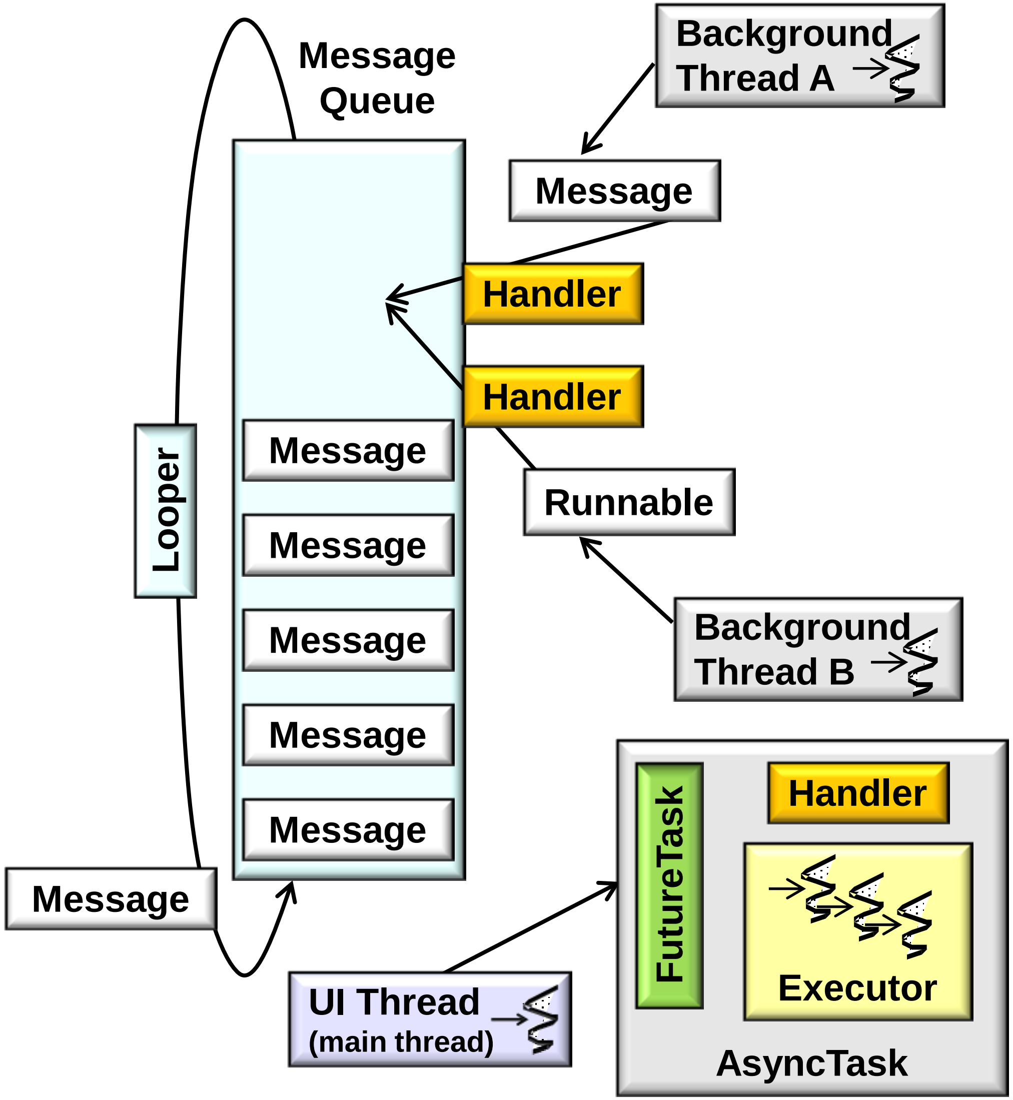
# Summary

- Android design constraints affect concurrent program development
- Android's common concurrency frameworks simplify concurrent programs
- Both frameworks are designed using *patterns* & yield *idioms*
  - An idiom is a pattern that's specific to a particular context



# Summary

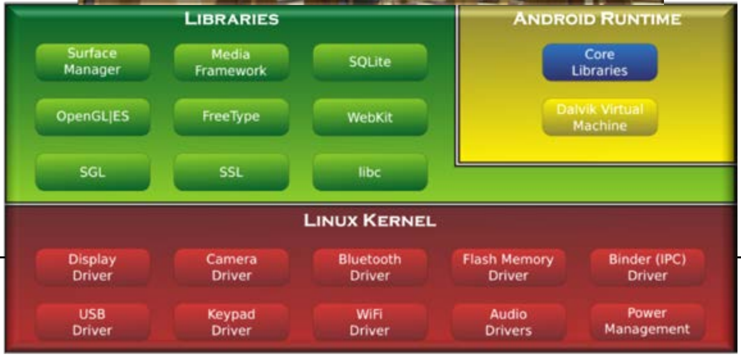
- Android design constraints affect concurrent program development
- Android's common concurrency frameworks simplify concurrent programs
- Both frameworks are designed using *patterns* & yield *idioms*
  - An idiom is a pattern that's specific to a particular context
    - e.g., development platform, programming language, or design method



See en.wikipedia.org/wiki/Programming_idiom for more on idioms

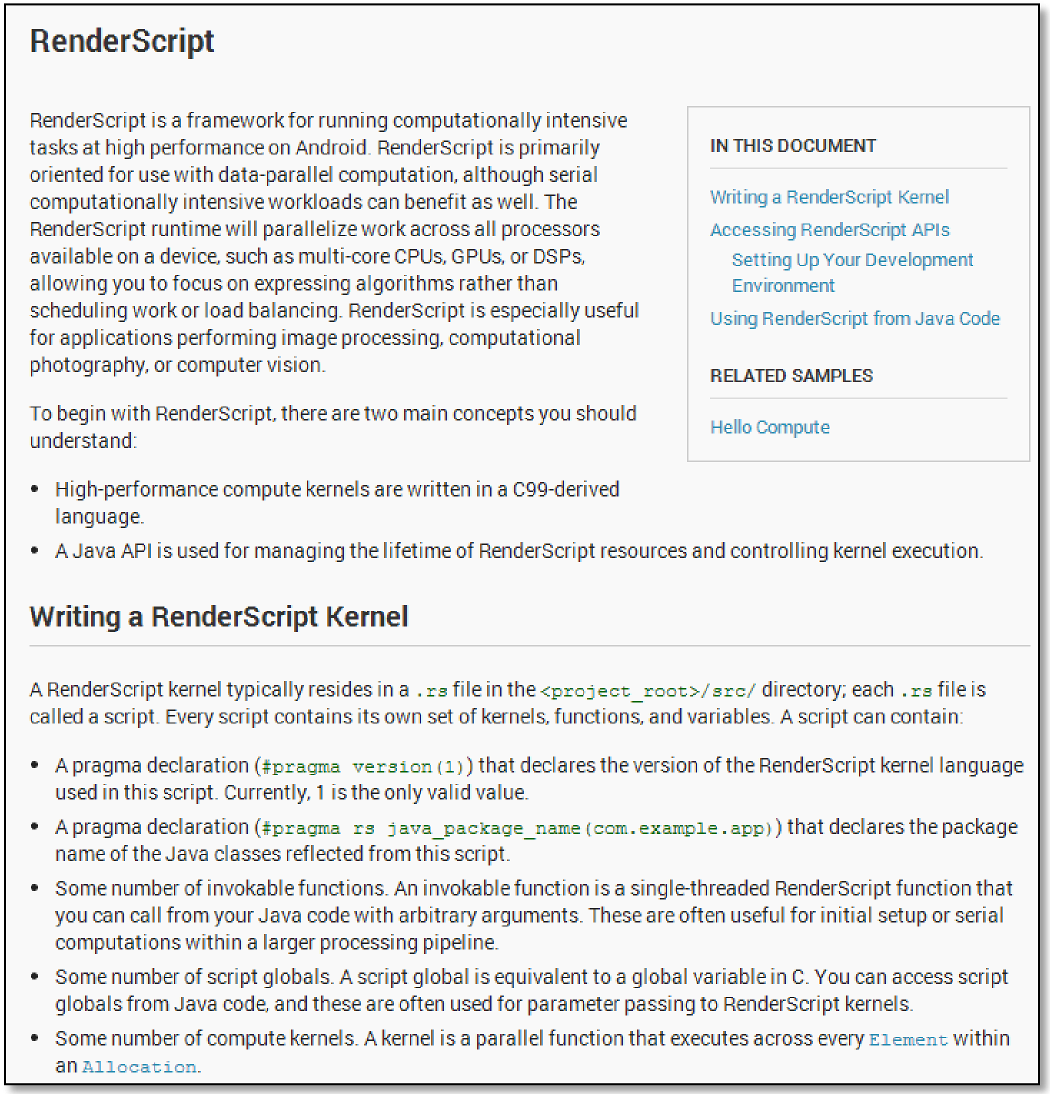
# Summary

- Android design constraints affect concurrent program development
- Android's common concurrency frameworks simplify concurrent programs
- Both frameworks are designed using *patterns* & yield *idioms*
- Several other concurrency frameworks are also available



# Summary

- Android design constraints affect concurrent program development
- Android's common concurrency frameworks simplify concurrent programs
- Both frameworks are designed using *patterns* & yield *idioms*
- Several other concurrency frameworks are also available
  - **RenderScript**
    - A framework for running computationally intensive tasks across all processors on a device



## RenderScript

RenderScript is a framework for running computationally intensive tasks at high performance on Android. RenderScript is primarily oriented for use with data-parallel computation, although serial computationally intensive workloads can benefit as well. The RenderScript runtime will parallelize work across all processors available on a device, such as multi-core CPUs, GPUs, or DSPs, allowing you to focus on expressing algorithms rather than scheduling work or load balancing. RenderScript is especially useful for applications performing image processing, computational photography, or computer vision.

To begin with RenderScript, there are two main concepts you should understand:

- High-performance compute kernels are written in a C99-derived language.
- A Java API is used for managing the lifetime of RenderScript resources and controlling kernel execution.

IN THIS DOCUMENT

Writing a RenderScript Kernel
Accessing RenderScript APIs
Setting Up Your Development Environment
Using RenderScript from Java Code

RELATED SAMPLES

Hello Compute

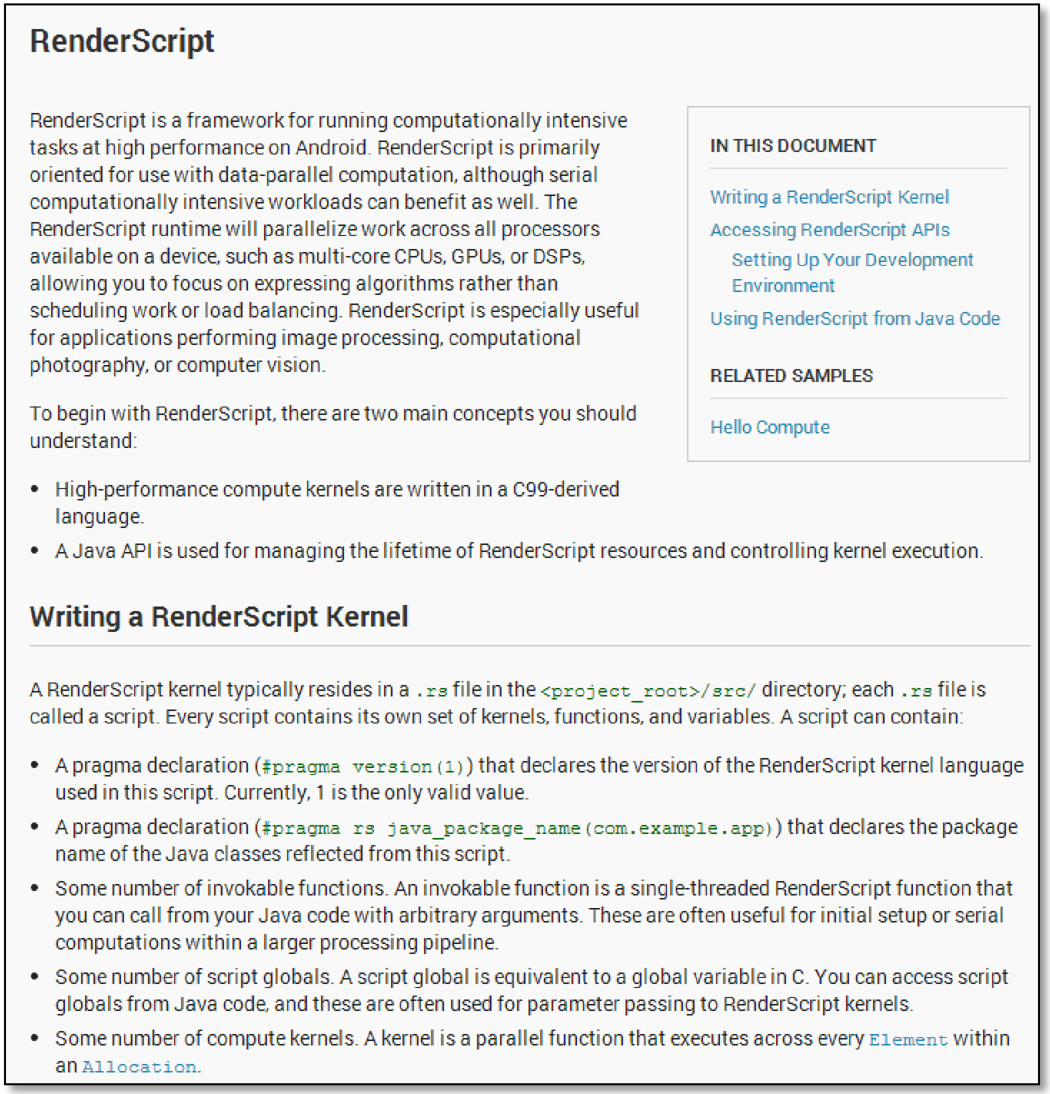
## Writing a RenderScript Kernel

A RenderScript kernel typically resides in a `.rs` file in the `<project_root>/src/` directory; each `.rs` file is called a script. Every script contains its own set of kernels, functions, and variables. A script can contain:

- A pragma declaration (`#pragma version(1)`) that declares the version of the RenderScript kernel language used in this script. Currently, 1 is the only valid value.
- A pragma declaration (`#pragma rs java_package_name(com.example.app)`) that declares the package name of the Java classes reflected from this script.
- Some number of invokable functions. An invokable function is a single-threaded RenderScript function that you can call from your Java code with arbitrary arguments. These are often useful for initial setup or serial computations within a larger processing pipeline.
- Some number of script globals. A script global is equivalent to a global variable in C. You can access script globals from Java code, and these are often used for parameter passing to RenderScript kernels.
- Some number of compute kernels. A kernel is a parallel function that executes across every `Element` within an `Allocation`.

# Summary

- Android design constraints affect concurrent program development
- Android's common concurrency frameworks simplify concurrent programs
- Both frameworks are designed using *patterns* & yield *idioms*
- Several other concurrency frameworks are also available
  - **RenderScript**
    - A framework for running computationally intensive tasks across all processors on a device
      - e.g., CPUs, GPUs, & DSPs



## RenderScript

RenderScript is a framework for running computationally intensive tasks at high performance on Android. RenderScript is primarily oriented for use with data-parallel computation, although serial computationally intensive workloads can benefit as well. The RenderScript runtime will parallelize work across all processors available on a device, such as multi-core CPUs, GPUs, or DSPs, allowing you to focus on expressing algorithms rather than scheduling work or load balancing. RenderScript is especially useful for applications performing image processing, computational photography, or computer vision.

To begin with RenderScript, there are two main concepts you should understand:

- High-performance compute kernels are written in a C99-derived language.
- A Java API is used for managing the lifetime of RenderScript resources and controlling kernel execution.

IN THIS DOCUMENT

Writing a RenderScript Kernel
Accessing RenderScript APIs
Setting Up Your Development Environment
Using RenderScript from Java Code

RELATED SAMPLES

Hello Compute

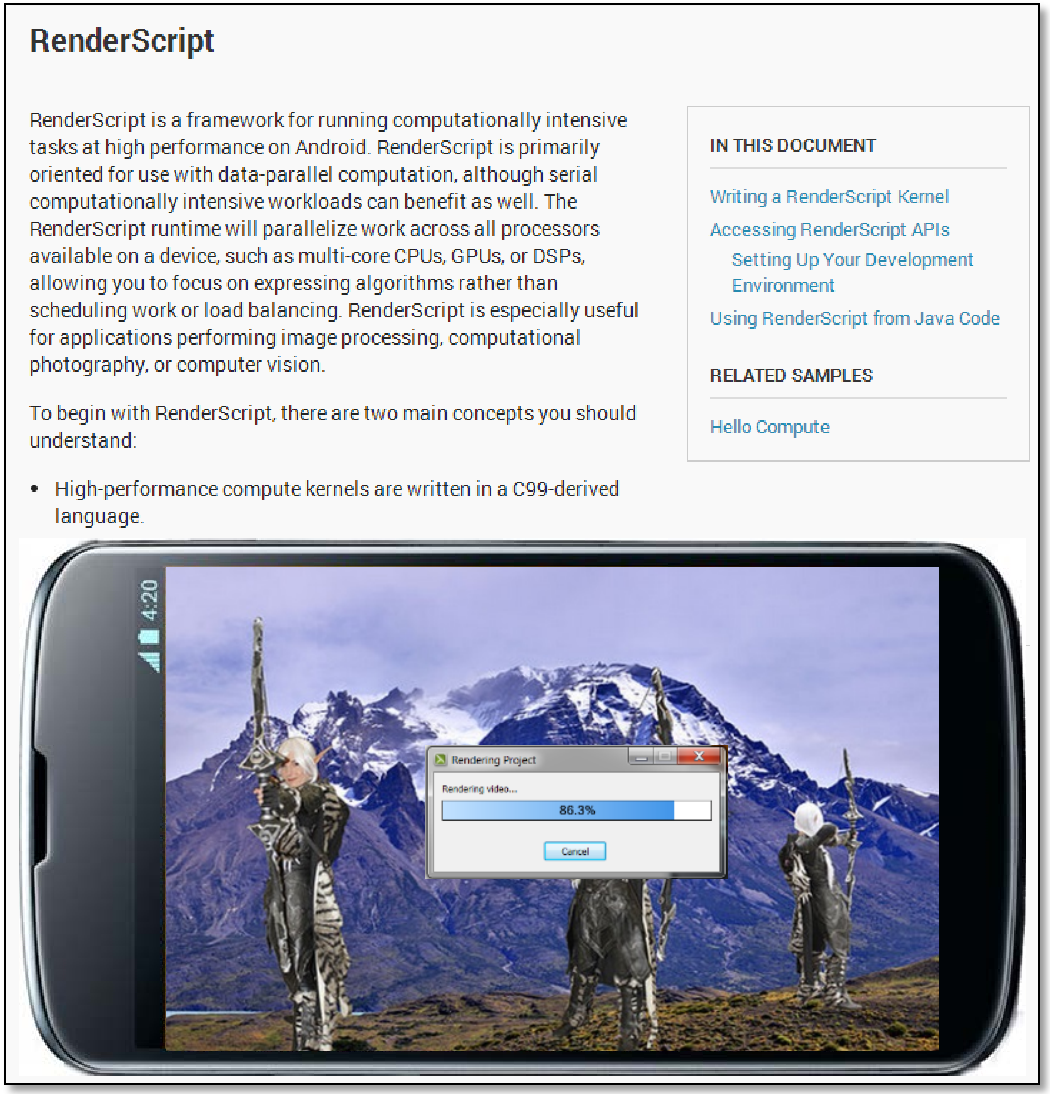
## Writing a RenderScript Kernel

A RenderScript kernel typically resides in a `.rs` file in the `<project_root>/src/` directory; each `.rs` file is called a script. Every script contains its own set of kernels, functions, and variables. A script can contain:

- A pragma declaration (`#pragma version(1)`) that declares the version of the RenderScript kernel language used in this script. Currently, 1 is the only valid value.
- A pragma declaration (`#pragma rs java_package_name(com.example.app)`) that declares the package name of the Java classes reflected from this script.
- Some number of invokable functions. An invokable function is a single-threaded RenderScript function that you can call from your Java code with arbitrary arguments. These are often useful for initial setup or serial computations within a larger processing pipeline.
- Some number of script globals. A script global is equivalent to a global variable in C. You can access script globals from Java code, and these are often used for parameter passing to RenderScript kernels.
- Some number of compute kernels. A kernel is a parallel function that executes across every `Element` within an `Allocation`.

# Summary

- Android design constraints affect concurrent program development
- Android's common concurrency frameworks simplify concurrent programs
- Both frameworks are designed using *patterns* & yield *idioms*
- Several other concurrency frameworks are also available
  - **RenderScript**
    - A framework for running computationally intensive tasks across all processors on a device
      - e.g., CPUs, GPUs, & DSPs



## RenderScript

RenderScript is a framework for running computationally intensive tasks at high performance on Android. RenderScript is primarily oriented for use with data-parallel computation, although serial computationally intensive workloads can benefit as well. The RenderScript runtime will parallelize work across all processors available on a device, such as multi-core CPUs, GPUs, or DSPs, allowing you to focus on expressing algorithms rather than scheduling work or load balancing. RenderScript is especially useful for applications performing image processing, computational photography, or computer vision.

To begin with RenderScript, there are two main concepts you should understand:

- High-performance compute kernels are written in a C99-derived language.

IN THIS DOCUMENT

Writing a RenderScript Kernel
Accessing RenderScript APIs
Setting Up Your Development Environment
Using RenderScript from Java Code

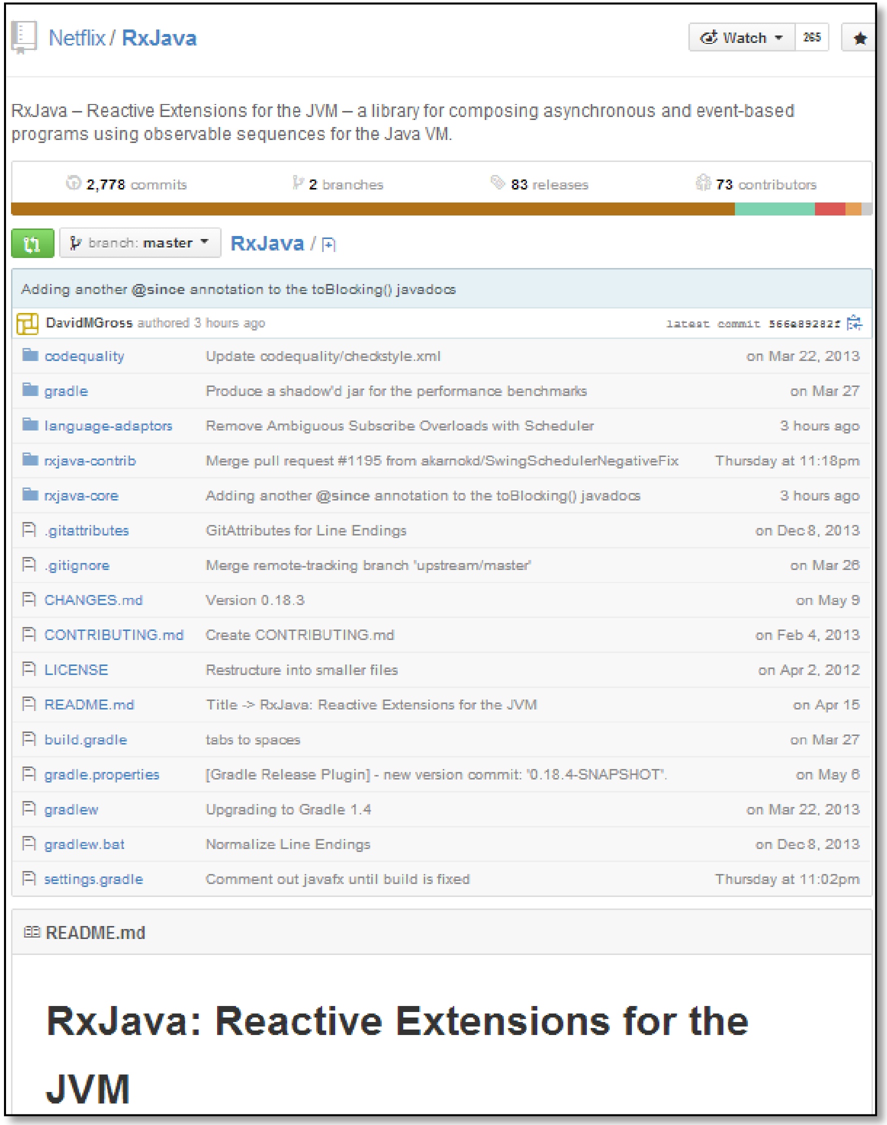
RELATED SAMPLES

Hello Compute

See developer.android.com/guide/topics/renderscript/compute.html

# Summary

- Android design constraints affect concurrent program development
- Android's common concurrency frameworks simplify concurrent programs
- Both frameworks are designed using *patterns* & yield *idioms*
- Several other concurrency frameworks are also available
  - **RenderScript**
  - **RxJava**
    - A library for composing asynchronous & event-based programs using observable sequences for the JVM



Netflix / **RxJava**

RxJava – Reactive Extensions for the JVM – a library for composing asynchronous and event-based programs using observable sequences for the Java VM.

**2,778** commits · **2** branches · **83** releases · **73** contributors

Adding another @since annotation to the toBlocking() javadocs

DavidMGross authored 3 hours ago — latest commit 566a89282f

| | | |
|---|---|---|
| codequality | Update codequality/checkstyle.xml | on Mar 22, 2013 |
| gradle | Produce a shadow'd jar for the performance benchmarks | on Mar 27 |
| language-adaptors | Remove Ambiguous Subscribe Overloads with Scheduler | 3 hours ago |
| rxjava-contrib | Merge pull request #1195 from akarnokd/SwingSchedulerNegativeFix | Thursday at 11:18pm |
| rxjava-core | Adding another @since annotation to the toBlocking() javadocs | 3 hours ago |
| .gitattributes | GitAttributes for Line Endings | on Dec 8, 2013 |
| .gitignore | Merge remote-tracking branch 'upstream/master' | on Mar 26 |
| CHANGES.md | Version 0.18.3 | on May 9 |
| CONTRIBUTING.md | Create CONTRIBUTING.md | on Feb 4, 2013 |
| LICENSE | Restructure into smaller files | on Apr 2, 2012 |
| README.md | Title -> RxJava: Reactive Extensions for the JVM | on Apr 15 |
| build.gradle | tabs to spaces | on Mar 27 |
| gradle.properties | [Gradle Release Plugin] - new version commit: '0.18.4-SNAPSHOT'. | on May 6 |
| gradlew | Upgrading to Gradle 1.4 | on Mar 22, 2013 |
| gradlew.bat | Normalize Line Endings | on Dec 8, 2013 |
| settings.gradle | Comment out javafx until build is fixed | Thursday at 11:02pm |

README.md

**RxJava: Reactive Extensions for the JVM**

See github.com/Netflix/RxJava for source code

# Summary

- Android design constraints affect concurrent program development
- Android's common concurrency frameworks simplify concurrent programs
- Both frameworks are designed using *patterns* & yield *idioms*
- Several other concurrency frameworks are also available
    - **RenderScript**
    - **RxJava**
        - A library for composing asynchronous & event-based programs using observable sequences for the JVM

```
Observable<File> downloadFileObs() {
  return Observable.create(new
              OnSubscribeFunc<File>() {
    public Subscription onSubscribe
               (Observer<? super File>
                  fileObserver) {
      try {
        byte[] fileContent =
          downloadFile();
        File file =
          writeToFile(fileContent);
        fileObserver.onNext(file);
        fileObserver.onCompleted();
      } catch (Exception e) {
          fileObserver.onError(e);
      }
      return Subscriptions.empty();
    }});
}
```

mttkay.github.io/blog/2013/08/25/functional-reactive-programming-on-android-with-rxjava

# Summary

- Android design constraints affect concurrent program development
- Android’s common concurrency frameworks simplify concurrent programs
- Both frameworks are designed using *patterns* & yield *idioms*
- Several other concurrency frameworks are also available
  - **RenderScript**
  - **RxJava**

These concurrency frameworks are interesting, but beyond scope of this class