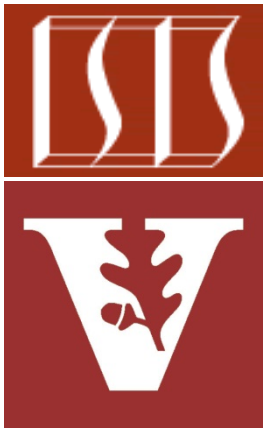
Android Concurrency: Overview of Android Handler

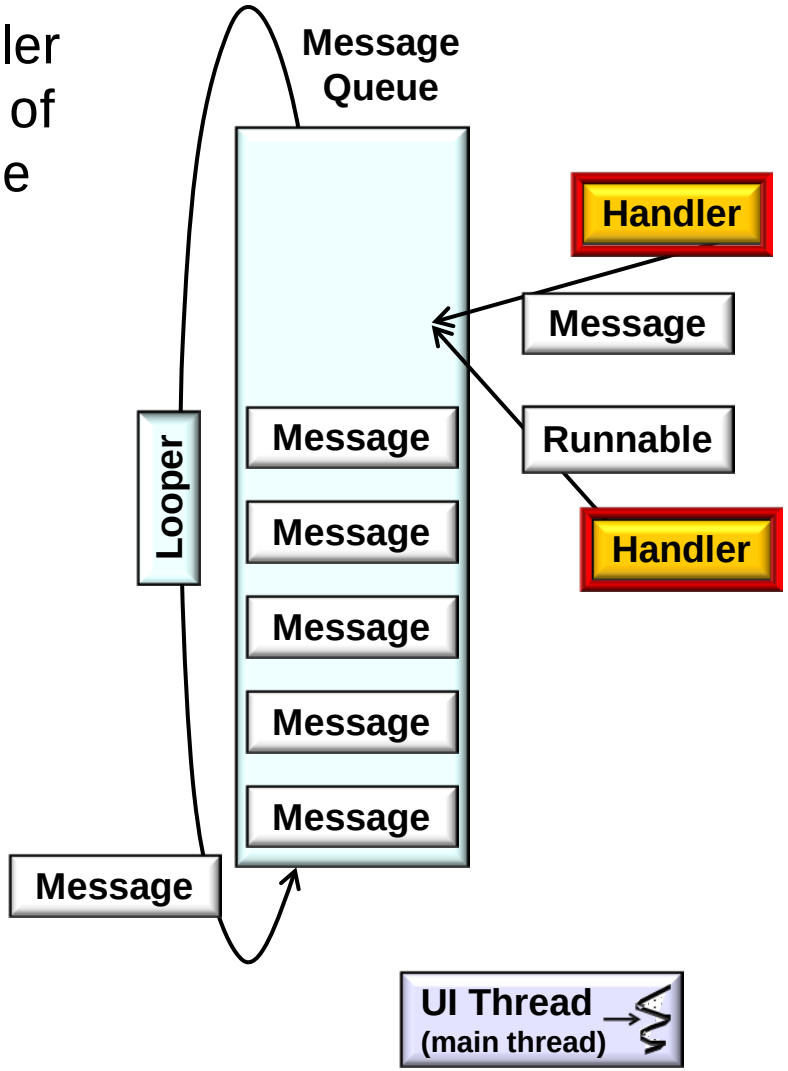


Douglas C. Schmidt
d.schmidt@vanderbilt.edu
www.dre.vanderbilt.edu/~schmidt

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

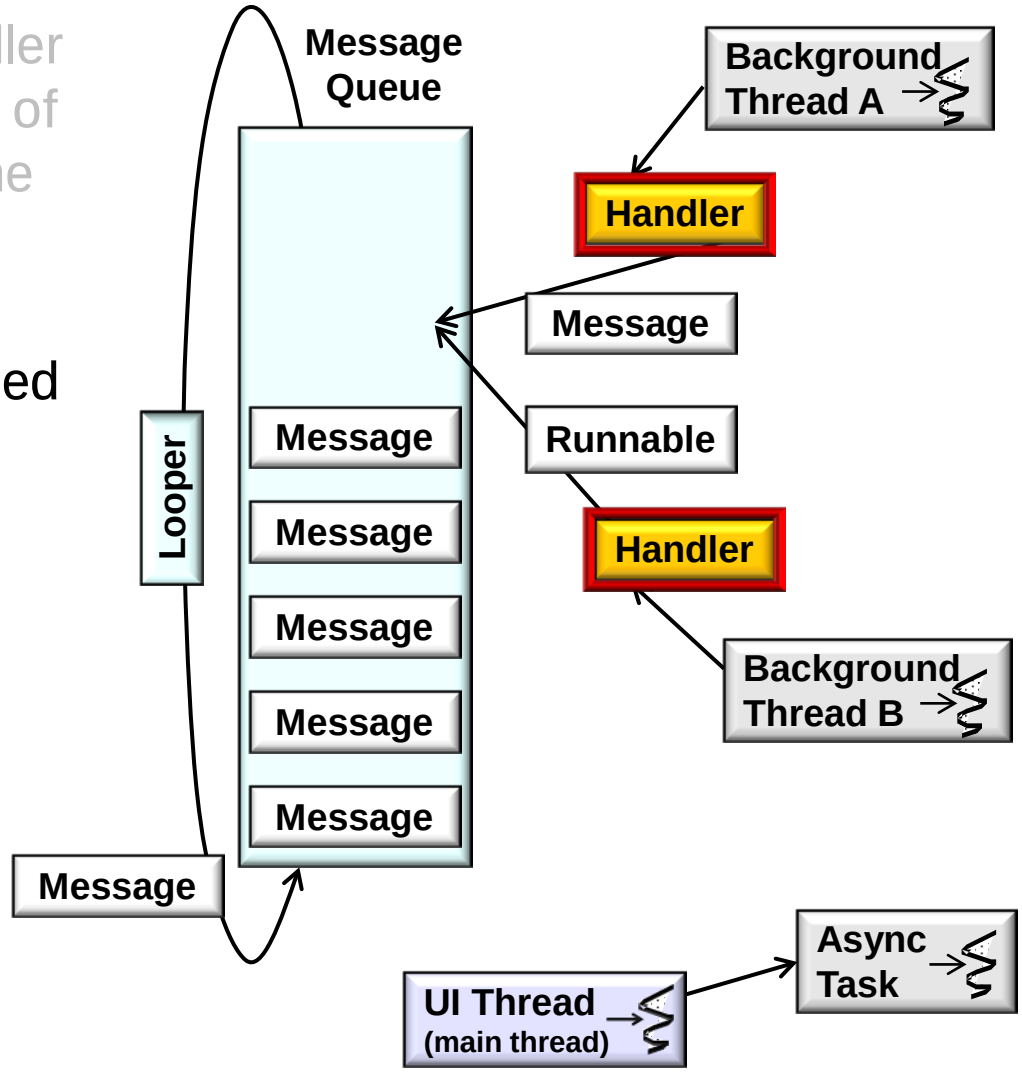
Learning Objectives in this Part of the Module

- Understand how an Android Handler enables the sending & processing of Message & Runnable objects in the MessageQueue associated with a Thread's Looper



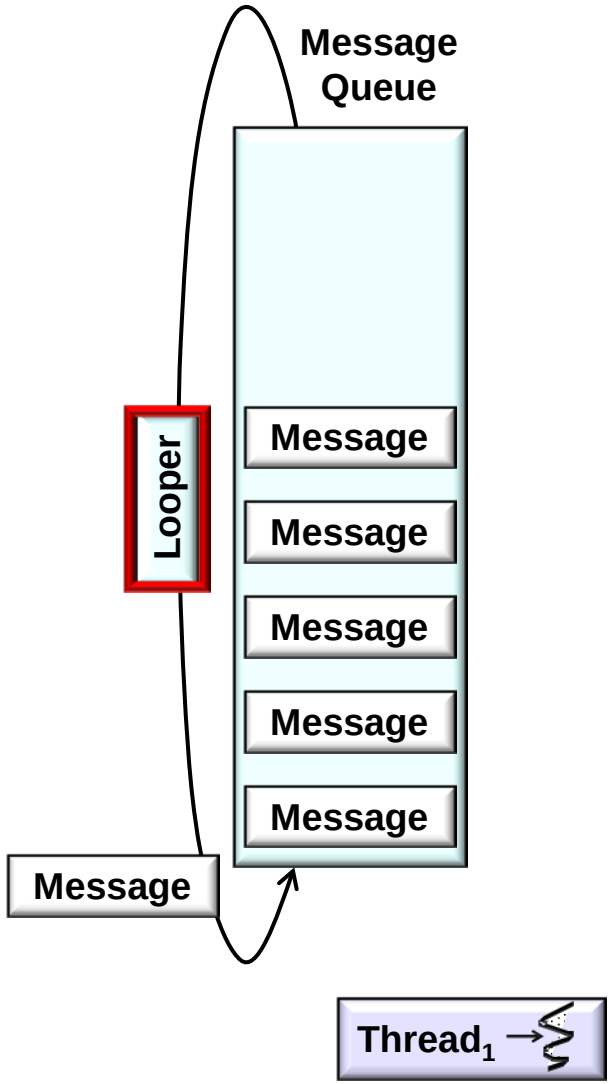
Learning Objectives in this Part of the Module

- Understand how an Android Handler enables the sending & processing of Message & Runnable objects in the MessageQueue associated with a Thread's Looper
- Recognize how Handlers are applied in Android applications & concurrency frameworks



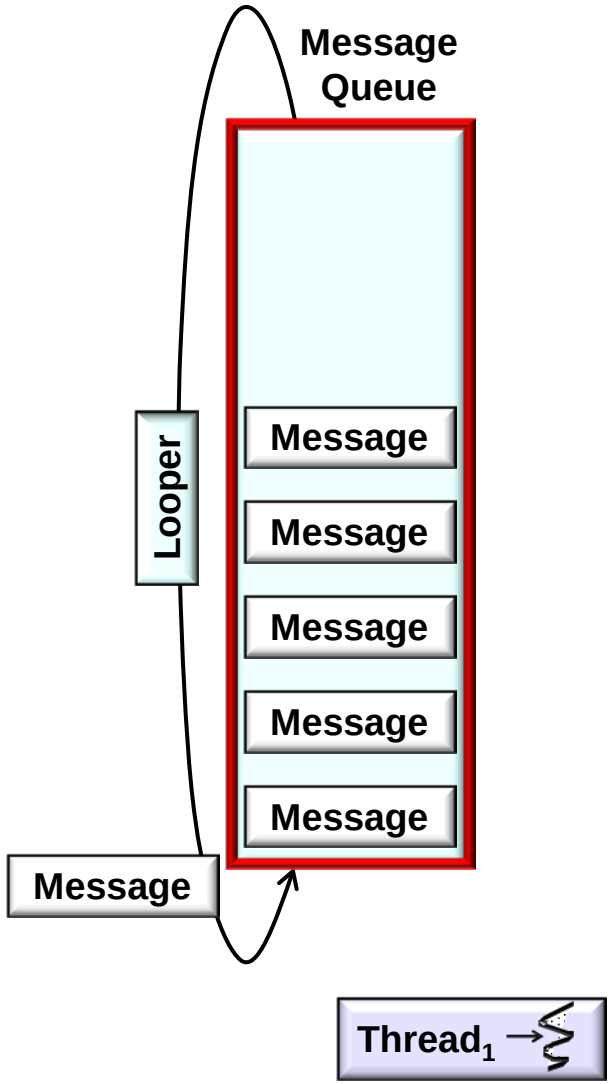
The Handler Class

- A Looper has a MessageQueue



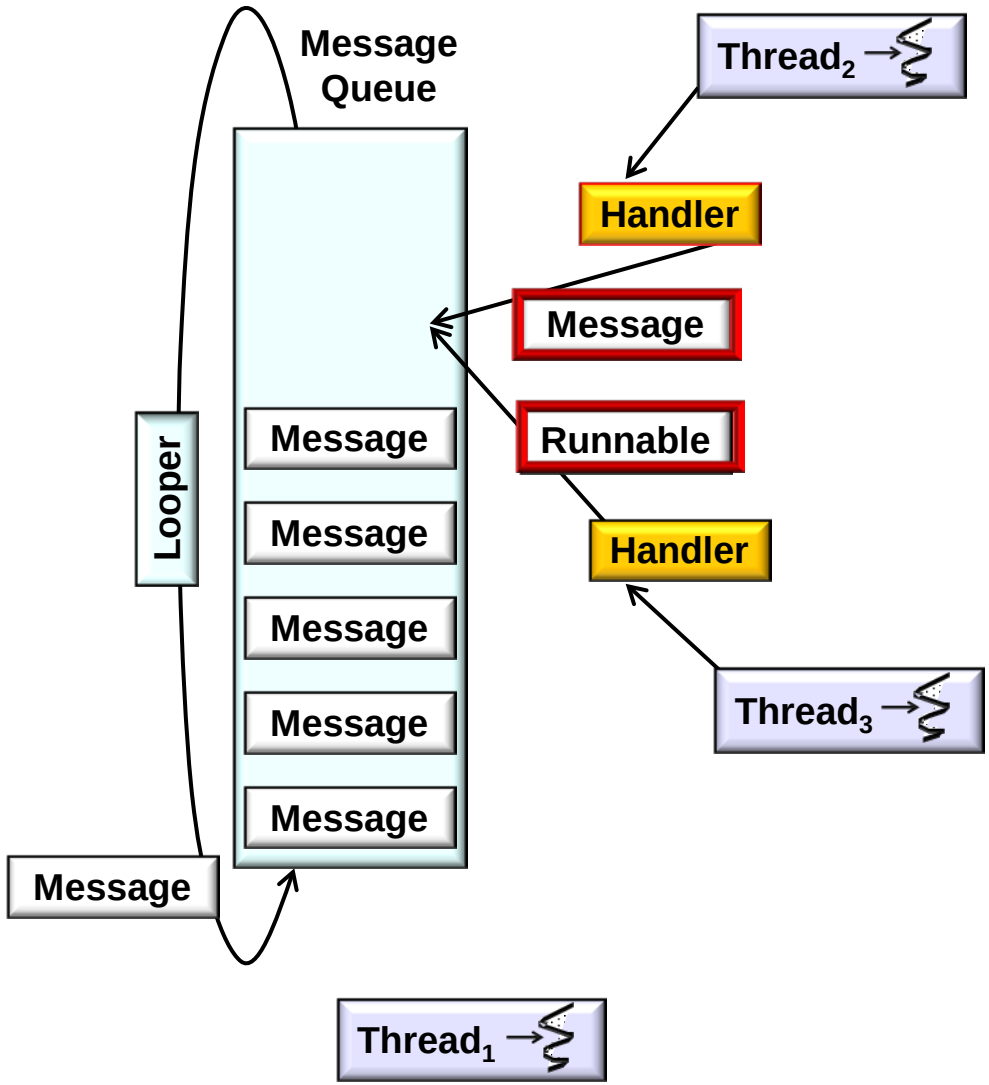
The Handler Class

- A Looper has a MessageQueue



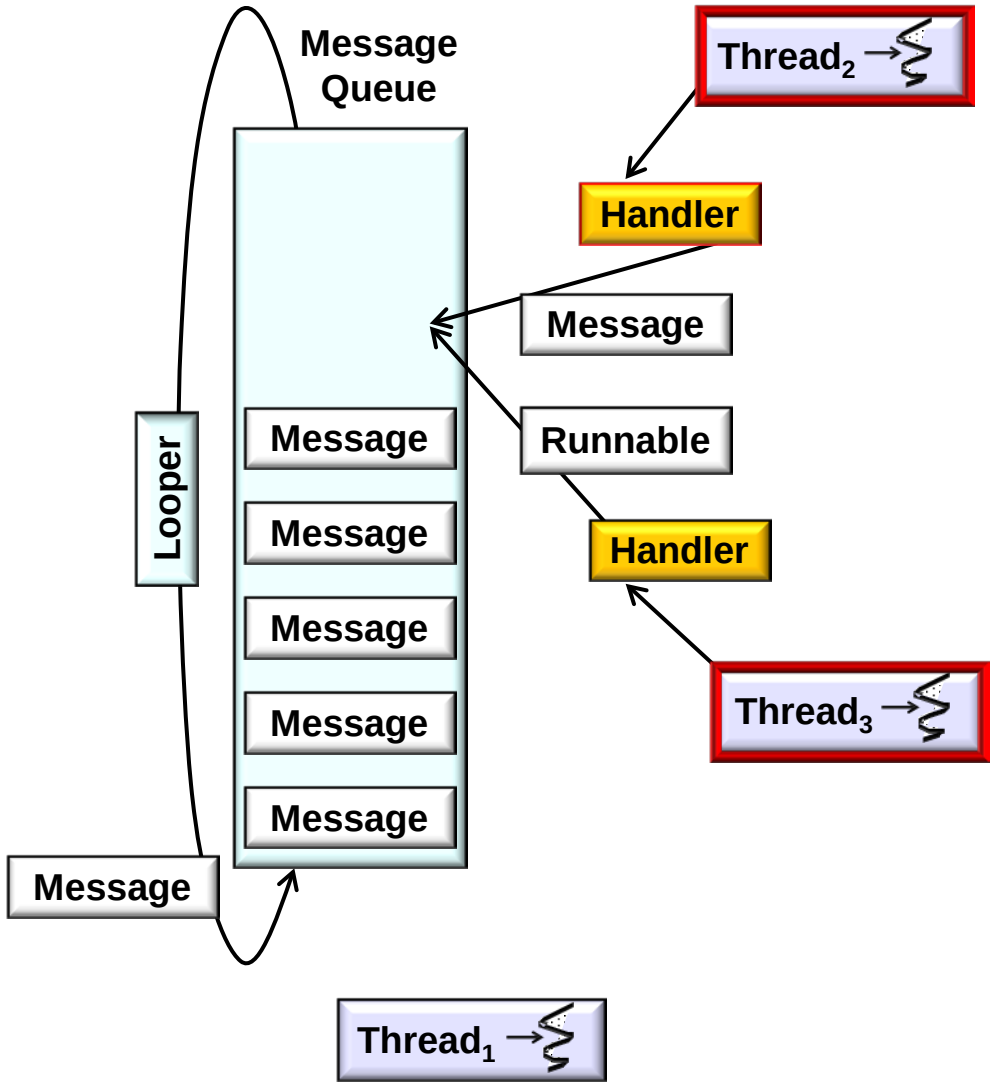
The Handler Class

- A Looper has a MessageQueue



The Handler Class

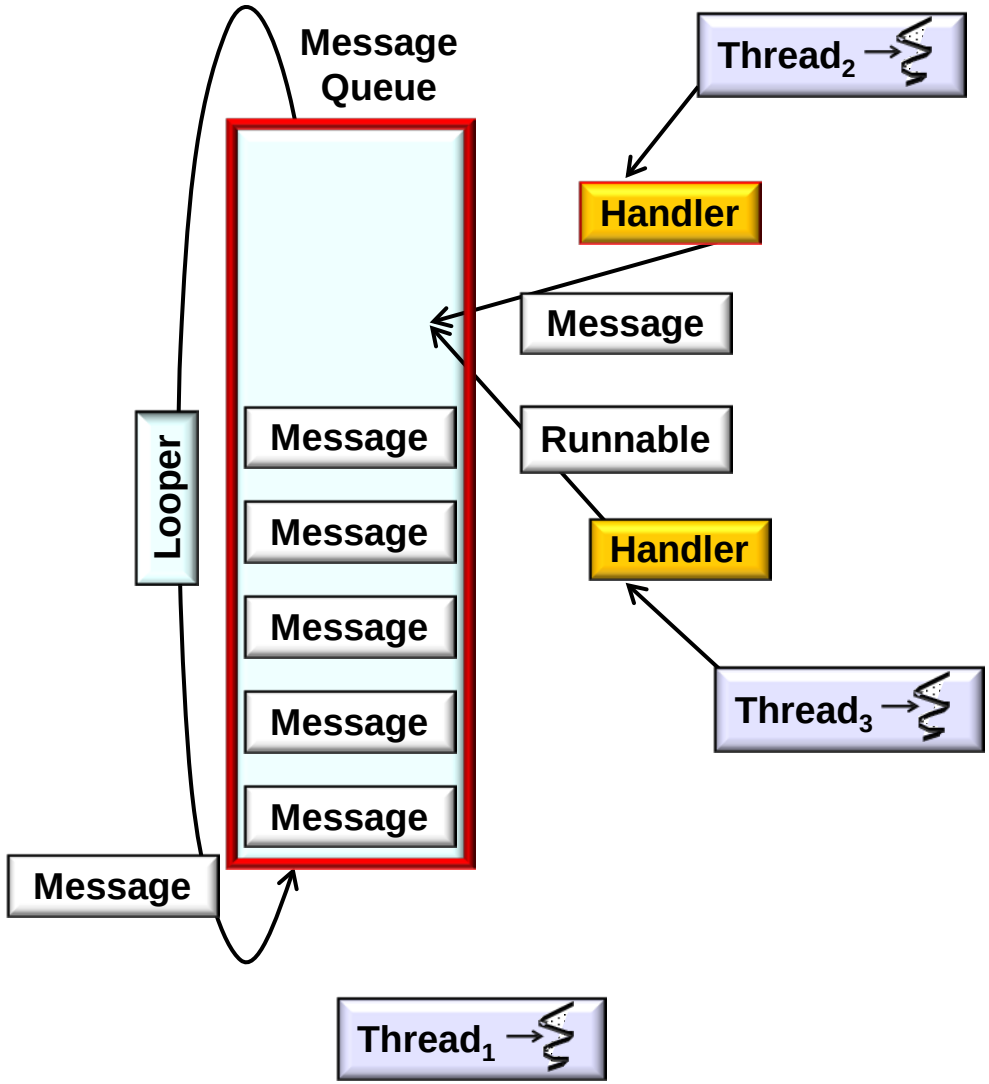
- A Looper has a MessageQueue



See previous part on “Android Looper”

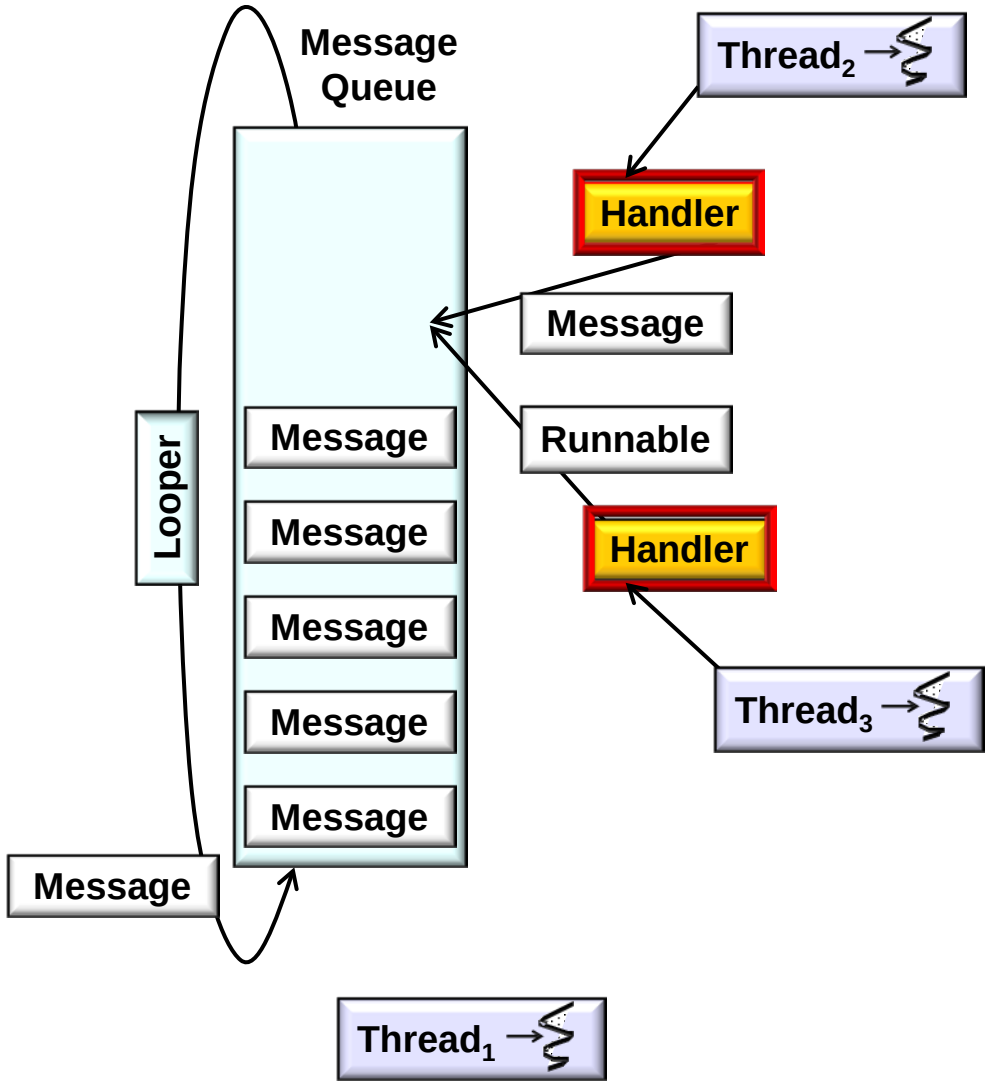
The Handler Class

- A Looper has a MessageQueue
  - The actual management of the MessageQueue is done by instances of Handlers



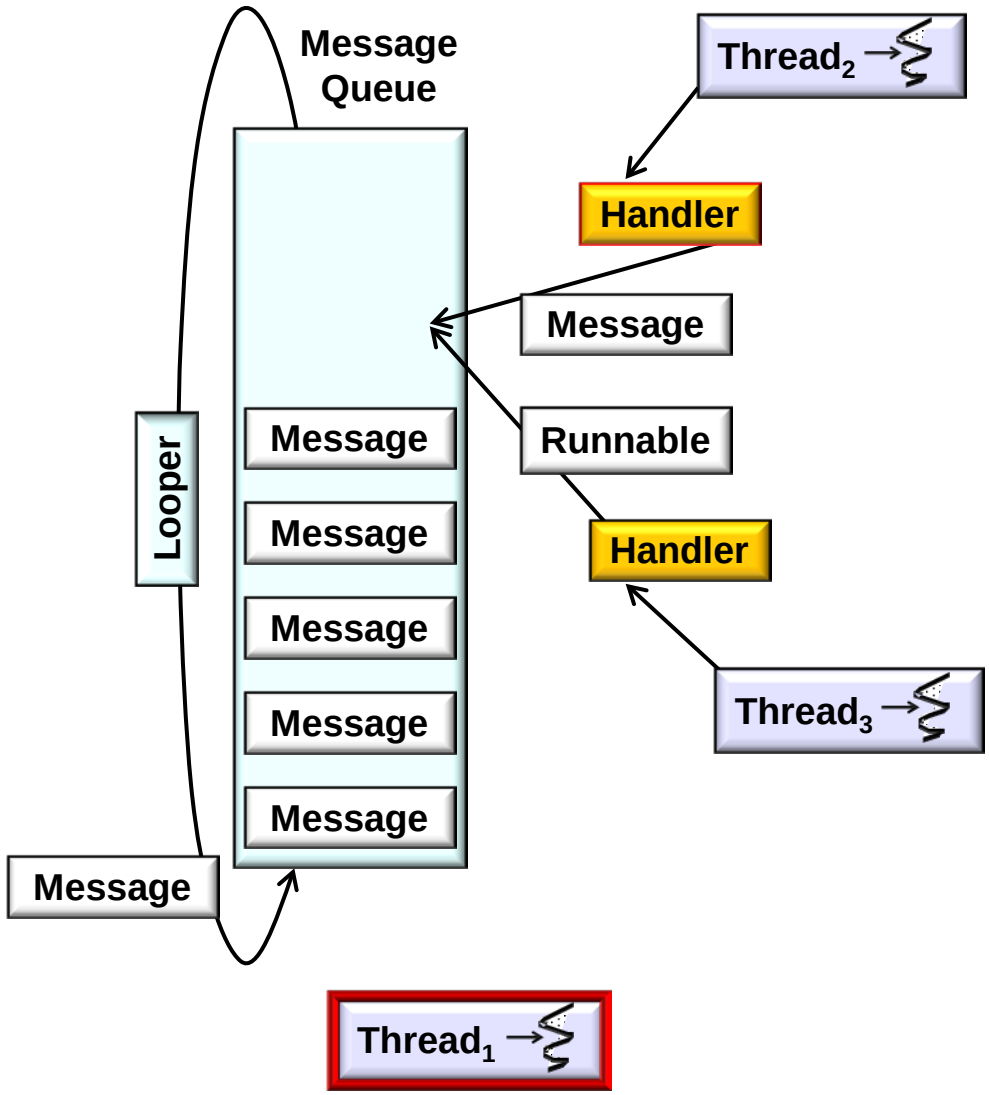
The Handler Class

- A Looper has a MessageQueue
  - The actual management of the MessageQueue is done by instances of Handlers



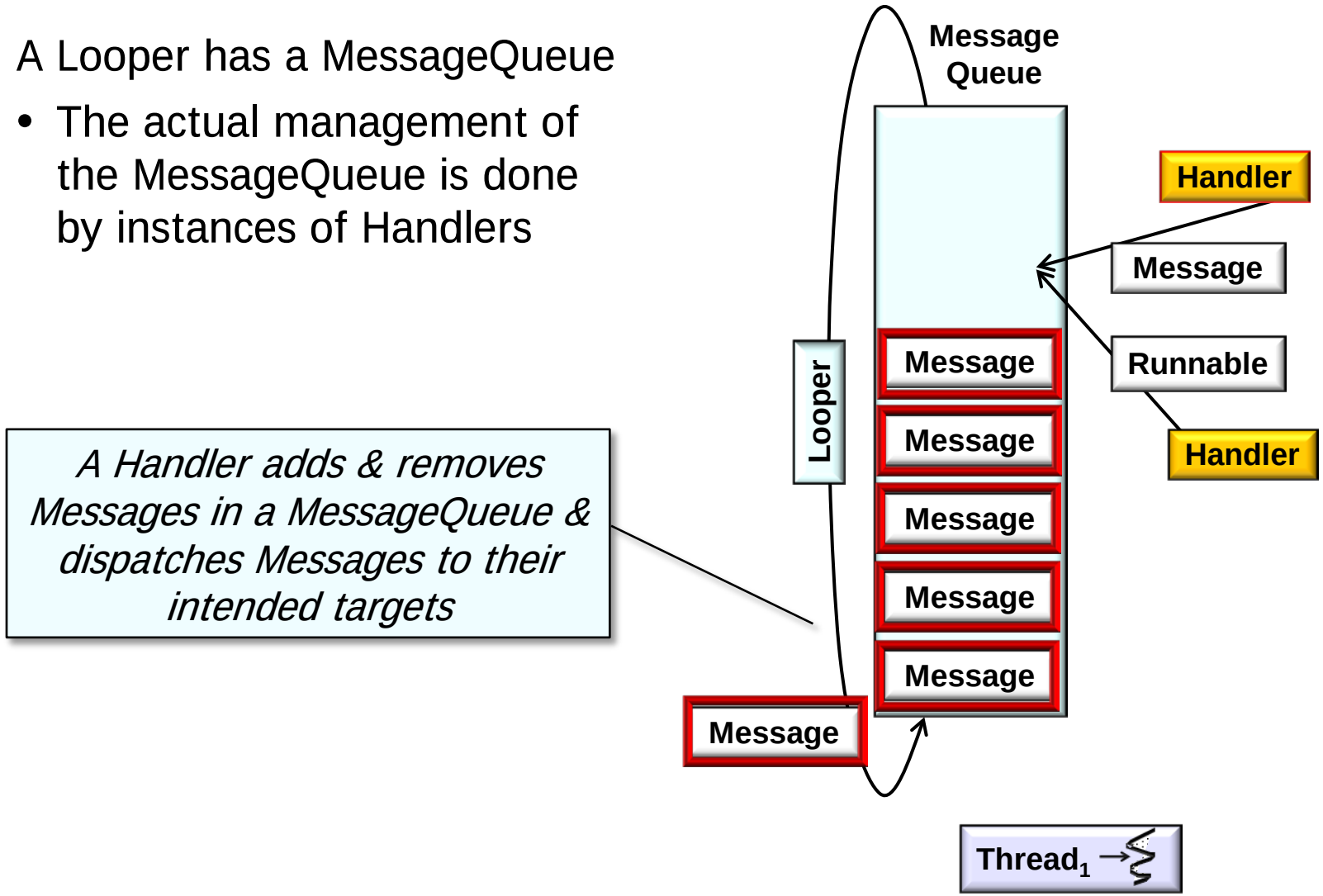
The Handler Class

- A Looper has a MessageQueue
  - The actual management of the MessageQueue is done by instances of Handlers



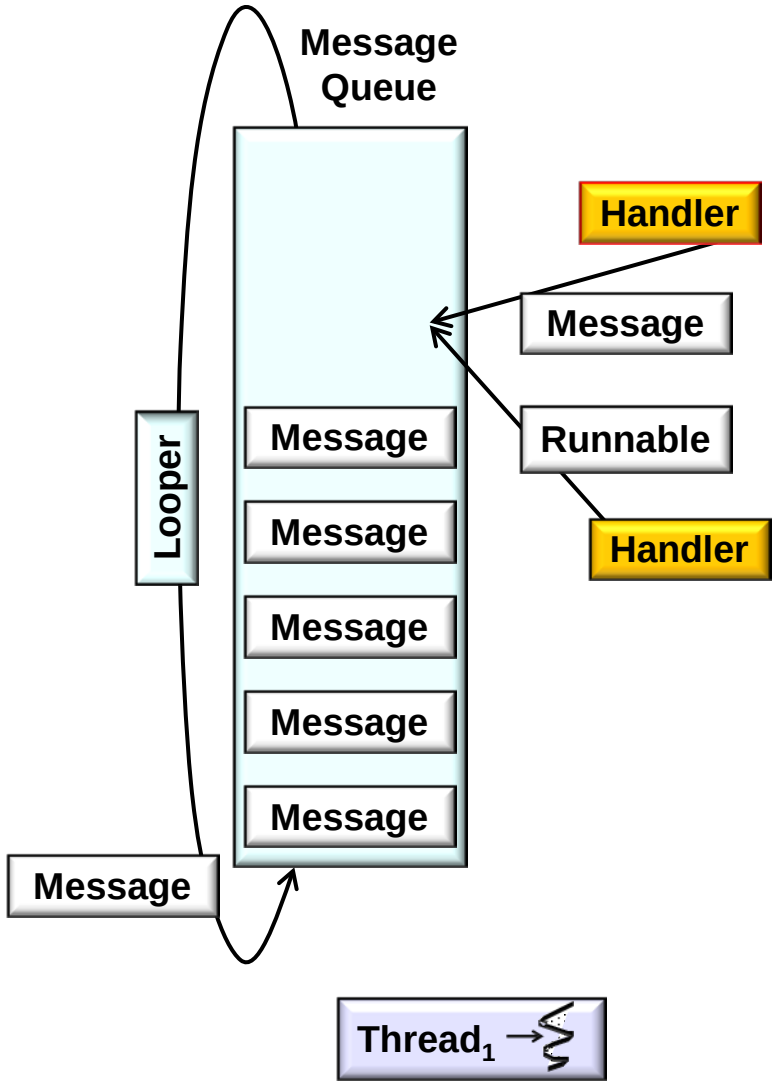
The Handler Class

- A Looper has a MessageQueue
  - The actual management of the MessageQueue is done by instances of Handlers



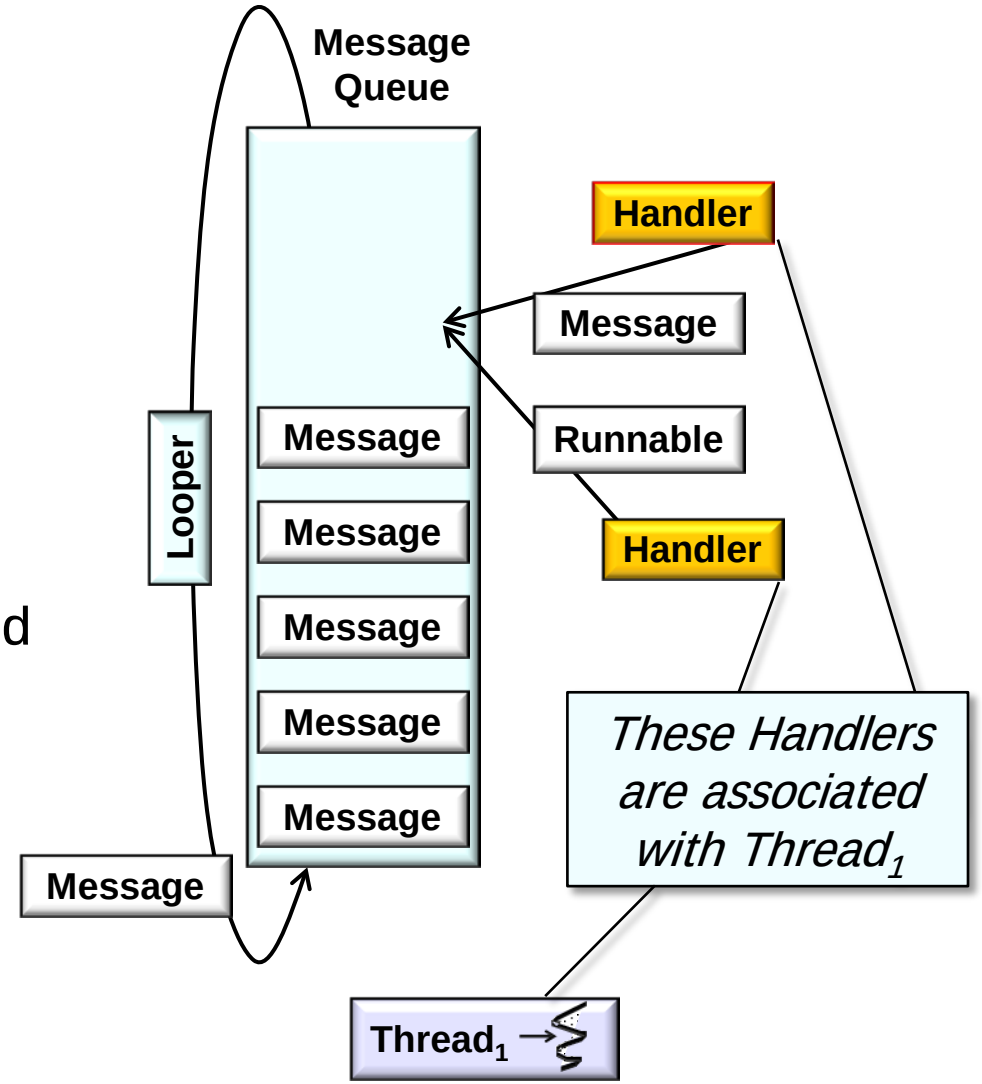
The Handler Class

- A Looper has a MessageQueue
  - The actual management of the MessageQueue is done by instances of Handlers
  - A Handler is associated with a particular Looper



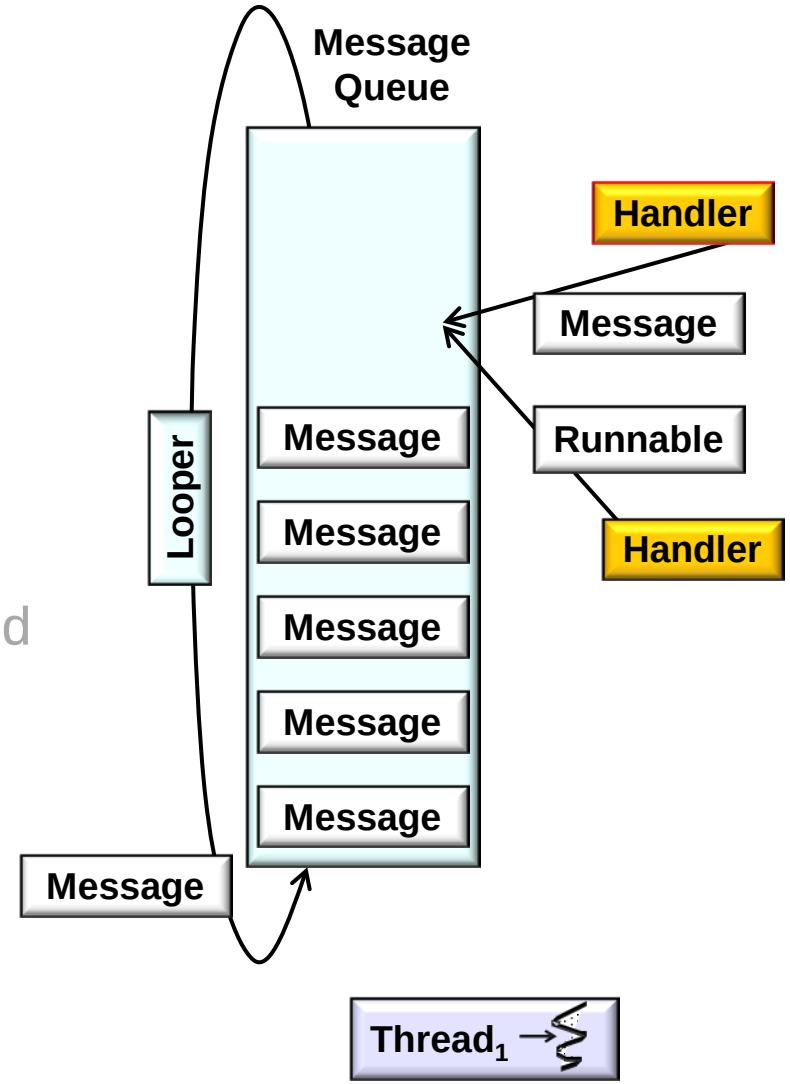
The Handler Class

- A Looper has a MessageQueue
  - The actual management of the MessageQueue is done by instances of Handlers
  - A Handler is associated with a particular Looper
    - Defaults to the Looper in which the current Thread in which the Handler was created



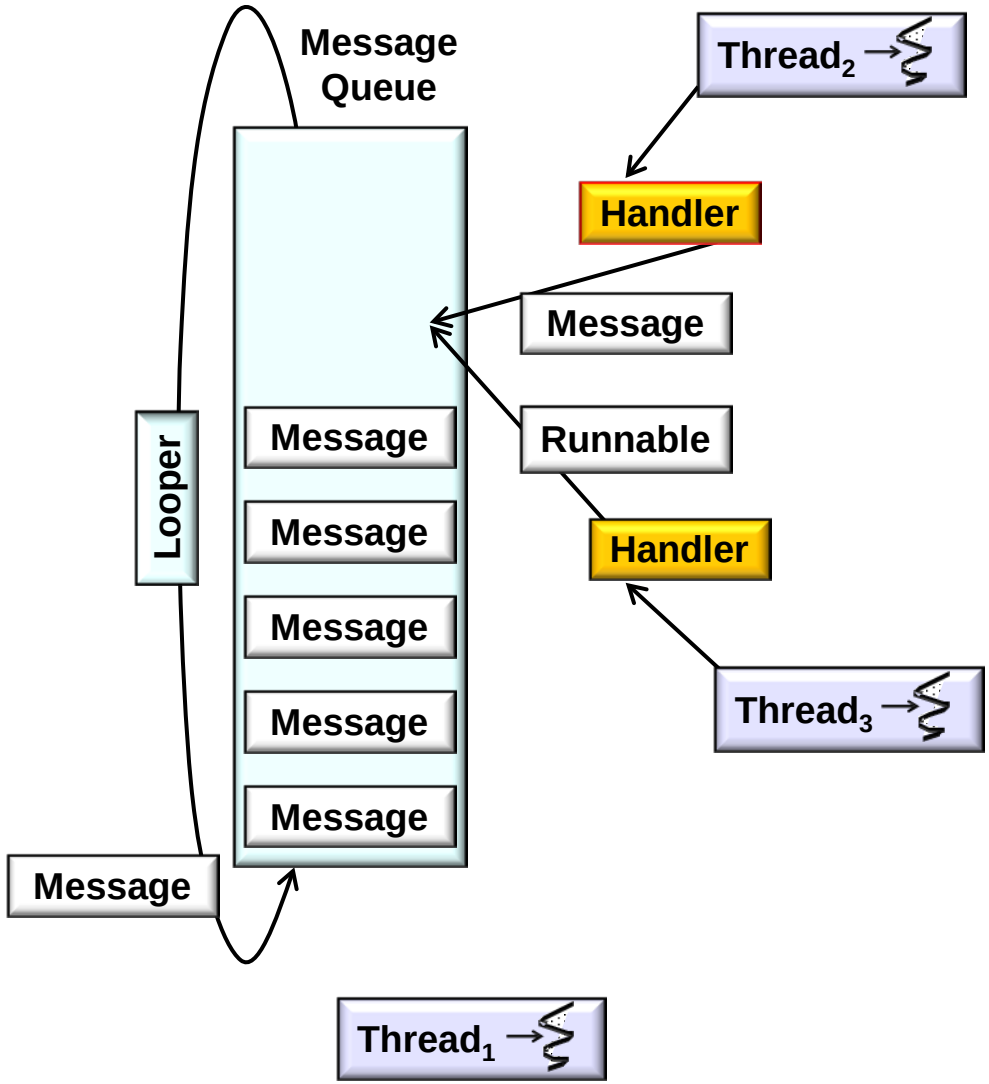
The Handler Class

- A Looper has a MessageQueue
  - The actual management of the MessageQueue is done by instances of Handlers
  - A Handler is associated with a particular Looper
    - Defaults to the Looper in which the current Thread in which the Handler was created
    - A different Looper can be passed as a parameter to the constructor



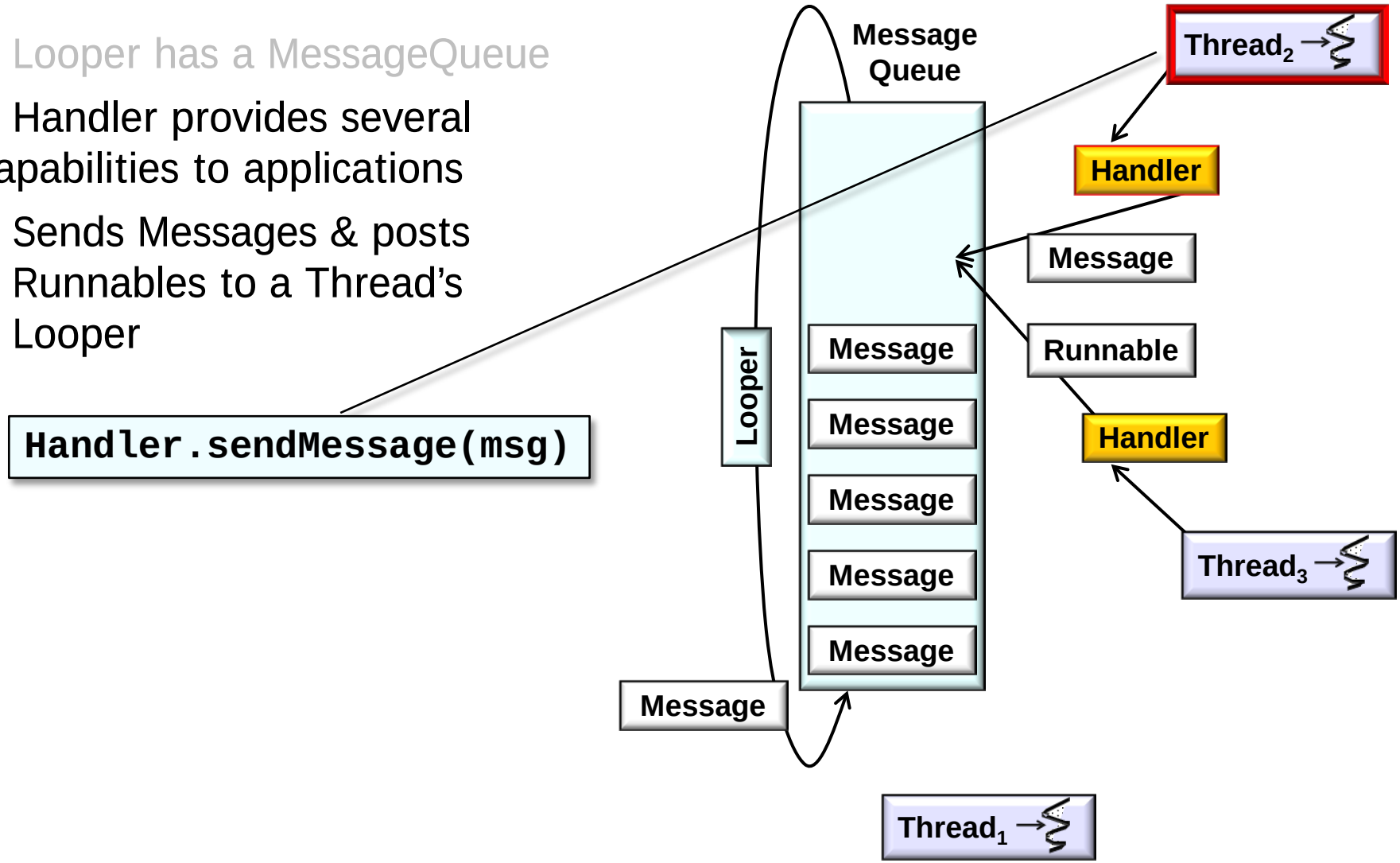
The Handler Class

- A Looper has a MessageQueue
- A Handler provides several capabilities to applications



The Handler Class

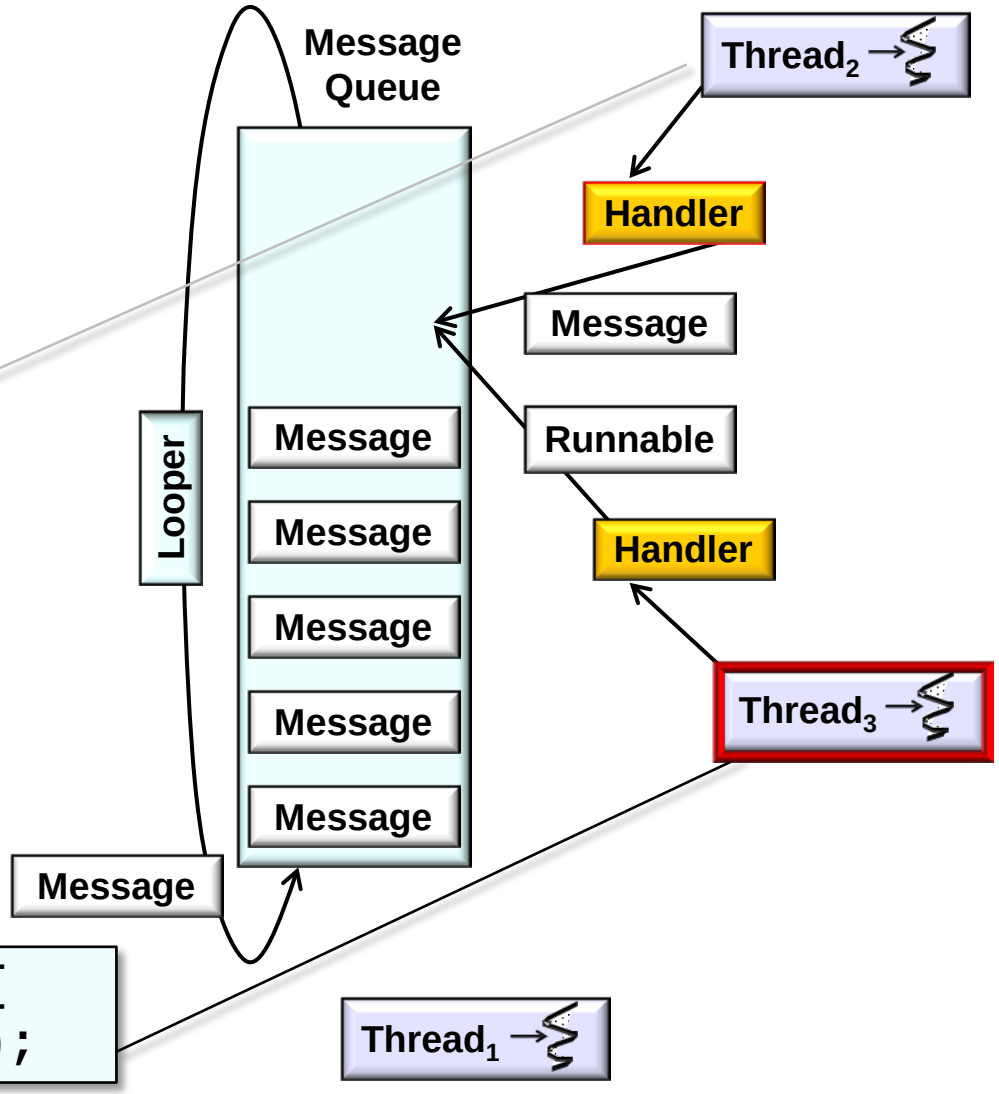
- A Looper has a MessageQueue
- A Handler provides several capabilities to applications
  - Sends Messages & posts Runnables to a Thread's Looper



The Handler Class

- A Looper has a MessageQueue
- A Handler provides several capabilities to applications
  - Sends Messages & posts Runnables to a Thread's Looper

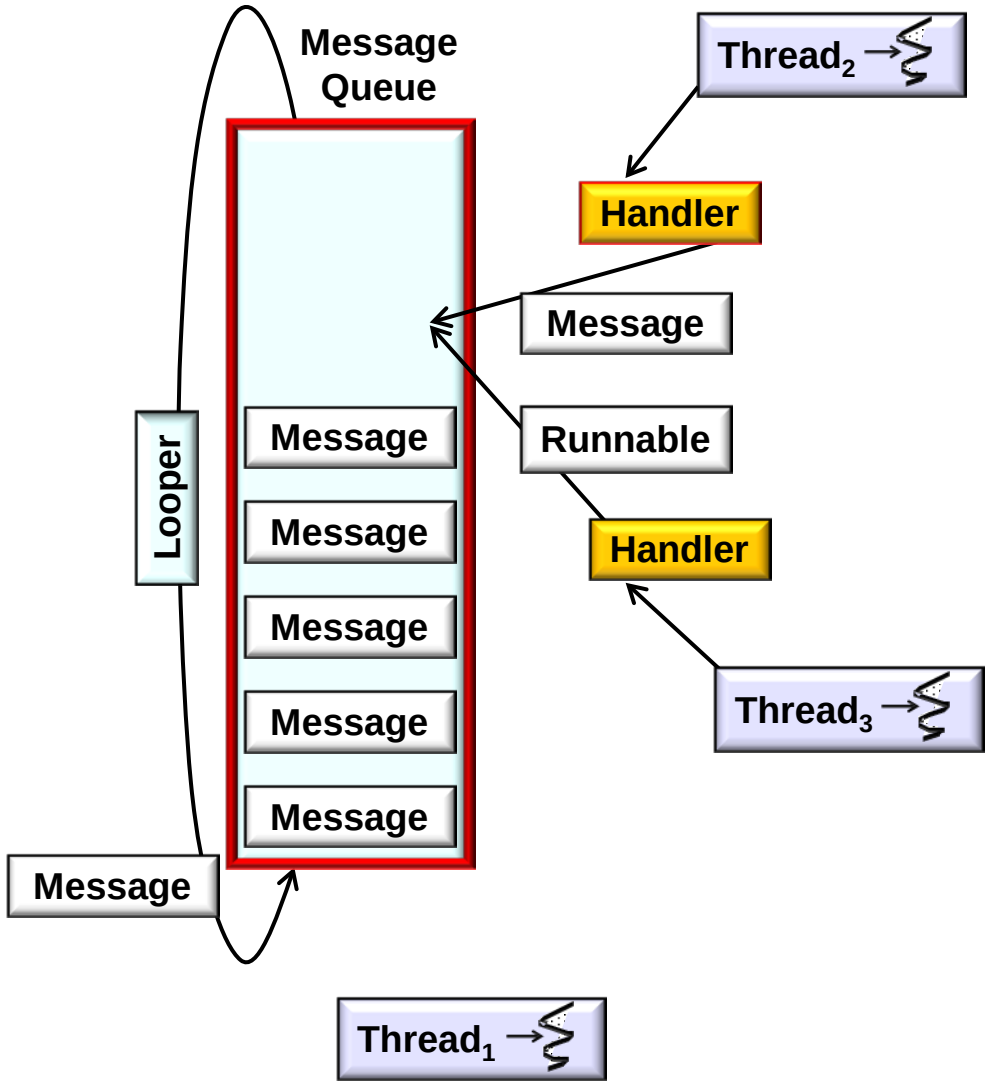
```
Handler.sendMessage(msg)
```




```
Handler.post(new Runnable(){
  public void run(){ ... }});
```

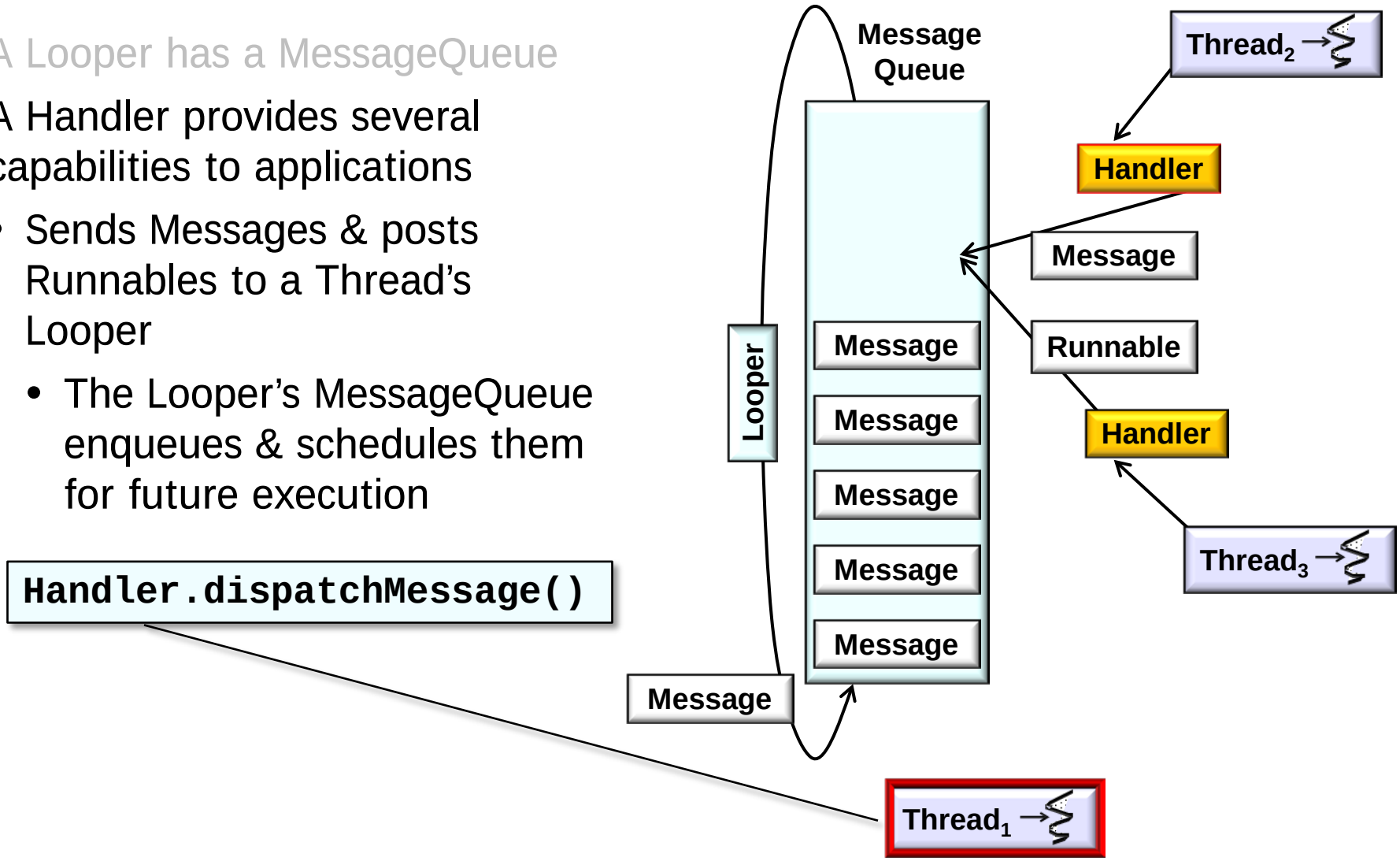
The Handler Class

- A Looper has a MessageQueue
- A Handler provides several capabilities to applications
  - Sends Messages & posts Runnables to a Thread's Looper
    - The Looper's MessageQueue enqueues & schedules them for future execution



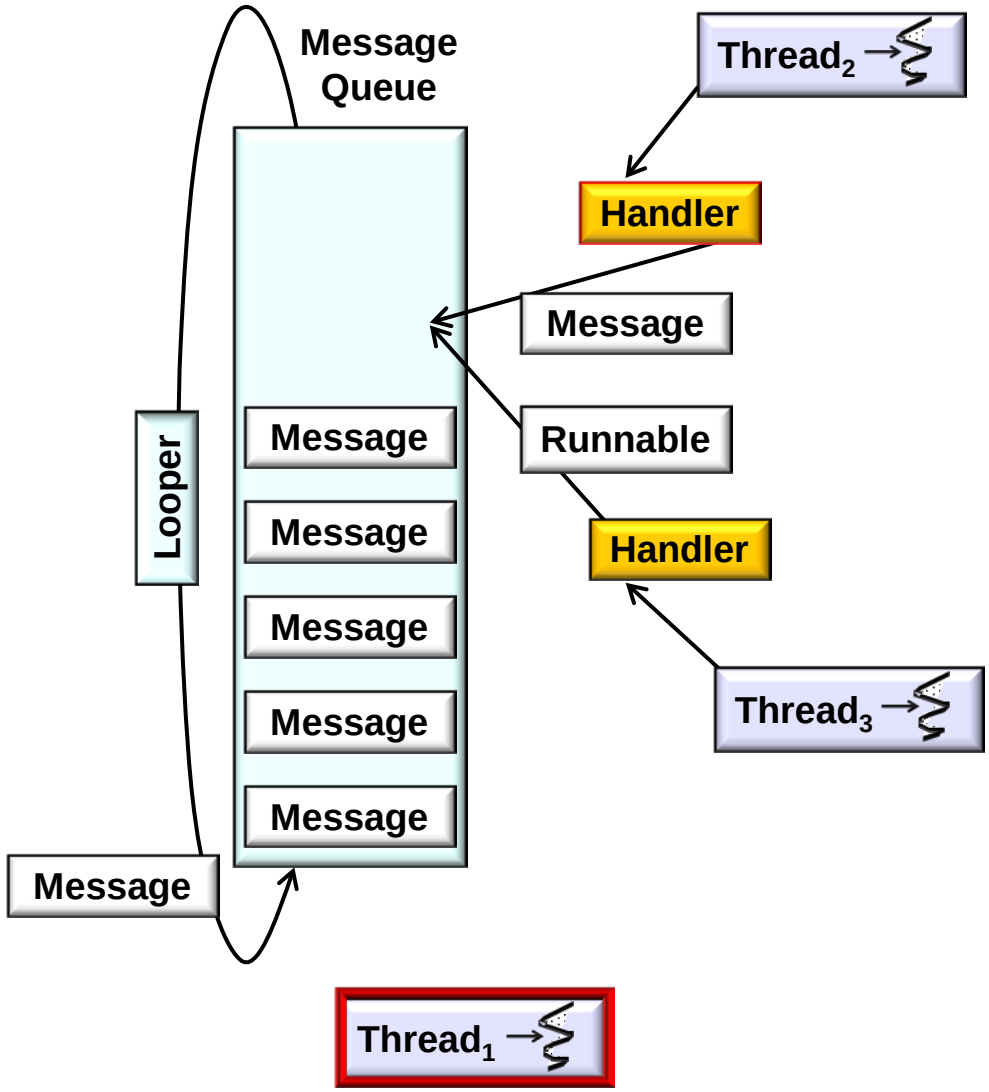
The Handler Class

- A Looper has a MessageQueue
- A Handler provides several capabilities to applications
  - Sends Messages & posts Runnables to a Thread's Looper
    - The Looper's MessageQueue enqueues & schedules them for future execution



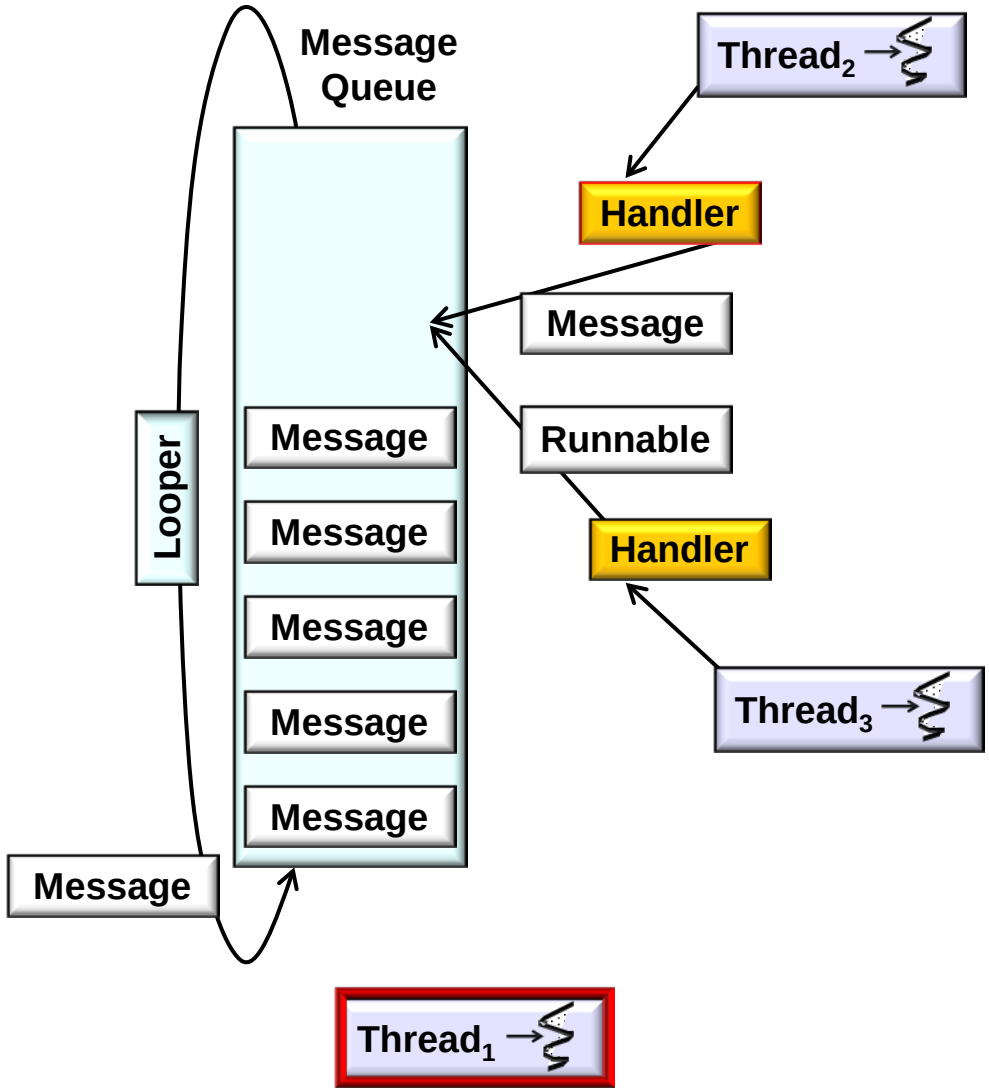
The Handler Class

- A Looper has a MessageQueue
- A Handler provides several capabilities to applications
  - Sends Messages & posts Runnables to a Thread's Looper
  - Collaborates with Looper to serialize the processing of Messages in a Thread



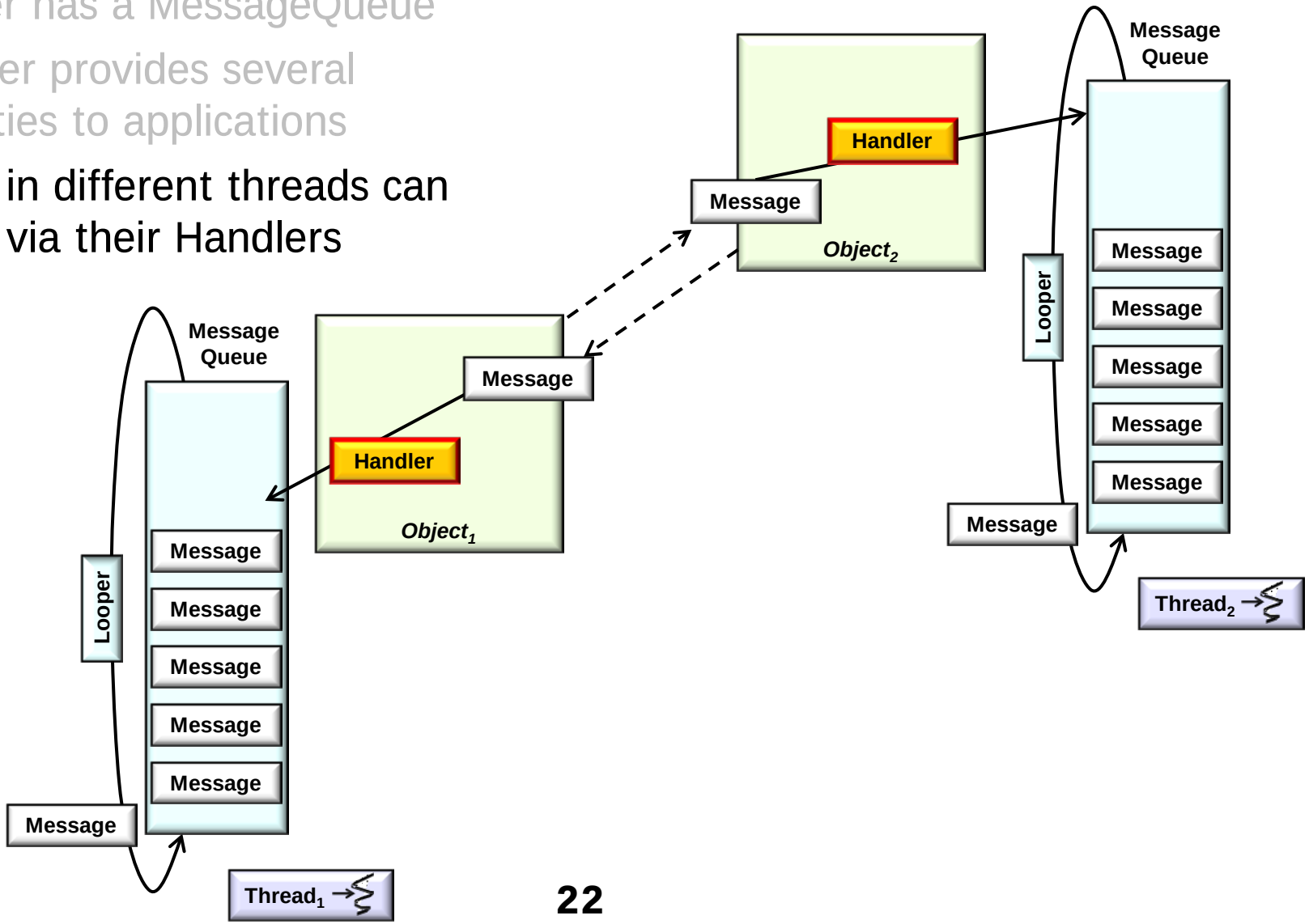
The Handler Class

- A Looper has a MessageQueue
- A Handler provides several capabilities to applications
  - Sends Messages & posts Runnables to a Thread's Looper
  - Collaborates with Looper to serialize the processing of Messages in a Thread
    - Can simplify concurrency control if design rules are followed



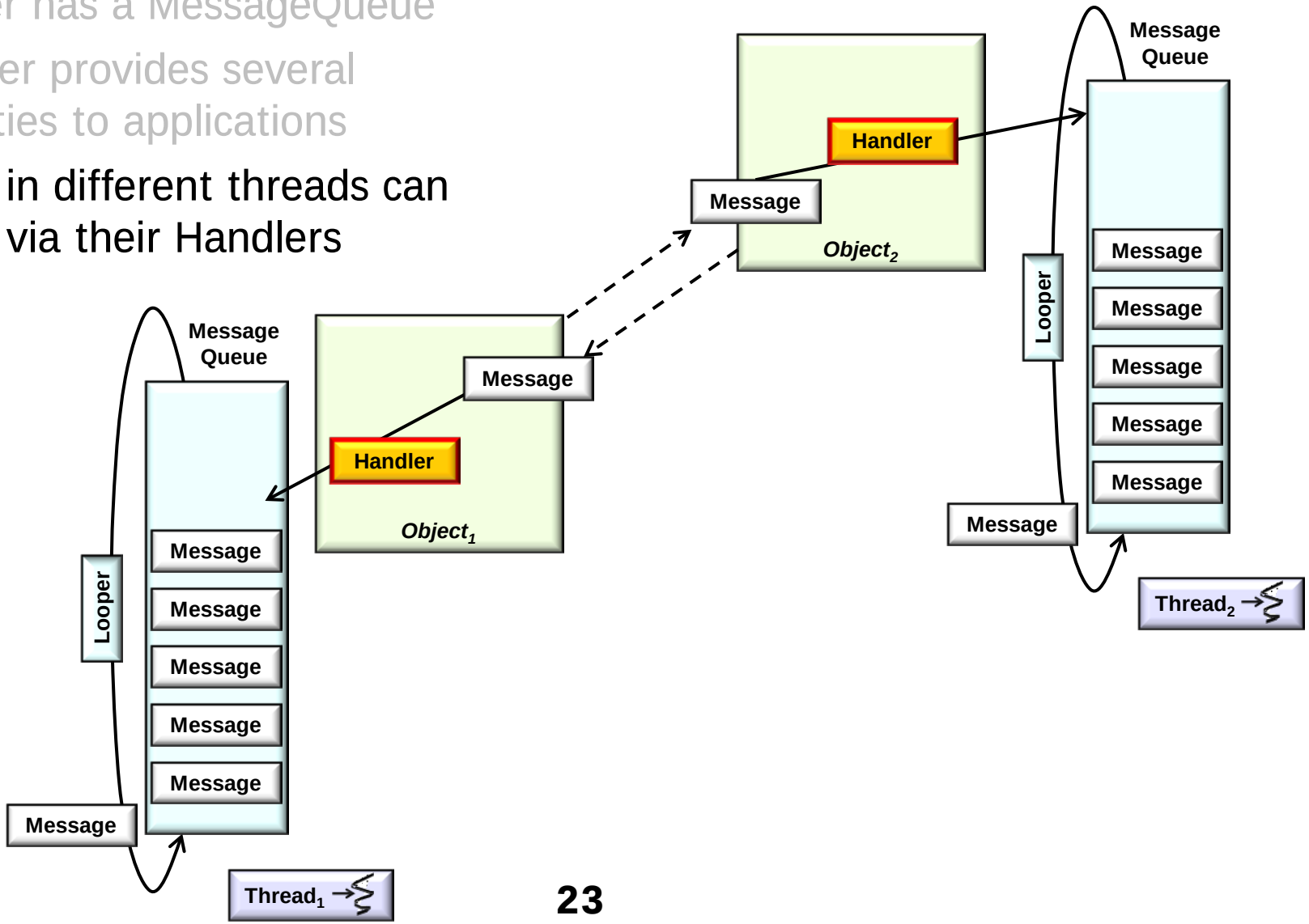
The Handler Class

- A Looper has a MessageQueue
- A Handler provides several capabilities to applications
- Objects in different threads can interact via their Handlers



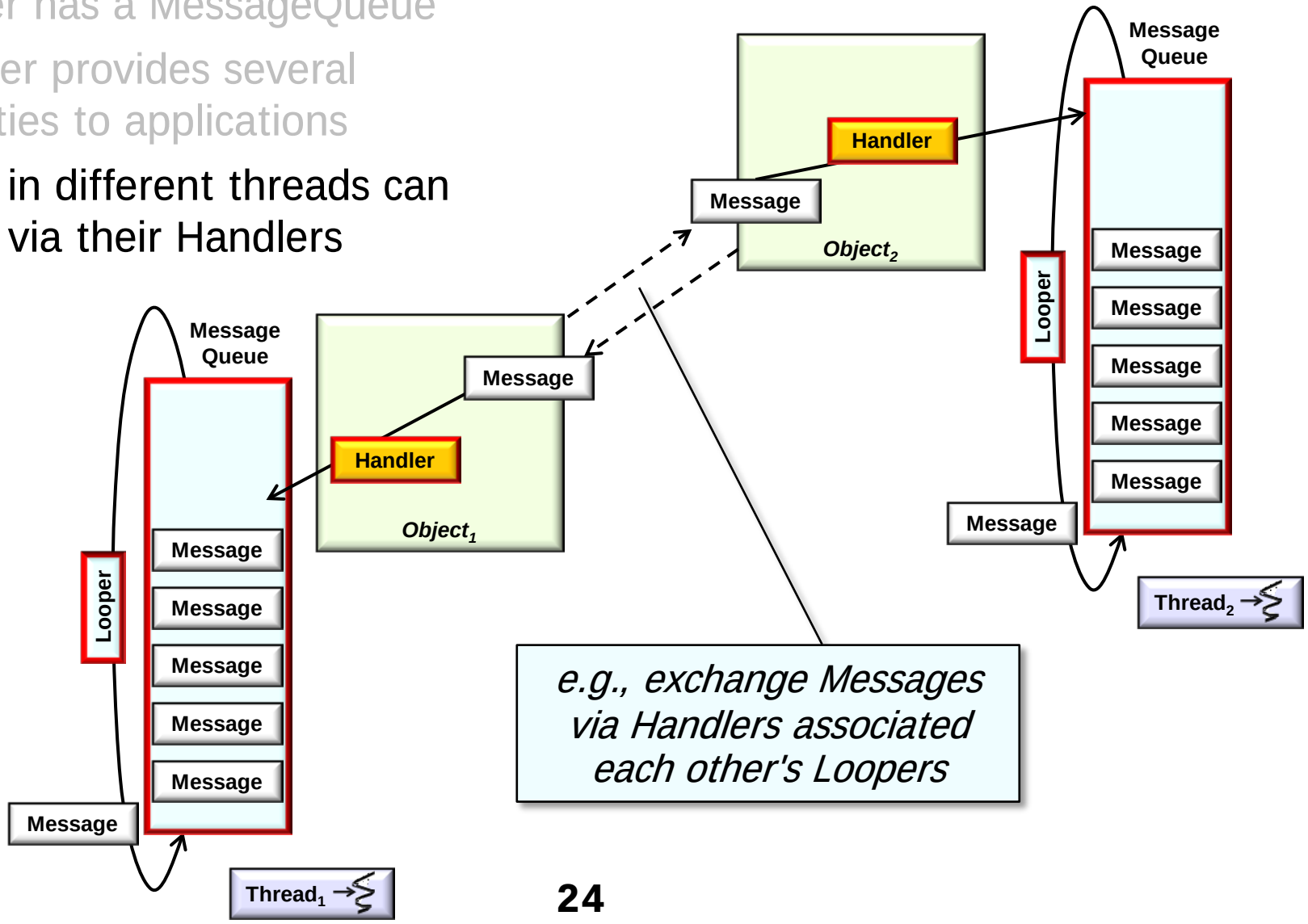
The Handler Class

- A Looper has a MessageQueue
- A Handler provides several capabilities to applications
- Objects in different threads can interact via their Handlers



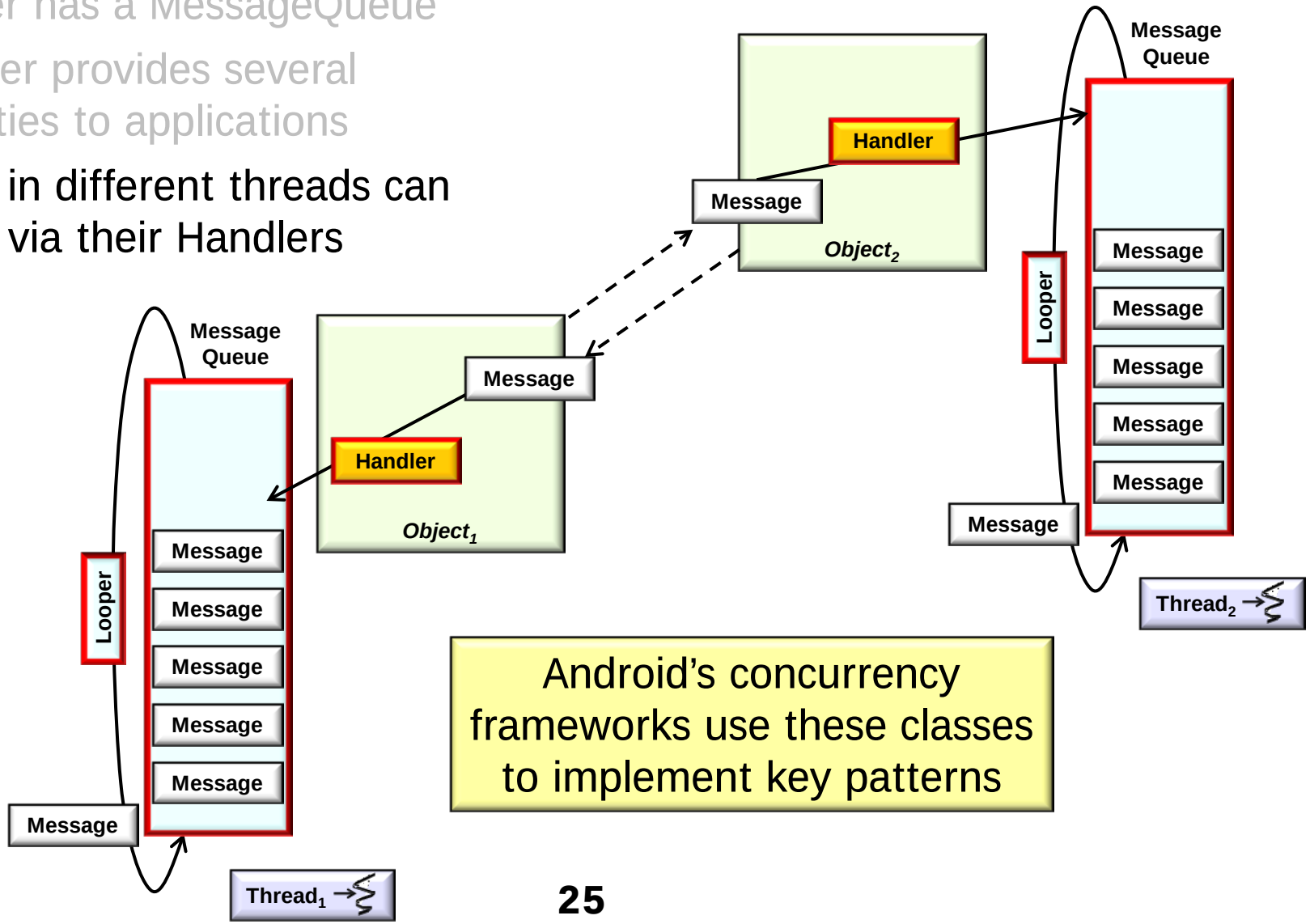
The Handler Class

- A Looper has a MessageQueue
- A Handler provides several capabilities to applications
- Objects in different threads can interact via their Handlers



The Handler Class

- A Looper has a MessageQueue
- A Handler provides several capabilities to applications
- Objects in different threads can interact via their Handlers

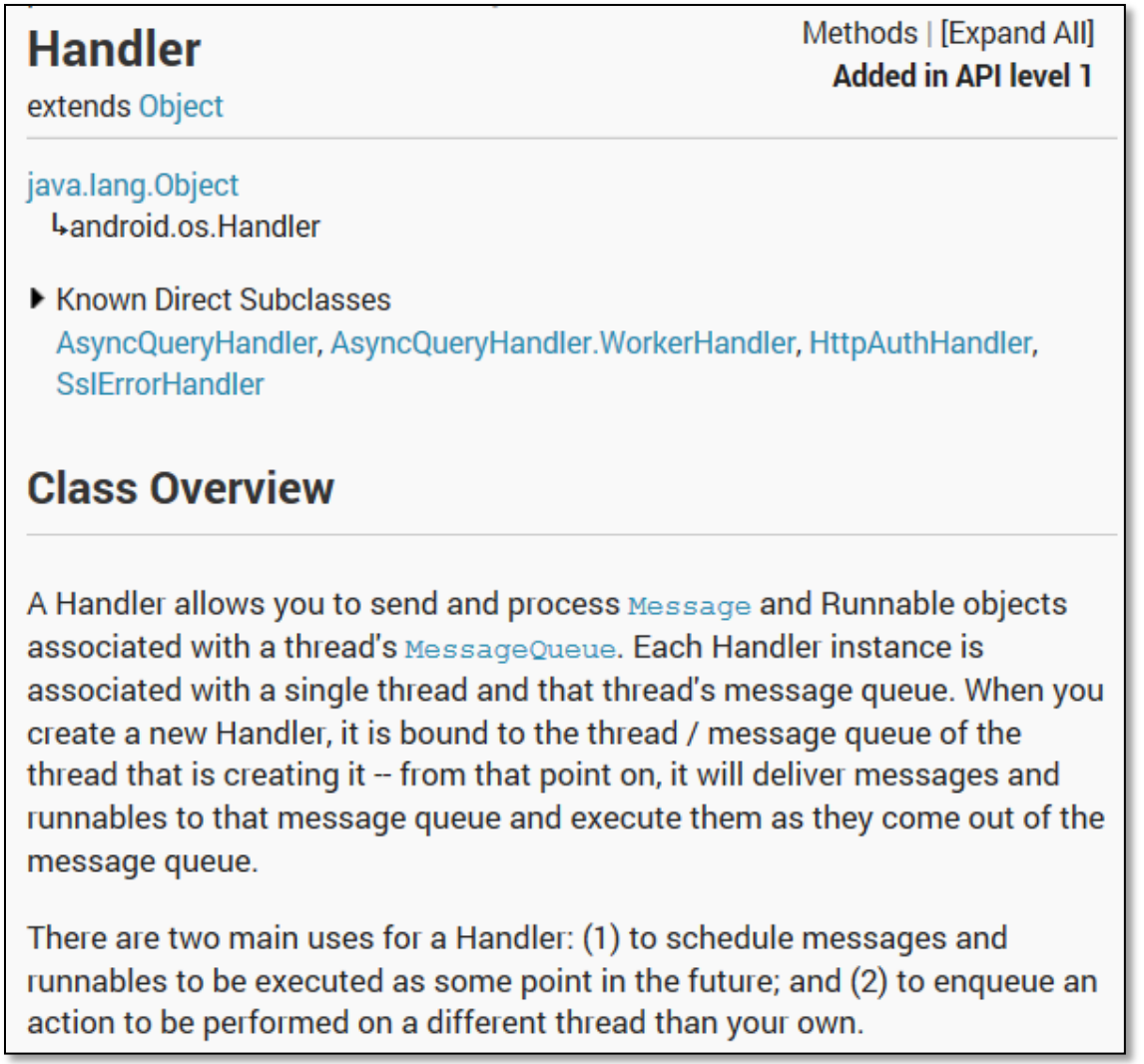


Android's concurrency frameworks use these classes to implement key patterns

Categories of Methods in the Handler Class

Categories of Methods in the Handler Class

- The Handler class has over two dozen methods



Handler

Methods | [Expand All]
Added in API level 1

extends Object

java.lang.Object
↳android.os.Handler

▸ Known Direct Subclasses
AsyncQueryHandler, AsyncQueryHandler.WorkerHandler, HttpAuthHandler, SslErrorHandler

Class Overview

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

There are two main uses for a Handler: (1) to schedule messages and runnables to be executed as some point in the future; and (2) to enqueue an action to be performed on a different thread than your own.

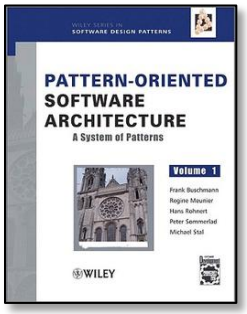
developer.android.com/reference/android/os/Handler.html has more info

Categories of Methods in the Handler Class

- The Handler class has over two dozen methods
- These methods can be grouped into four main categories

Categories of Methods in the Handler Class

- The Handler class has over two dozen methods
- These methods can be grouped into four main categories



Categories of Methods in the Handler Class

- The Handler class has over two dozen methods
- These methods can be grouped into four main categories
  - Posting/removing Runnables

`boolean post(Runnable r)`

- Add Runnable to MessageQueue & run when MessageQueue is ready

`void removeCallbacks(Runnable r)`

- Remove any pending posts of Runnable r that are in the MessageQueue

…

Categories of Methods in the Handler Class

- The Handler class has over two dozen methods
- These methods can be grouped into four main categories
  - Posting/removing Runnables
    - Insert/delete a Runnable into/from MessageQueue associated with the Handler

`boolean post(Runnable r)`

- Add Runnable to MessageQueue & run when MessageQueue is ready

`void removeCallbacks(Runnable r)`

- Remove any pending posts of Runnable r that are in the MessageQueue

…

Categories of Methods in the Handler Class

- The Handler class has over two dozen methods
- These methods can be grouped into four main categories
  - Posting/removing Runnables
    - Insert/delete a Runnable into/from MessageQueue associated with the Handler
    - The Handler & its Thread-specific Looper dequeue each Runnable & dispatch it's run() hook method

`boolean post(Runnable r)`

- Add Runnable to MessageQueue & run when MessageQueue is ready

`void removeCallbacks(Runnable r)`

- Remove any pending posts of Runnable r that are in the MessageQueue

…

Categories of Methods in the Handler Class

- The Handler class has over two dozen methods
- These methods can be grouped into four main categories
  - Posting/removing Runnables
  - Sending/removing Messages

`boolean sendMessage(Message msg)`

- Puts msg at end of queue immediately

`void removeMessages(int what)`

- Remove any pending posts of Messages with code ‘what’ that are in the MessageQueue

...

Categories of Methods in the Handler Class

- The Handler class has over two dozen methods
- These methods can be grouped into four main categories
  - Posting/removing Runnables
  - Sending/removing Messages
    - Insert/delete a Message into/from MessageQueue associated with the Handler

`boolean sendMessage(Message msg)`

- Puts msg at end of queue immediately

`void removeMessages(int what)`

- Remove any pending posts of Messages with code ‘what’ that are in the MessageQueue

…

Categories of Methods in the Handler Class

- The Handler class has over two dozen methods
- These methods can be grouped into four main categories
  - Posting/removing Runnables
  - Sending/removing Messages
    - Insert/delete a Message into/from MessageQueue associated with the Handler
    - A Message contains a bundle of data processed by the Handler's handleMessage() hook method

`boolean sendMessage(Message msg)`

- Puts msg at end of queue immediately

`void removeMessages(int what)`

- Remove any pending posts of Messages with code 'what' that are in the MessageQueue

...

Categories of Methods in the Handler Class

- The Handler class has over two dozen methods
- These methods can be grouped into four main categories
  - Posting/removing Runnables
  - Sending/removing Messages
  - Obtaining Messages

`Message obtainMessage()`

- Returns a new Message from the global message pool

`Message obtainMessage (int what)`

- Same as obtainMessage(), except that it also sets the what member of the returned Message

`Message obtainMessage (int what, int arg1, int arg2, Object obj)`

- Same as obtainMessage(), except that it also sets the what, obj, arg1,and arg2 values on the returned Message

...

Categories of Methods in the Handler Class

- The Handler class has over two dozen methods
- These methods can be grouped into four main categories
  - Posting/removing Runnables
  - Sending/removing Messages
  - Obtaining Messages
    - Factories that create Messages passed to sendMessage()

`Message obtainMessage()`

- Returns a new Message from the global message pool

`Message obtainMessage (int what)`

- Same as obtainMessage(), except that it also sets the what member of the returned Message

`Message obtainMessage (int what, int arg1, int arg2, Object obj)`

- Same as obtainMessage(), except that it also sets the what, obj, arg1,and arg2 values on the returned Message

...

See en.wikipedia.org/wiki/Creational_pattern

Categories of Methods in the Handler Class

- The Handler class has over two dozen methods
- These methods can be grouped into four main categories
  - Posting/removing Runnables
  - Sending/removing Messages
  - Obtaining Messages
  - Dispatching/handling Messages

`void dispatchMessage(Message msg)`

- Invoke the appropriate callback (e.g., run() or handleMessage()) based on the type of the Message

`void handleMessage(Message msg)`

- Subclasses must implement this method to receive messages

Categories of Methods in the Handler Class

- The Handler class has over two dozen methods
- These methods can be grouped into four main categories
  - Posting/removing Runnables
  - Sending/removing Messages
  - Obtaining Messages
  - Dispatching/handling Messages
    - handleMessage() hook method must be overridden receive & process Messages enqueued via sendMessage()

`void dispatchMessage(Message msg)`

- Invoke the appropriate callback (e.g., run() or handleMessage()) based on the type of the Message

`void handleMessage(Message msg)`

- Subclasses must implement this method to receive messages

Categories of Methods in the Handler Class

- The Handler class has over two dozen methods
- These methods can be grouped into four main categories
  - Posting/removing Runnables
  - Sending/removing Messages
  - Obtaining Messages
  - Dispatching/handling Messages
    - handleMessage() hook method must be overridden receive & process Messages enqueued via sendMessage()
    - handleMessage() runs in the context of the Handler Thread

`void dispatchMessage(Message msg)`

- Invoke the appropriate callback (e.g., run() or handleMessage()) based on the type of the Message

`void handleMessage(Message msg)`

- Subclasses must implement this method to receive messages

See upcoming parts on the "*Active Object* pattern"

Categories of Methods in the Handler Class

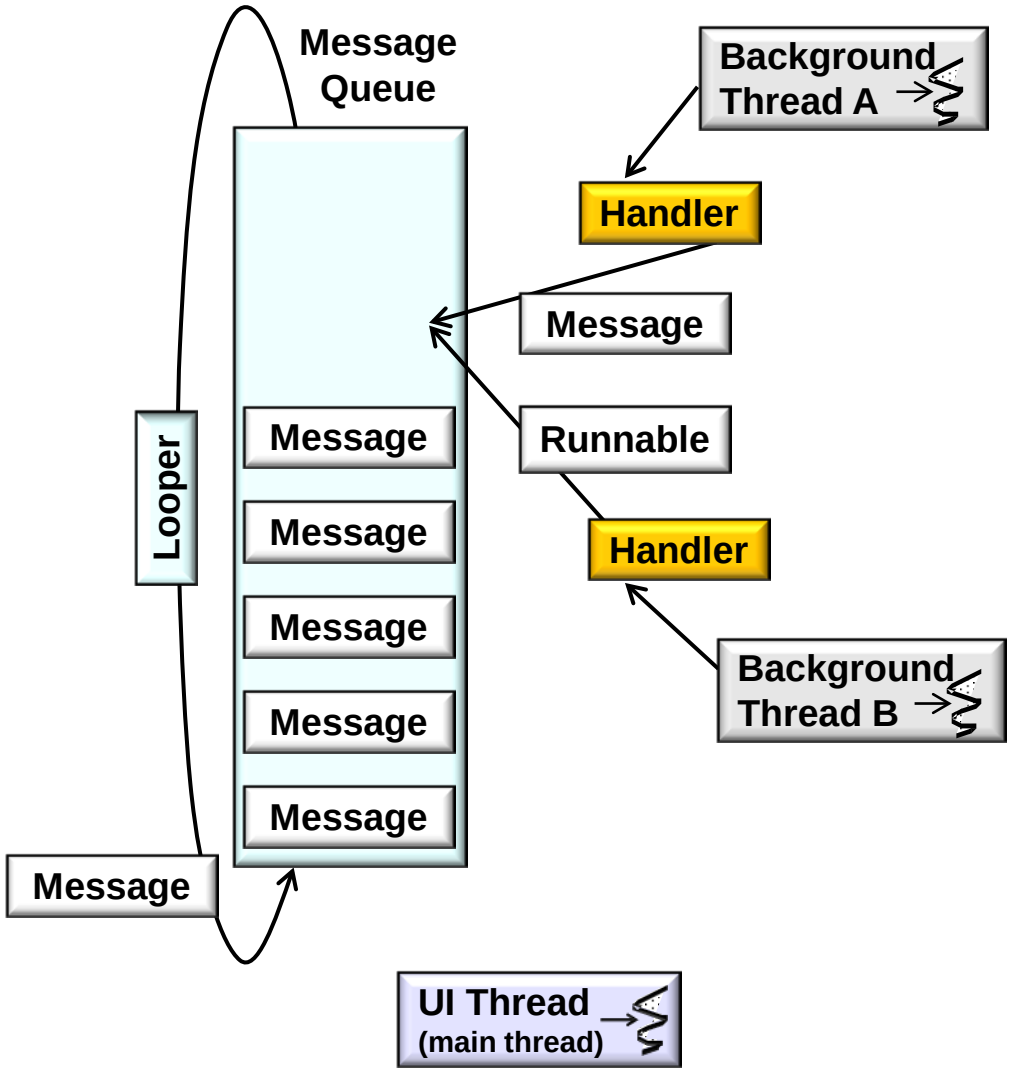
- The Handler class has over two dozen methods
- These methods can be grouped into four main categories

The next two parts of the module cover these Handler methods in more detail

Summary

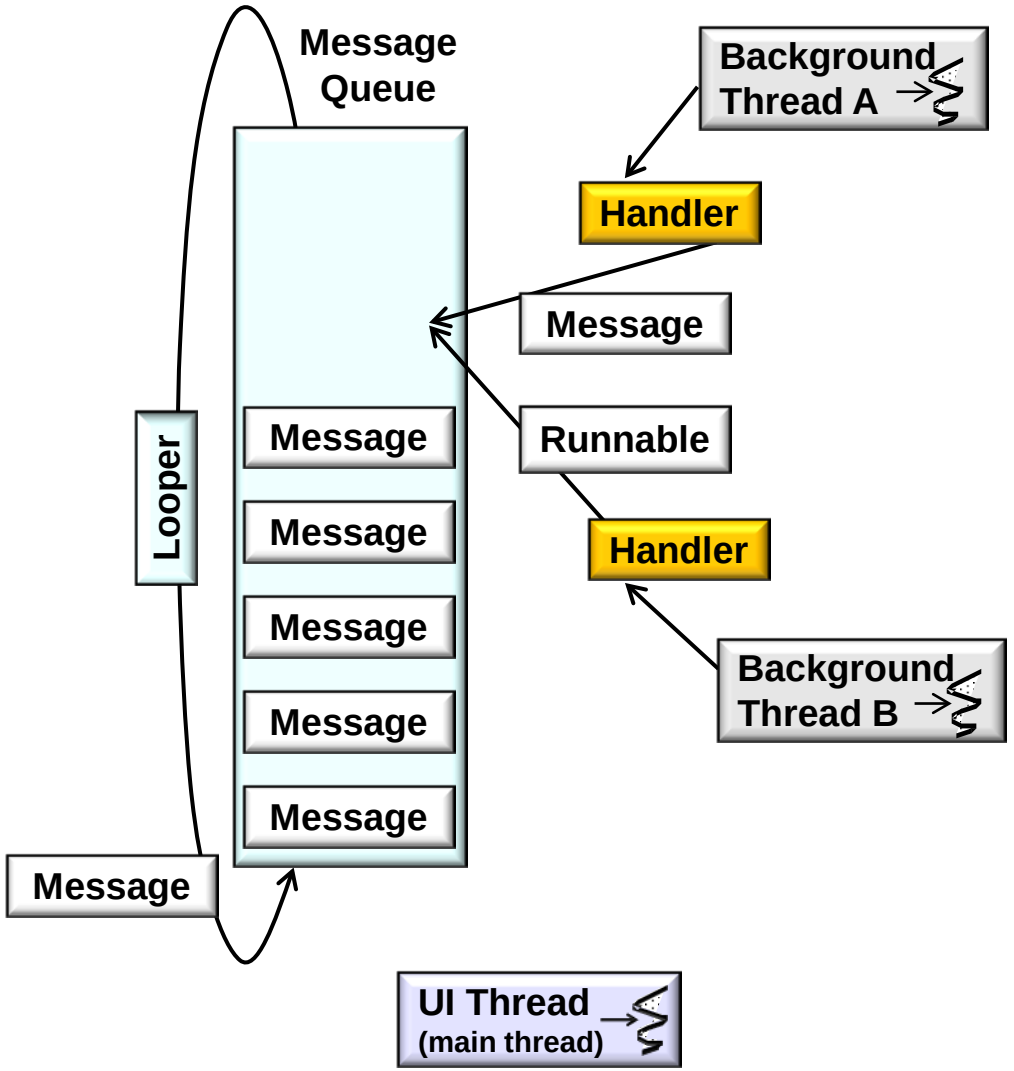
Summary

- UI & background threads often need to communicate



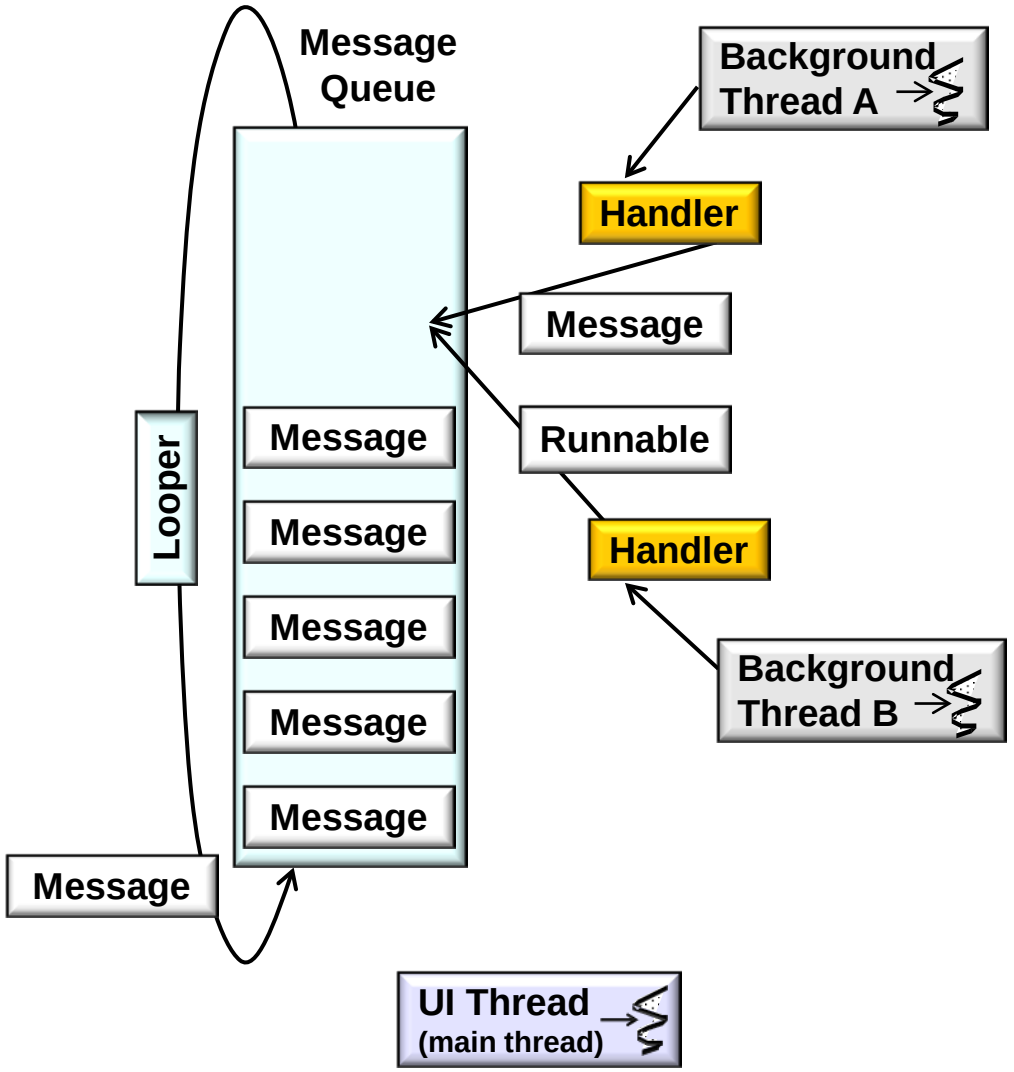
Summary

- UI & background threads often need to communicate, e.g.
  - To perform their operations concurrently



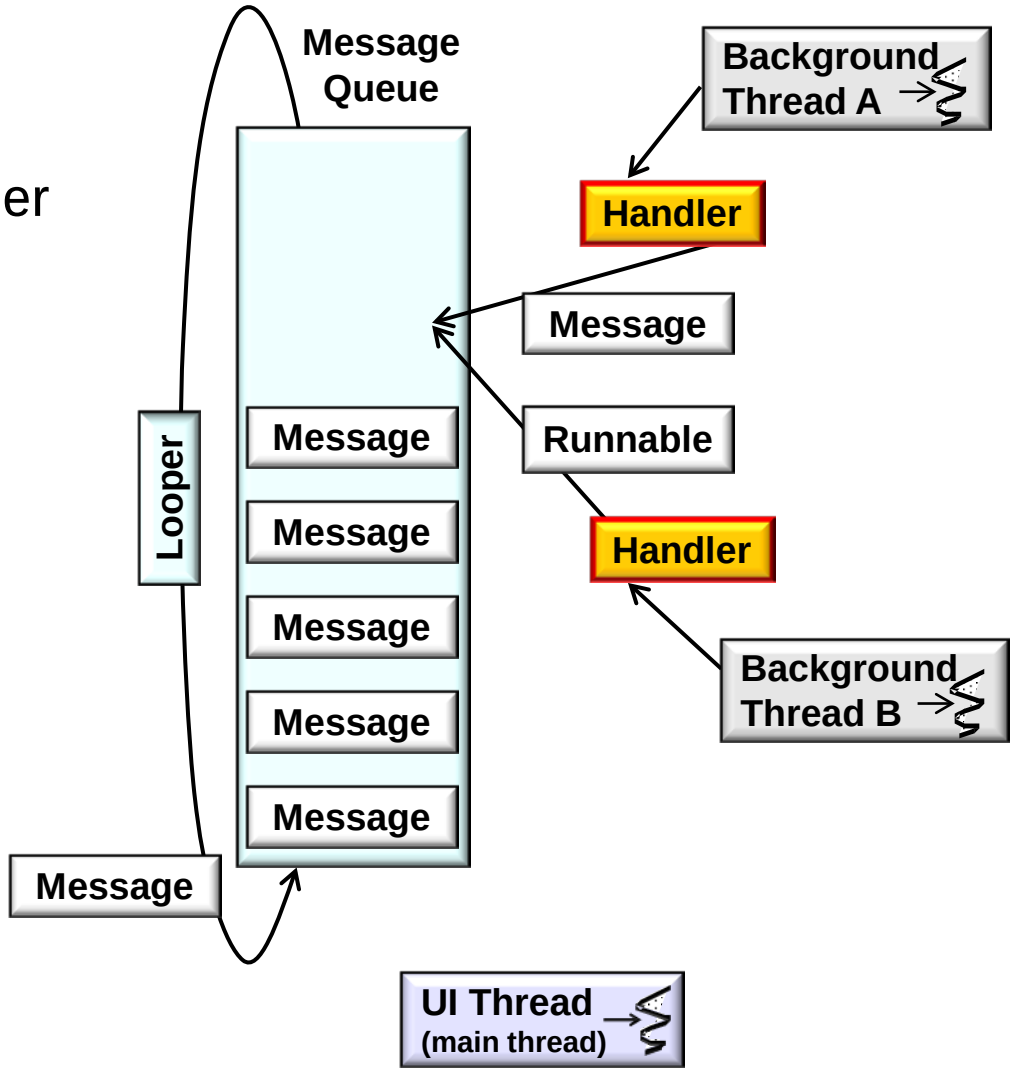
Summary

- UI & background threads often need to communicate, e.g.
  - To perform their operations concurrently
  - To coordinate their behavior



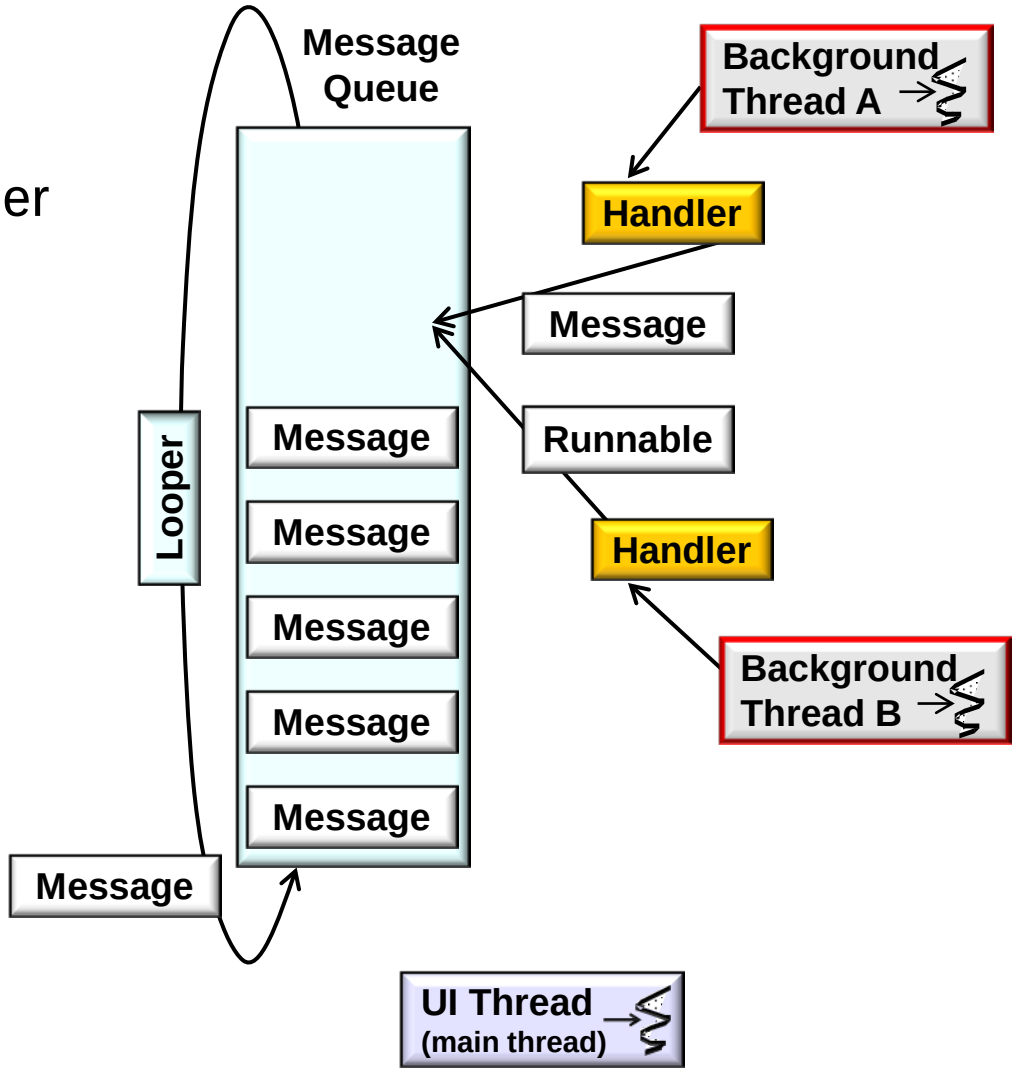
Summary

- UI & background threads often need to communicate
- HaMeR framework provides Handler class to support this use case



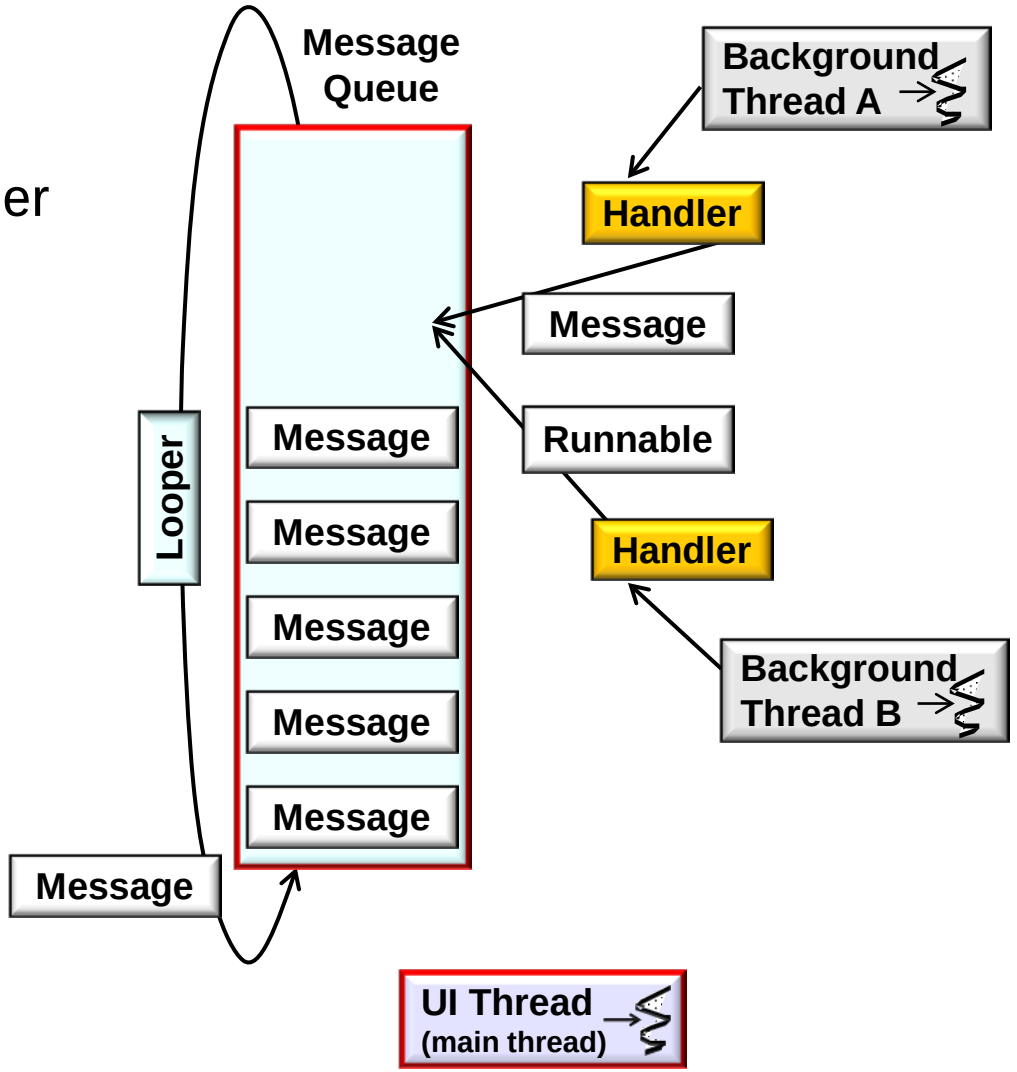
Summary

- UI & background threads often need to communicate
- HaMeR framework provides Handler class to support this use case
  - A Handler allows background Threads to send Messages or post Runnables to the UI Thread's MessageQueue



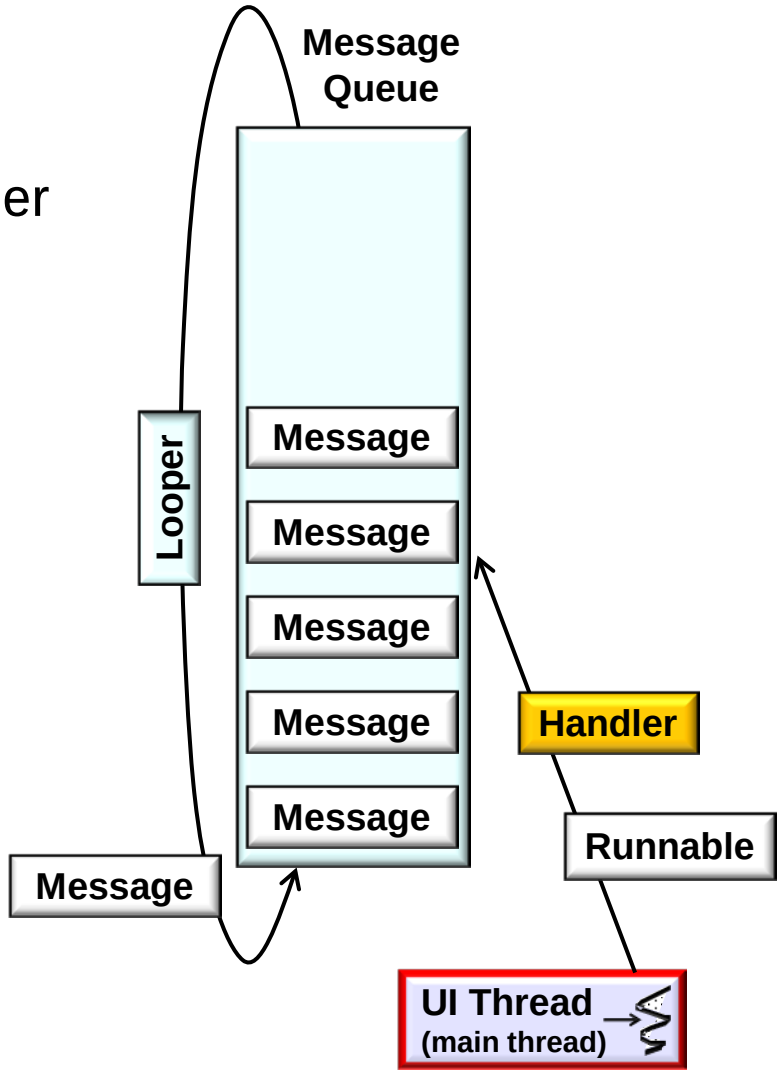
Summary

- UI & background threads often need to communicate
- HaMeR framework provides Handler class to support this use case
  - A Handler allows background Threads to send Messages or post Runnables to the UI Thread's MessageQueue



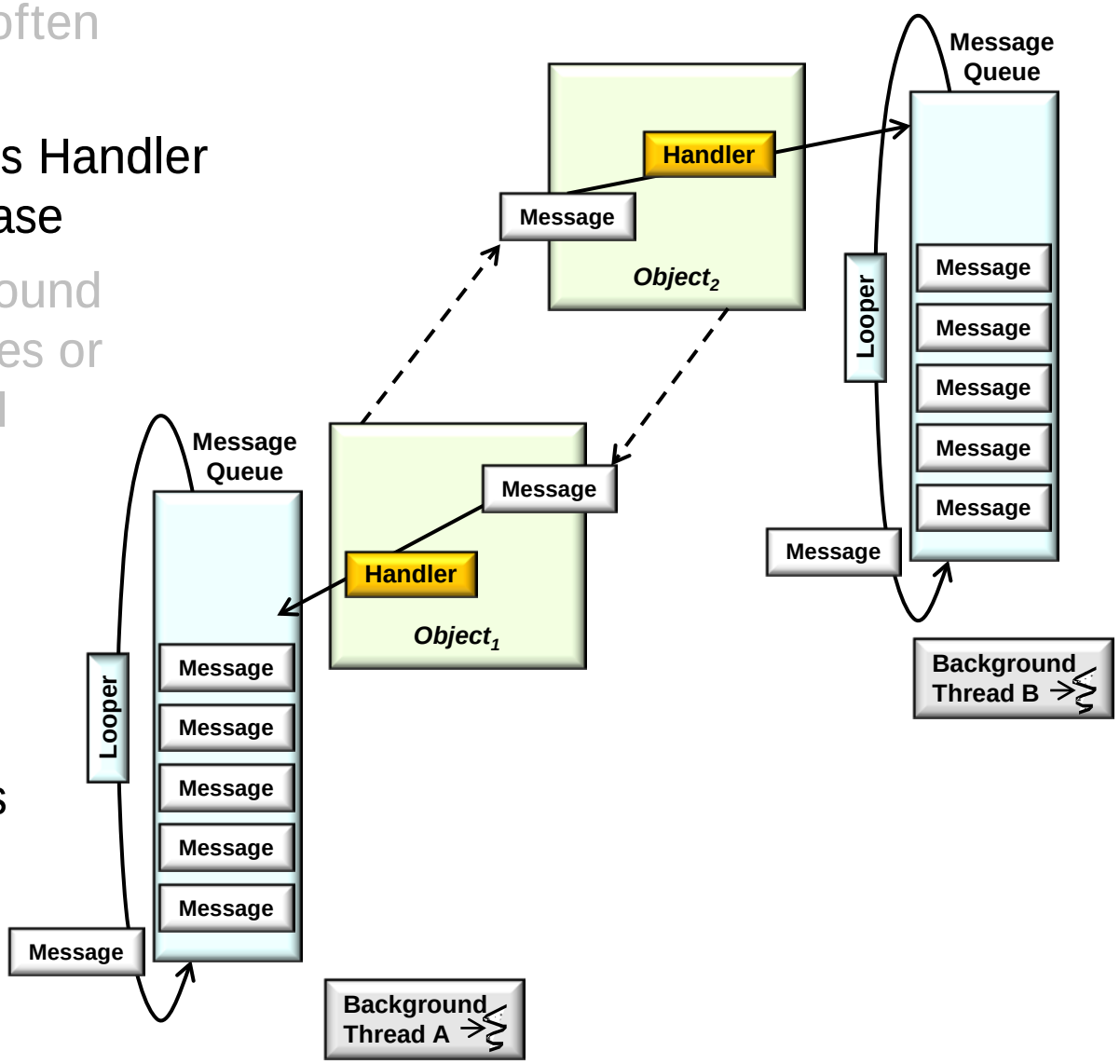
Summary

- UI & background threads often need to communicate
- HaMeR framework provides Handler class to support this use case
  - A Handler allows background Threads to send Messages or post Runnables to the UI Thread's MessageQueue
  - It can also allow a Thread to send/post to itself



Summary

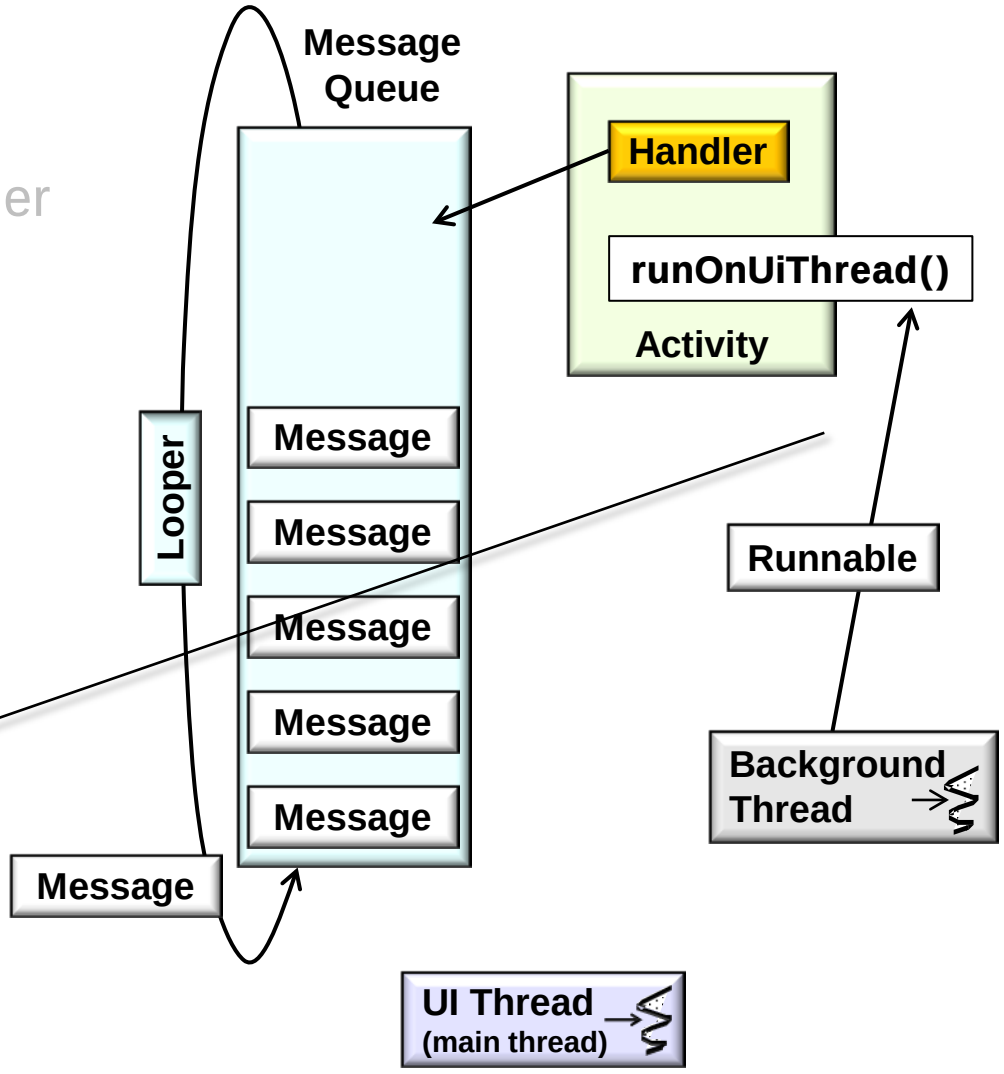
- UI & background threads often need to communicate
- HaMeR framework provides Handler class to support this use case
  - A Handler allows background Threads to send Messages or post Runnables to the UI Thread's MessageQueue
  - It can also allow a Thread to send/post to itself
  - Background Threads can interact via Handlers



Summary

- UI & background threads often need to communicate
- HaMeR framework provides Handler class to support this use case
- post() methods used when senders know what operations to perform

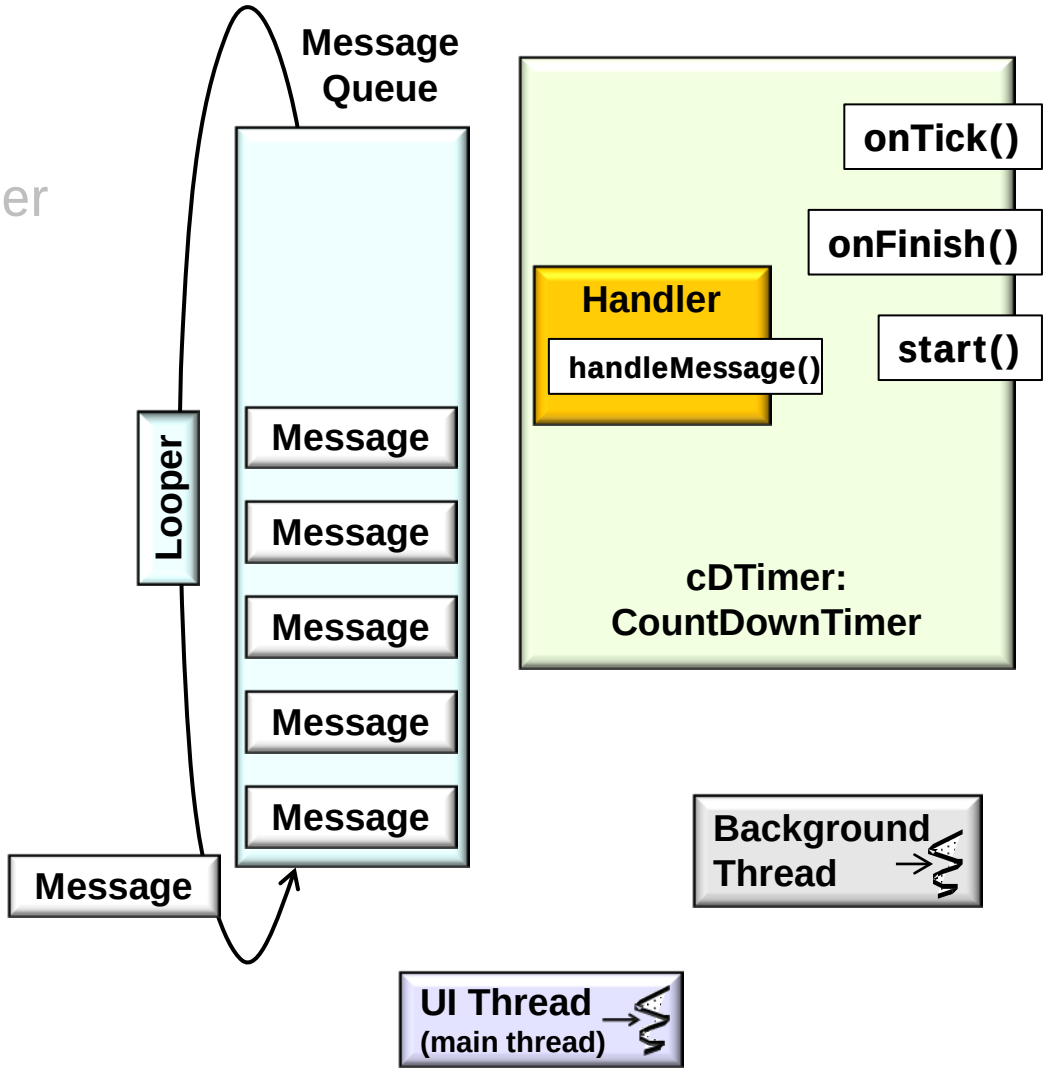
```
runOnUiThread
  (new Runnable() {
     public void run()
       { ... }
  });
```



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

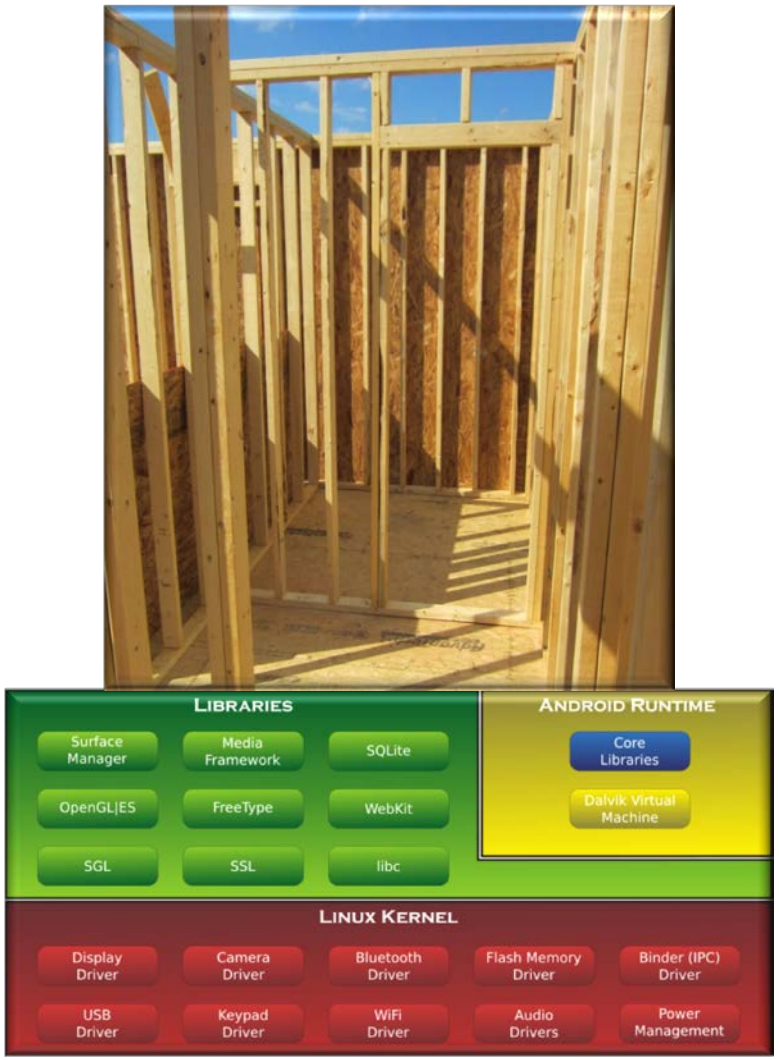
Summary

- UI & background threads often need to communicate
- HaMeR framework provides Handler class to support this use case
- post() methods used when senders know what operations to perform
- sendMethod() methods used when receivers know what operations to perform



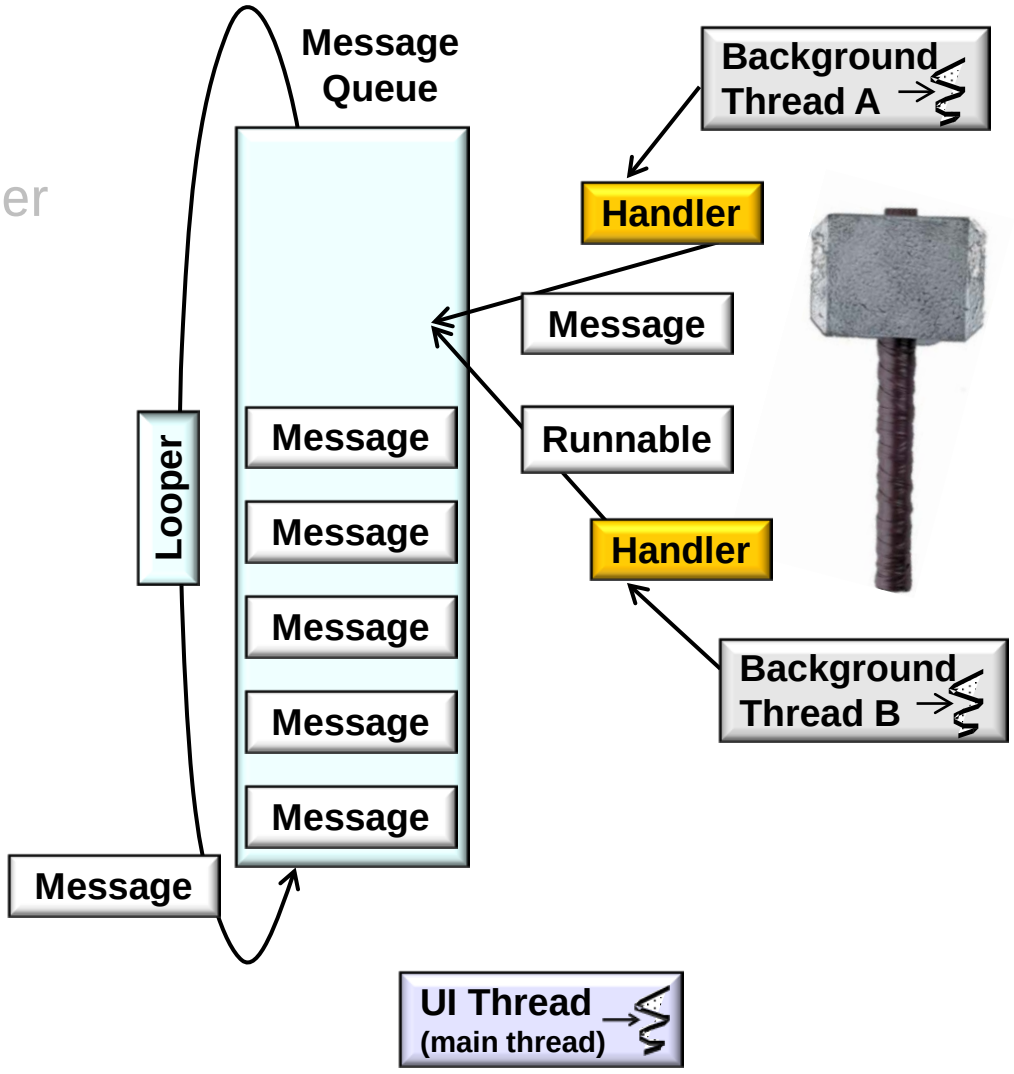
Summary

- UI & background threads often need to communicate
- HaMeR framework provides Handler class to support this use case
- post() methods used when senders know what operations to perform
- sendMethod() methods used when receivers know what operations to perform
- Handlers used in many Android applications & frameworks



Summary

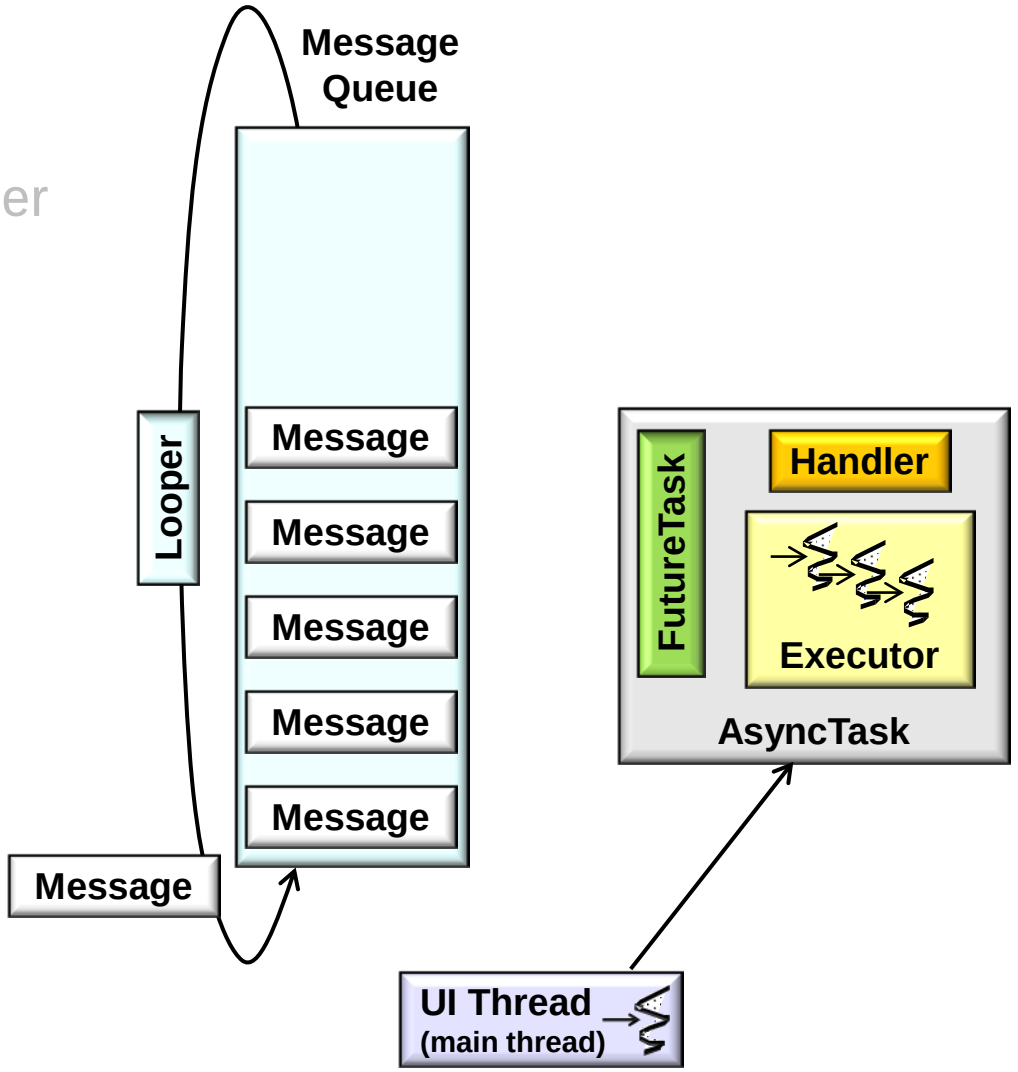
- UI & background threads often need to communicate
- HaMeR framework provides Handler class to support this use case
- post() methods used when senders know what operations to perform
- sendMethod() methods used when receivers know what operations to perform
- Handlers used in many Android applications & frameworks, e.g.
  - HaMeR framework



Next two parts focus on the HaMeR framework's use of Handlers et al

Summary

- UI & background threads often need to communicate
- HaMeR framework provides Handler class to support this use case
- post() methods used when senders know what operations to perform
- sendMethod() methods used when receivers know what operations to perform
- Handlers used in many Android applications & frameworks, e.g.
  - HaMeR framework
  - AsyncTask framework



See upcoming part on “the AsyncTask Framework”

Summary

- UI & background threads often need to communicate
- HaMeR framework provides Handler class to support this use case
- post() methods used when senders know what operations to perform
- sendMethod() methods used when receivers know what operations to perform
- Handlers used in many Android applications & frameworks
- Other sources explain how to use Handlers in Android concurrency frameworks & applications