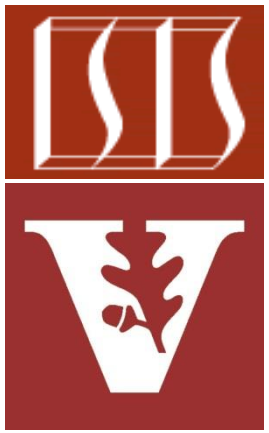
Java Thread:
Overview of the Case Study App

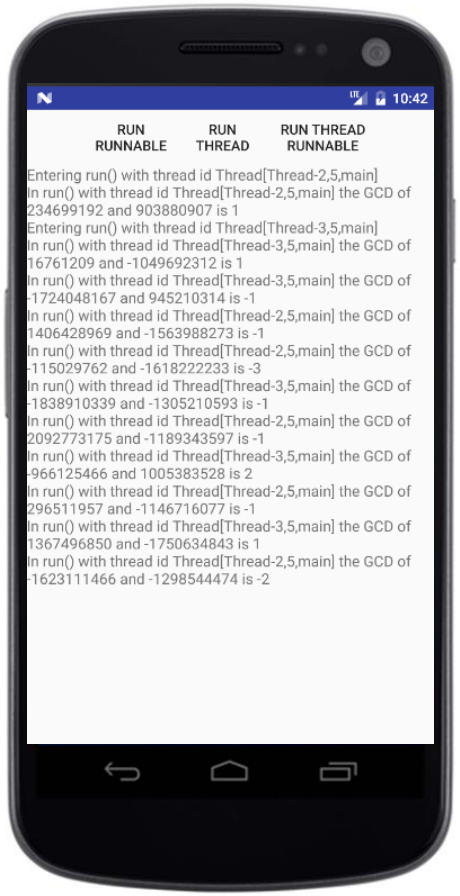


Douglas C. Schmidt
d.schmidt@vanderbilt.edu
www.dre.vanderbilt.edu/~schmidt

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

Learning Objectives in this Part of the Lesson

- Understand how Java threads support concurrency
- Learn how our case study app works



See github.com/douglascraigschmidt/POSA/tree/master/ex/M3/GCD/Concurrent

Runtime Behavior of the GCD Concurrent App

Runtime Behavior of the GCD Concurrent App

- Concurrently compute the greatest common divisor (GCD) of two #'s, which is the largest integer that divides two integers without a remainder

<<Java Class>>

Thread

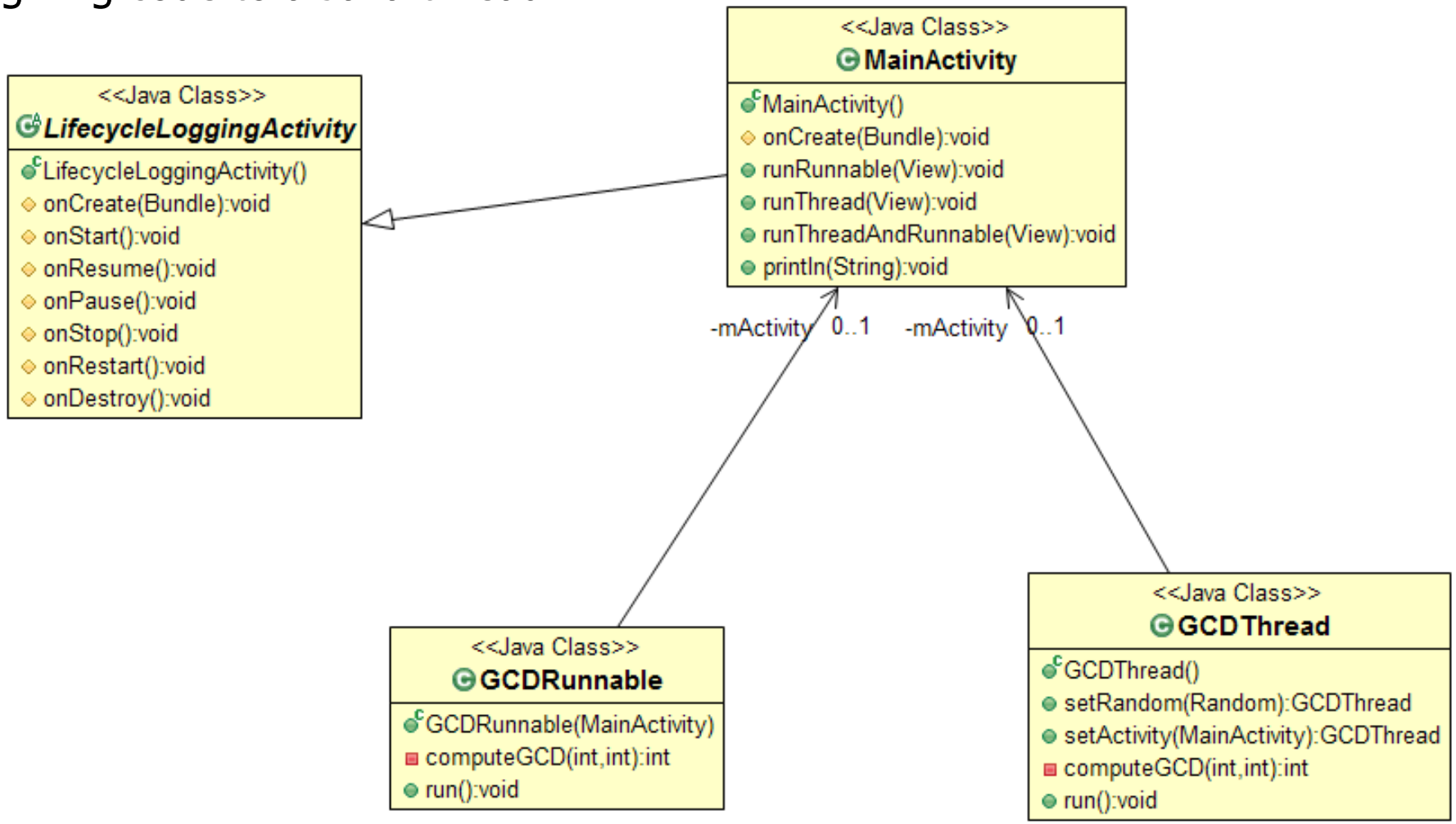
- yield():void
- currentThread():Thread
- sleep(long):void
- sleep(long,int):void
- Thread()
- Thread(Runnable)
- Thread(String)
- start():void
- run():void
- exit():void
- interrupt():void
- interrupted():boolean
- isInterrupted():boolean
- isAlive():boolean
- setPriority(int):void
- getPriority():int
- join(long):void
- join(long,int):void
- join():void
- setDaemon(boolean):void
- isDaemon():boolean

See en.wikipedia.org/wiki/Greatest_common_divisor

Design of the GCD Concurrent App

Design of the GCD Concurrent App

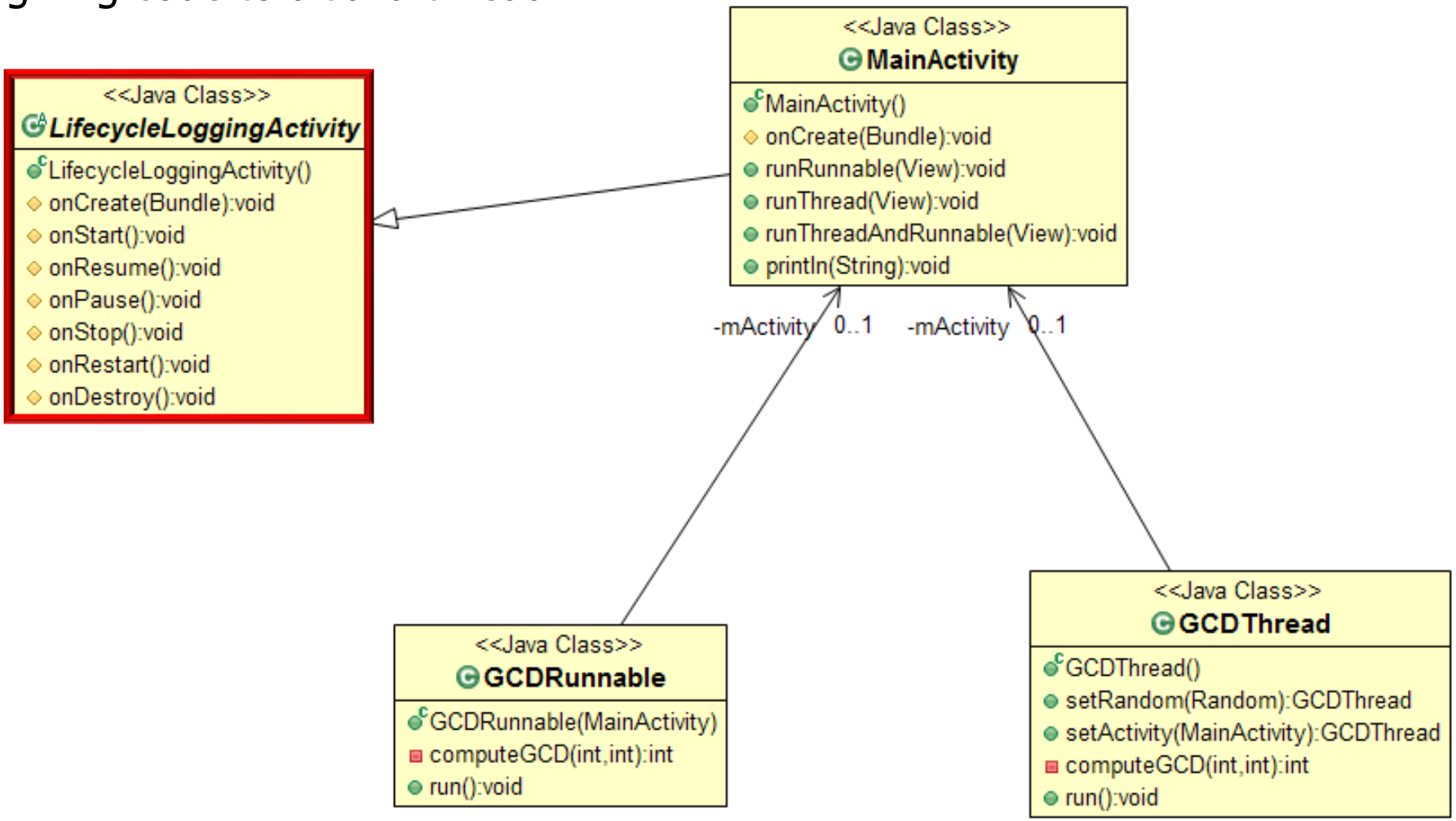
- This app shows various methods in Java's Thread class & alternative ways of giving code to a Java thread



See github.com/douglascraigschmidt/POSA/tree/master/ex/M3/GCD/Concurrent

Design of the GCD Concurrent App

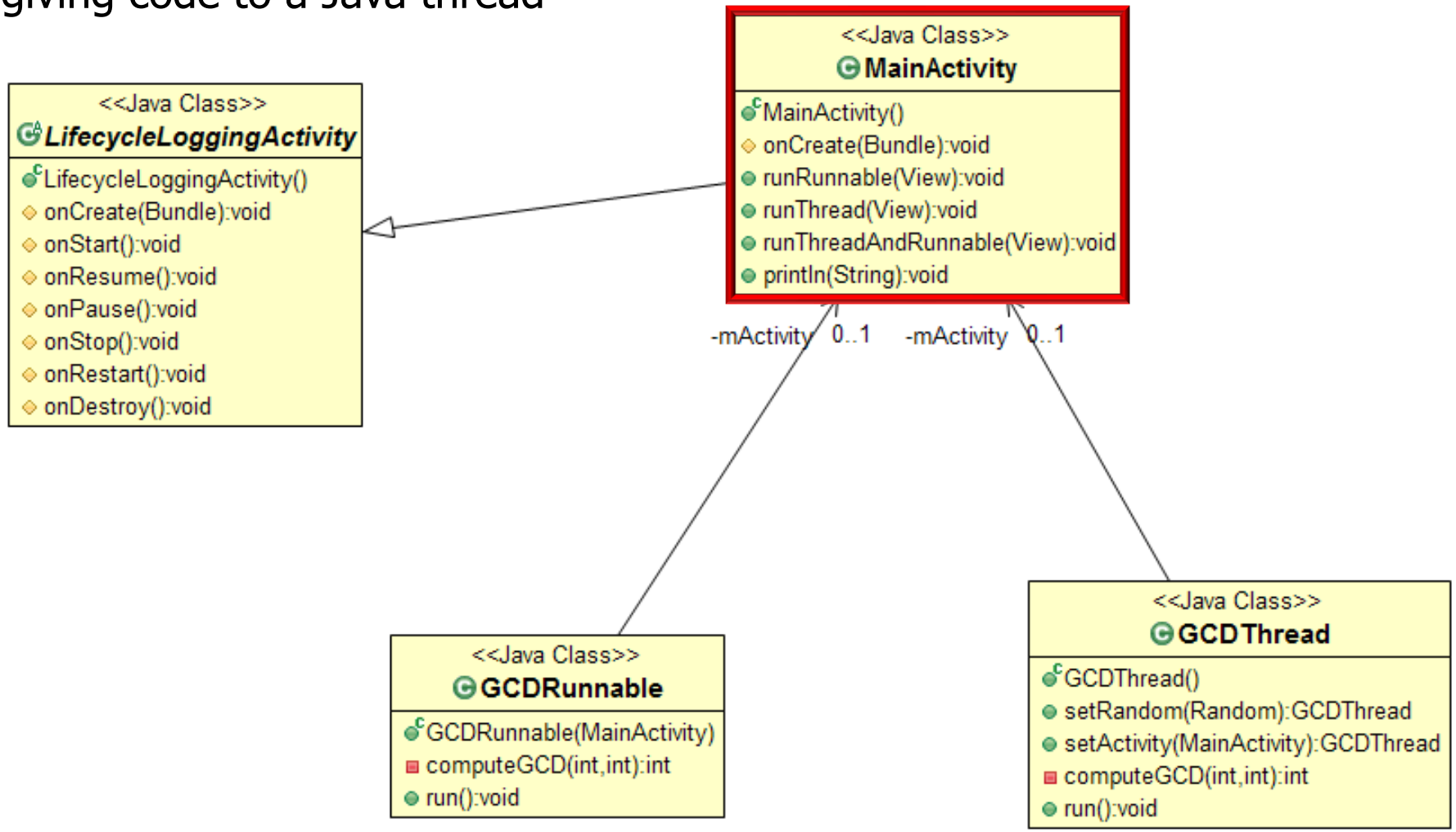
- This app shows various methods in Java's Thread class & alternative ways of giving code to a Java thread



Super class that logs various activity lifecycle hook methods to aid debugging

Design of the GCD Concurrent App

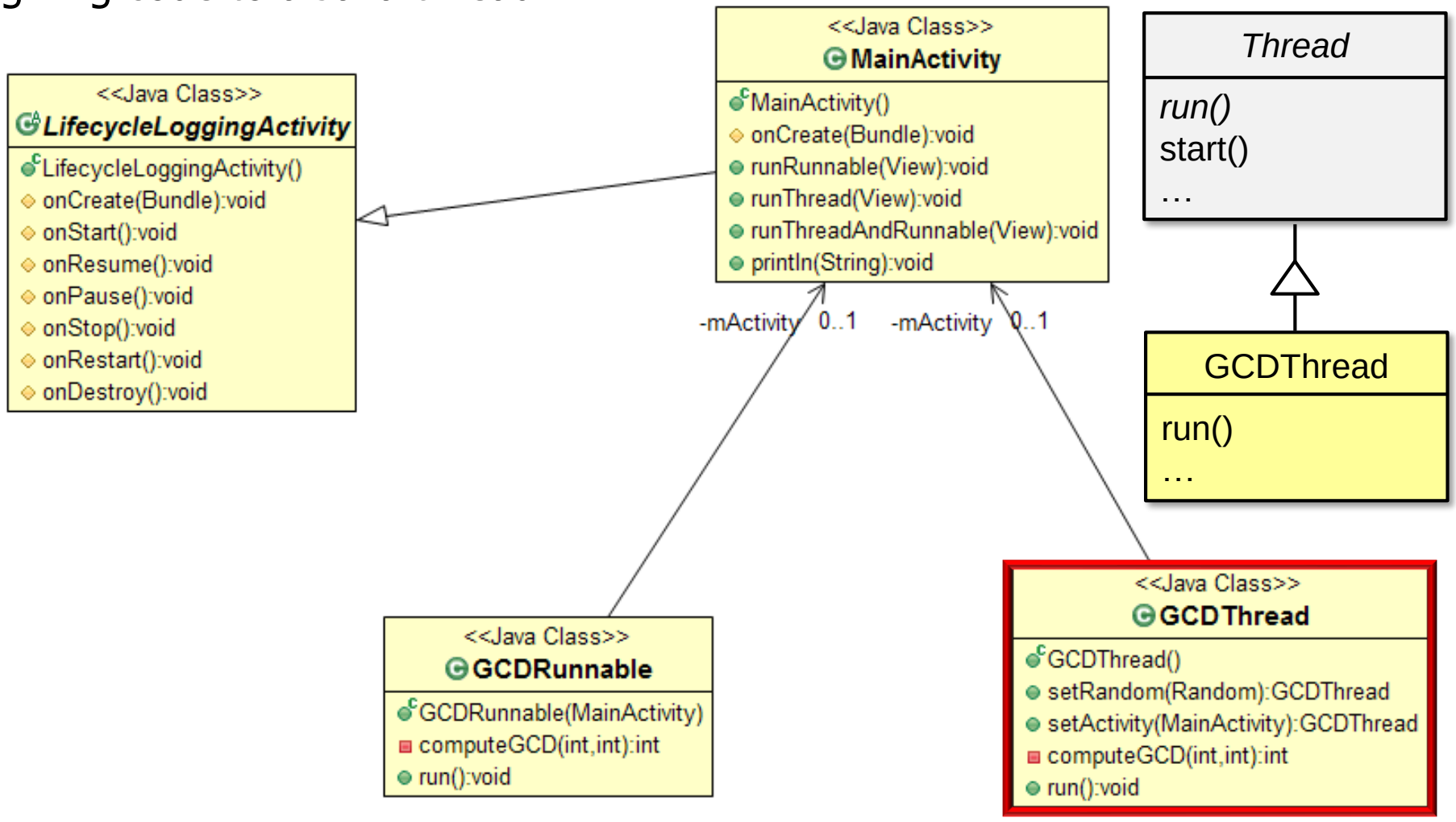
- This app shows various methods in Java's Thread class & alternative ways of giving code to a Java thread



Main entry point into the app that handles button presses from the user

Design of the GCD Concurrent App

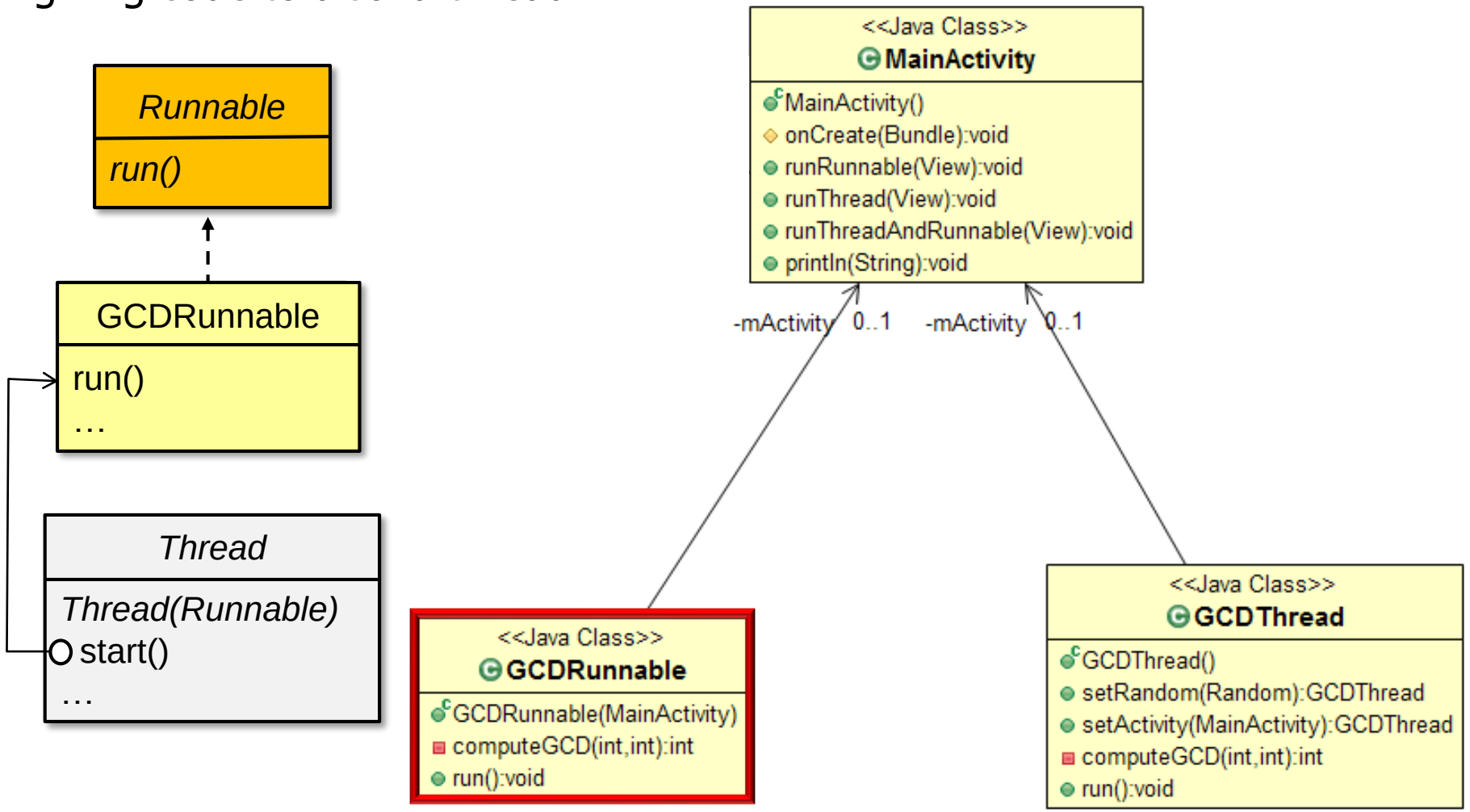
- This app shows various methods in Java's Thread class & alternative ways of giving code to a Java thread



Computes the GCD of two numbers by extending the Thread super class

Design of the GCD Concurrent App

- This app shows various methods in Java's Thread class & alternative ways of giving code to a Java thread



Computes the GCD of two numbers by implementing the Runnable interface

Design of the GCD Concurrent App

- We'll explore the implementations of these threading alternatives shortly

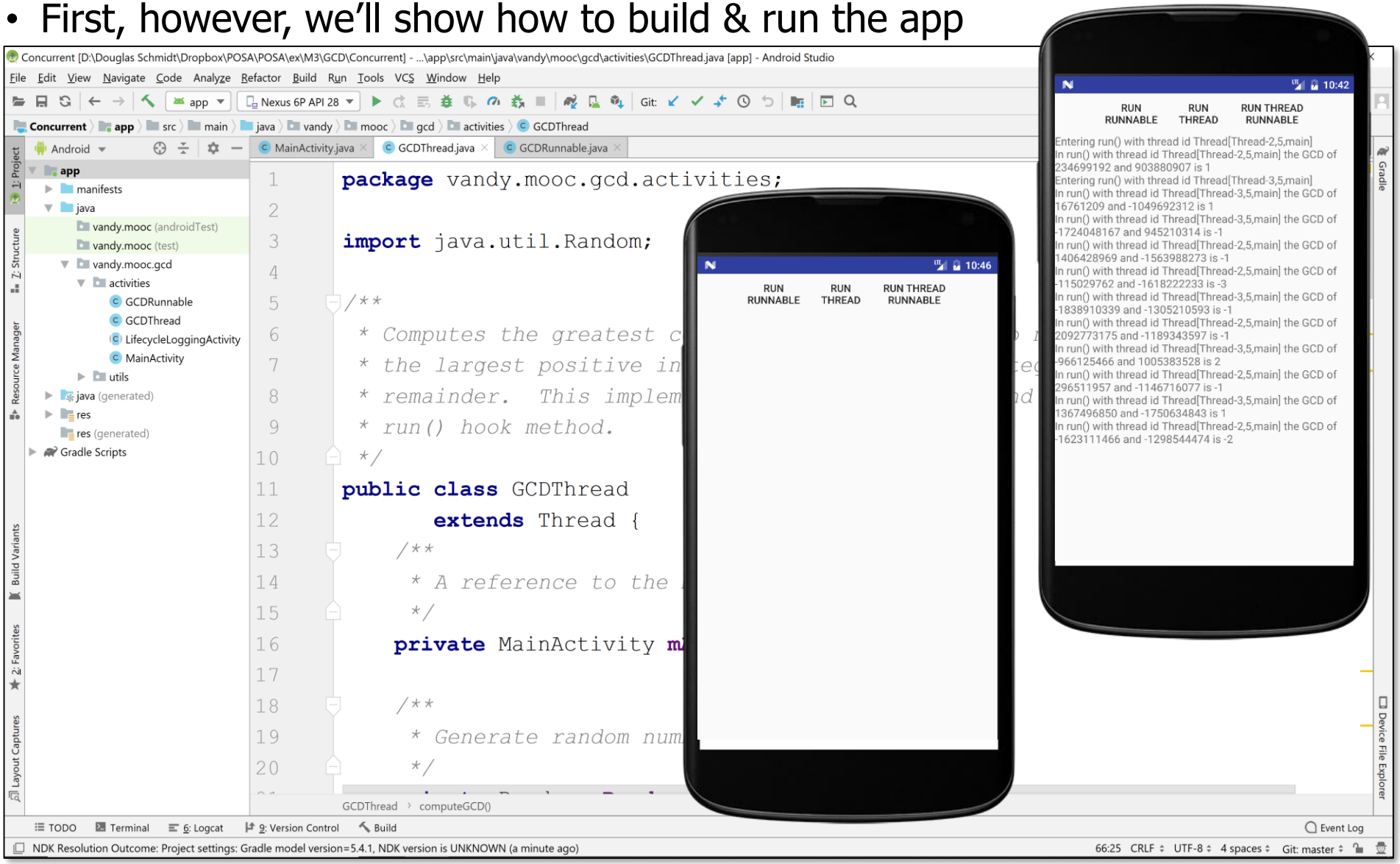
```
/**
 * Computes the greatest common divisor (GCD) of two numbers, which is
 * the largest positive integer that divides two integers without a
 * remainder.  This implementation extends Random and implements the
 * Runnable interface's run() hook method.
 */
public class GCDRunnable
       extends Random // Inherits random number generation capabilities.
       implements Runnable {
    /**
     * A reference to the MainActivity.
     */
    private final MainActivity mActivity;

    /**
     * Number of times to iterate, which is 100 million to ensure the
     * program runs for a while.
     */
    private final int MAX_ITERATIONS = 100000000;

    /**
     * Number of times to iterate before calling print, which is 10
     * million to ensure the program runs for a while.
     */
    private final int MAX_PRINT_ITERATIONS = 10000000;

    /**
     * Hook method that runs for MAX_ITERATIONs computing the GCD of
     * randomly generated numbers.
     */
    public void run() {
        final String threadString = " with thread id " + Thread.currentThread();

        mActivity.println("Entering run()" + threadString);

        // Generate random numbers and compute their GCDs.

        for (int i = 0; i < MAX_ITERATIONS; ++i) {
            // Generate two random numbers.
            int number1 = nextInt();
            int number2 = nextInt();

            // Print results every 10 million iterations.
            if ((i % MAX_PRINT_ITERATIONS) == 0)
                mActivity.println("In run()"
                                  + threadString
                                  + " the GCD of "
                                  + number1
                                  + " and "
                                  + number2
                                  + " is "
                                  + computeGCD(number1,
                                               number2));
        }

        mActivity.println("Leaving run() " + threadString);
    }
```

```
/**
 * Computes the greatest common divisor (GCD) of two numbers, which is
 * the largest positive integer that divides two integers without a
 * remainder.  This implementation extends Thread and overrides its
 * run() hook method.
 */
public class GCDThread
       extends Thread {
    /**
     * A reference to the MainActivity.
     */
    private MainActivity mActivity;

    /**
     * Generate random numbers.
     */
    private Random mRandom;

    /**
     * Number of times to iterate, which is 100 million to ensure the
     * program runs for a while.
     */
    private final int MAX_ITERATIONS = 100000000;

    /**
     * Number of times to iterate before calling print, which is 10
     * million to ensure the program runs for a while.
     */
    private final int MAX_PRINT_ITERATIONS = 10000000;

    /**
     * Hook method that runs for MAX_ITERATIONs computing the GCD of
     * randomly generated numbers.
     */
    public void run() {
        final String threadString = " with thread id " + Thread.currentThread();

        mActivity.println("Entering run()" + threadString);

        // Generate random numbers and compute their GCDs.

        for (int i = 0; i < MAX_ITERATIONS; ++i) {
            // Generate two random numbers.
            int number1 = mRandom.nextInt();
            int number2 = mRandom.nextInt();

            // Print results every 10 million iterations.
            if ((i % MAX_PRINT_ITERATIONS) == 0)
                mActivity.println("In run()"
                                  + threadString + " the GCD of "
                                  + number1 + " and " + number2 + " is "
                                  + computeGCD(number1,
                                               number2));
        }

        mActivity.println("Leaving run() " + threadString);
    }
```

See github.com/douglascraigschmidt/POSA/tree/master/ex/M3/GCD/Concurrent

Design of the GCD Concurrent App

- First, however, we'll show how to build & run the app



End of Java Thread: Overview of the Case Study App