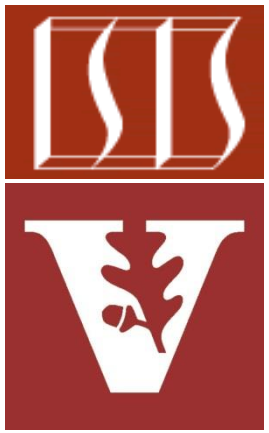
Java Thread:
Ways of Giving Code to a Thread

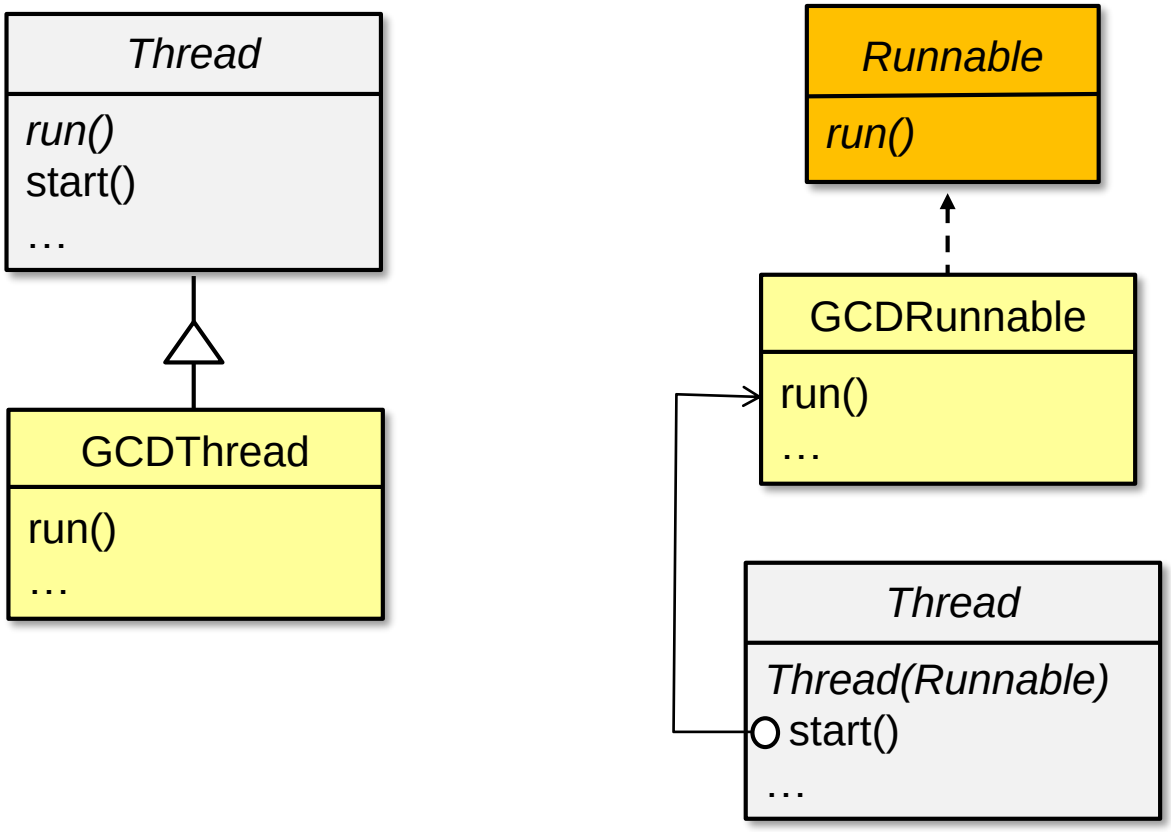


Douglas C. Schmidt
d.schmidt@vanderbilt.edu
www.dre.vanderbilt.edu/~schmidt

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

Learning Objectives in this Part of the Lesson

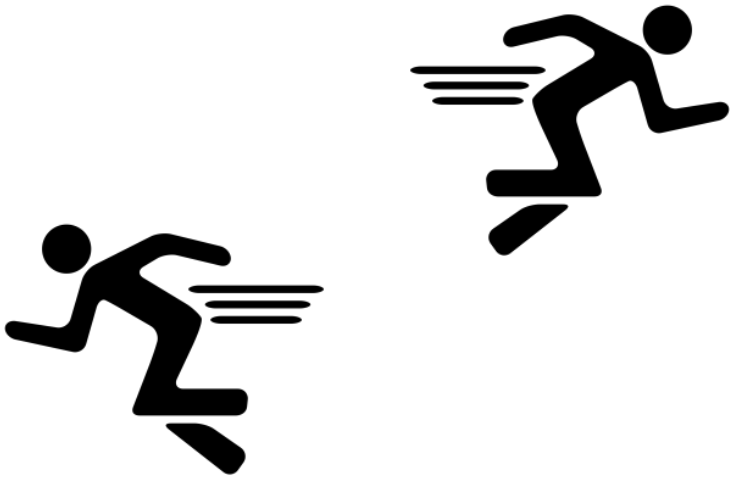
- Understand how Java threads support concurrency
- Learn how our case study app works
- Know alternative ways of giving code to a thread

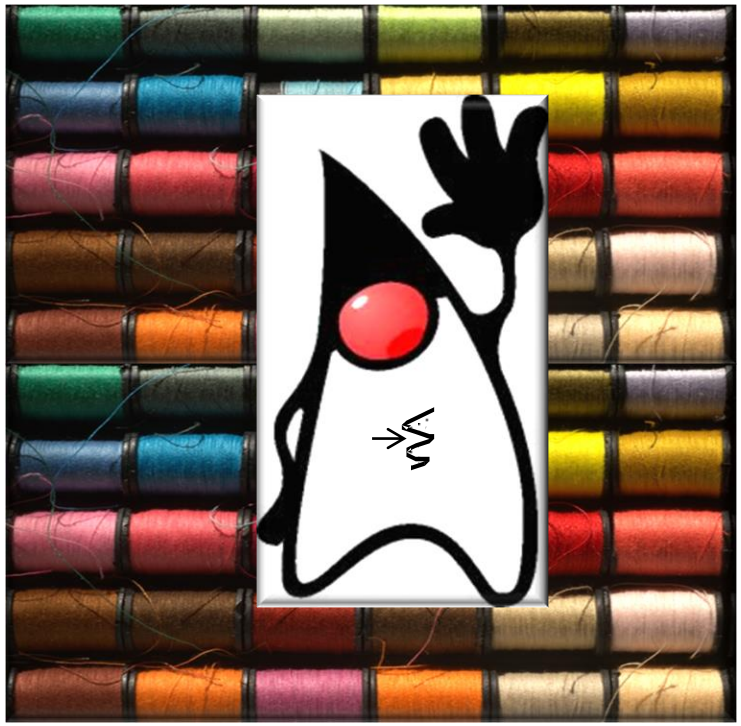


Ways of Giving Code to Java Threads

Ways of Giving Code to Java Threads

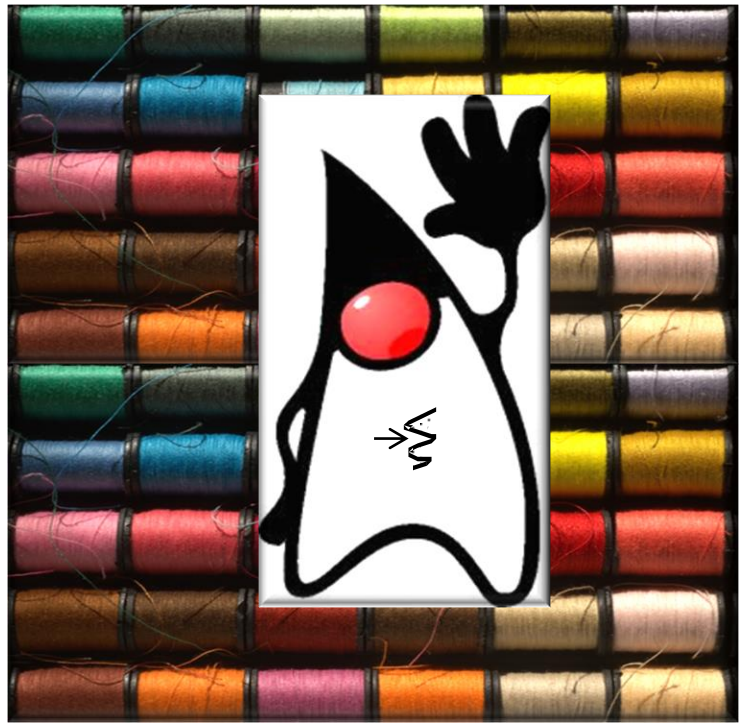
- Java threads *must* be given code to run

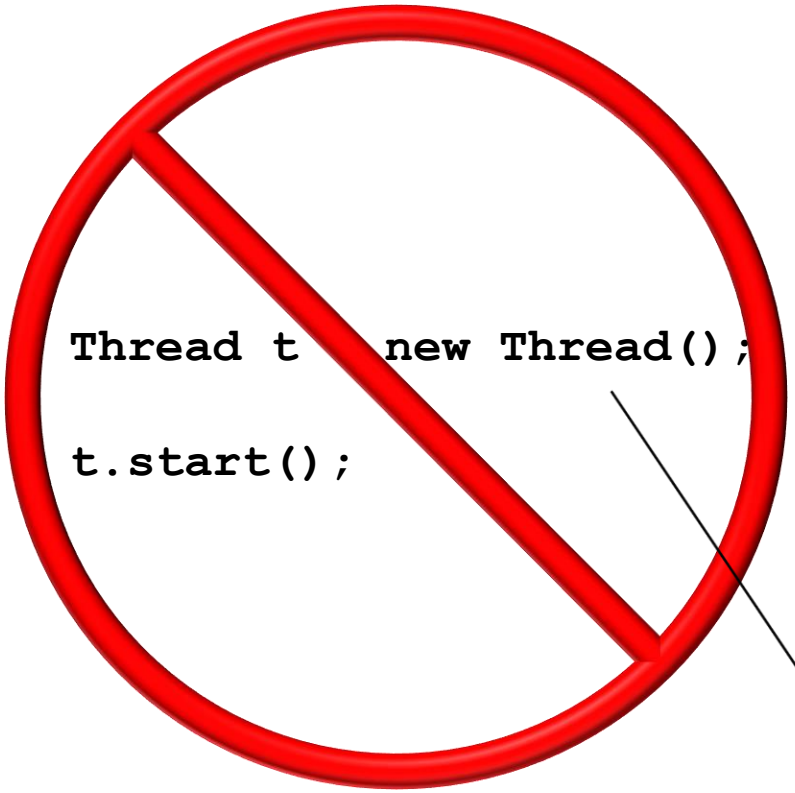




Ways of Giving Code to Java Threads

- Java threads *must* be given code to run



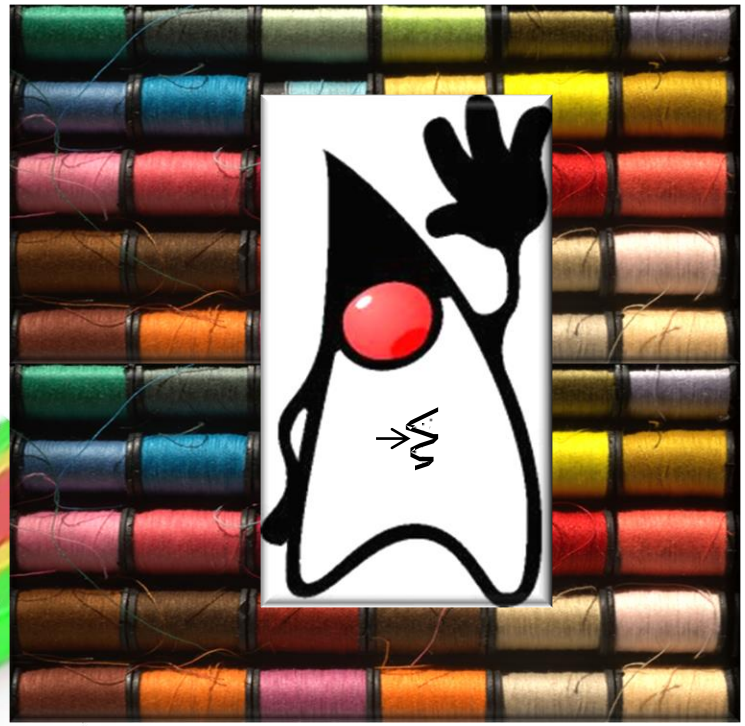


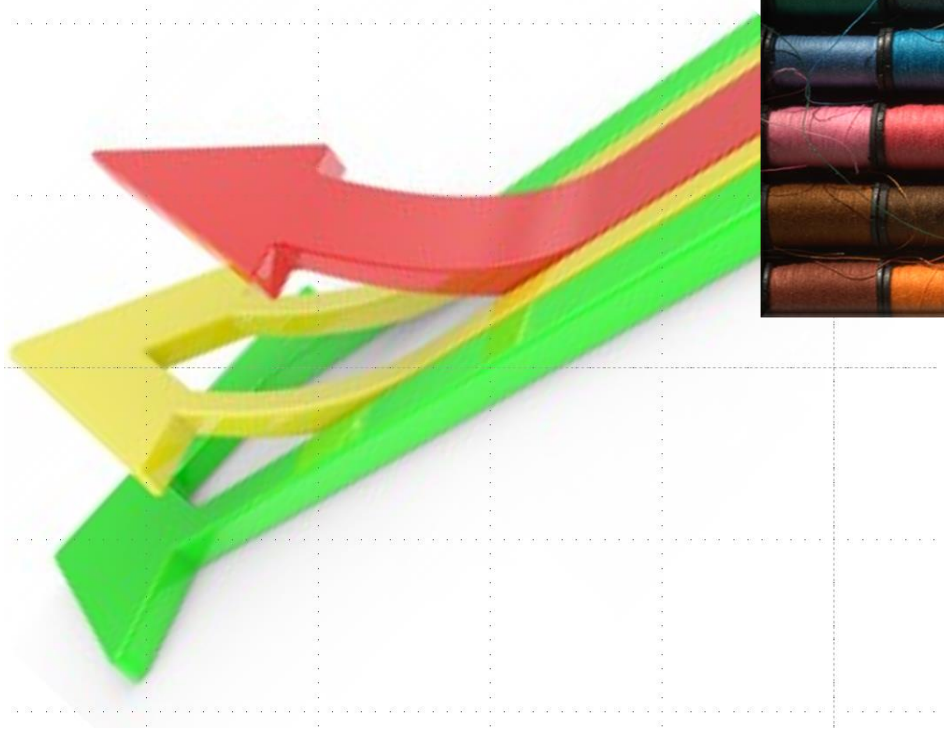
Do not use the "no argument" Thread constructor directly!!!

See stackoverflow.com/questions/7572527/why-would-anyone-ever-use-the-java-thread-no-argument-constructor

Ways of Giving Code to Java Threads

- Java threads *must* be given code to run

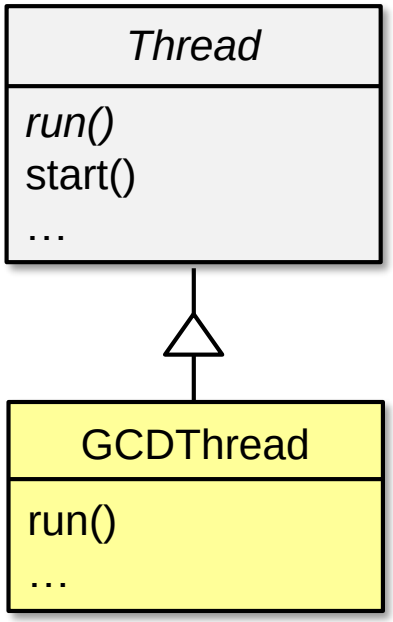




There are alternative ways to give code to Java threads

Ways of Giving Code to Java Threads

- Java threads *must* be given code to run, e.g.
  1. Extend the Thread class



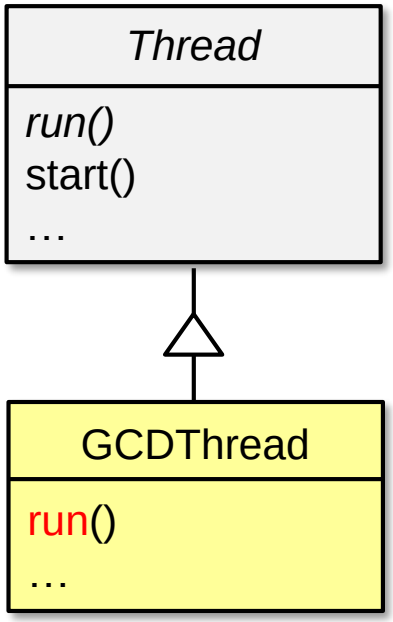

```
public class GCDThread
                extends Thread {
    public void run() {
        // code to run goes here
    }
}
```

See docs.oracle.com/javase/8/docs/api/java/lang/Thread.html

Ways of Giving Code to Java Threads

- Java threads *must* be given code to run, e.g.

  1. Extend the Thread class



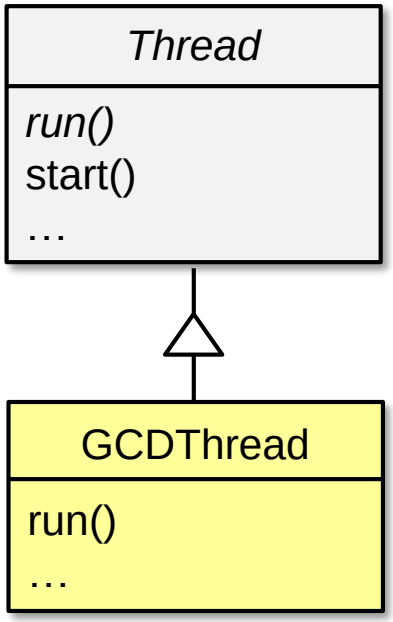

```
public class GCDThread
                extends Thread {
    public void run() {
        // code to run goes here
    }
}
```

Override the run() hook method in the subclass & define the thread's computations

See wiki.c2.com/?HookMethod

Ways of Giving Code to Java Threads

- Java threads *must* be given code to run, e.g.
  1. Extend the Thread class



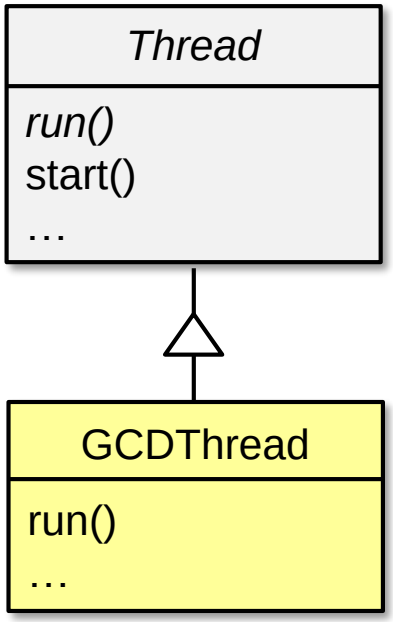

```
public class GCDThread
                extends Thread {
    public void run() {
        // code to run goes here
    }
}

Thread gCDThread =
  new GCDThread();
gCDThread.start();
```

Create & start a thread using a named subclass of Thread

Ways of Giving Code to Java Threads

- Java threads *must* be given code to run, e.g.
  1. Extend the Thread class



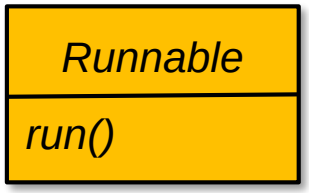

```
public class GCDThread
                extends Thread {
    public void run() {
        // code to run goes here
    }
}

new GCDThread().start();
```

You can also write a one-liner to create & start an anonymous thread

Ways of Giving Code to Java Threads

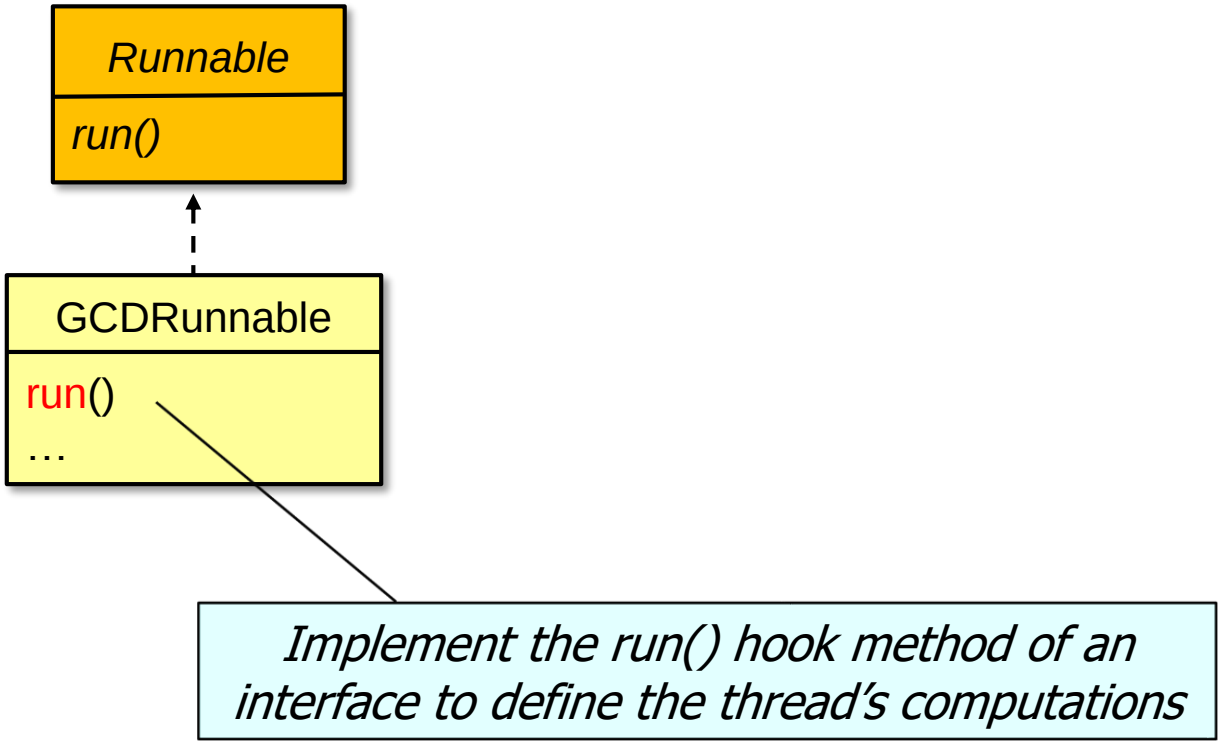
- Java threads *must* be given code to run, e.g.
  1. Extend the Thread class
  2. Implement the Runnable interface



See docs.oracle.com/javase/8/docs/api/java/lang/Thread.html

Ways of Giving Code to Java Threads

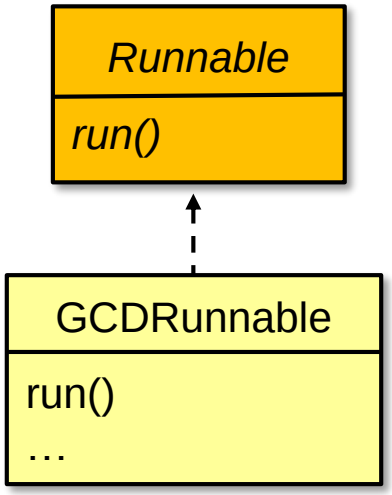
- Java threads *must* be given code to run, e.g.
  1. Extend the Thread class
  2. Implement the Runnable interface

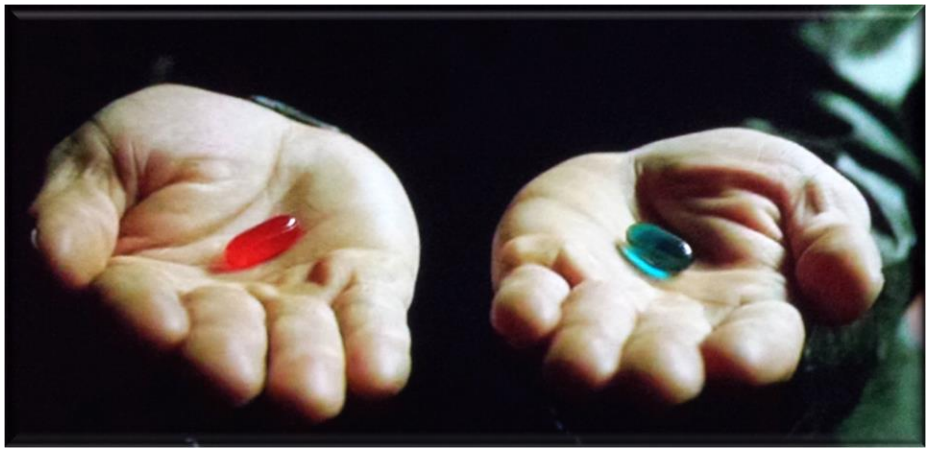


See docs.oracle.com/javase/8/docs/api/java/lang/Runnable.html

Ways of Giving Code to Java Threads

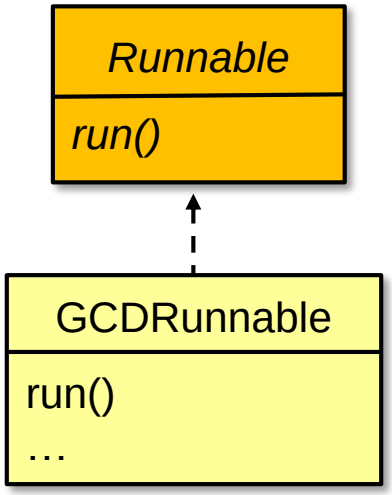
- Java threads *must* be given code to run, e.g.
  1. Extend the Thread class
  2. Implement the Runnable interface





Ways of Giving Code to Java Threads

- Java threads *must* be given code to run, e.g.
  1. Extend the Thread class
  2. Implement the Runnable interface



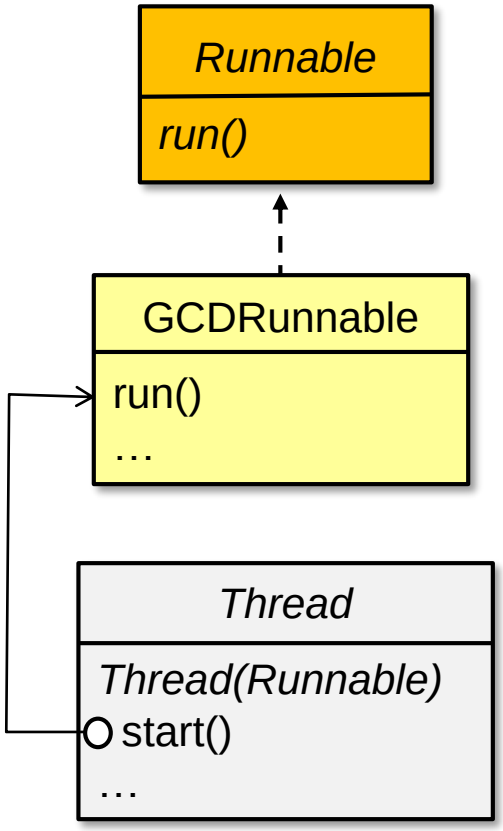

```
public class GCDRunnable
        implements Runnable {
  public void run() {
    // code to run goes here
  }
}

Runnable gCDRunnable =
  new GCDRunnable();
```

Create an instance of a named class as the runnable

Ways of Giving Code to Java Threads

- Java threads *must* be given code to run, e.g.
  1. Extend the Thread class
  2. Implement the Runnable interface



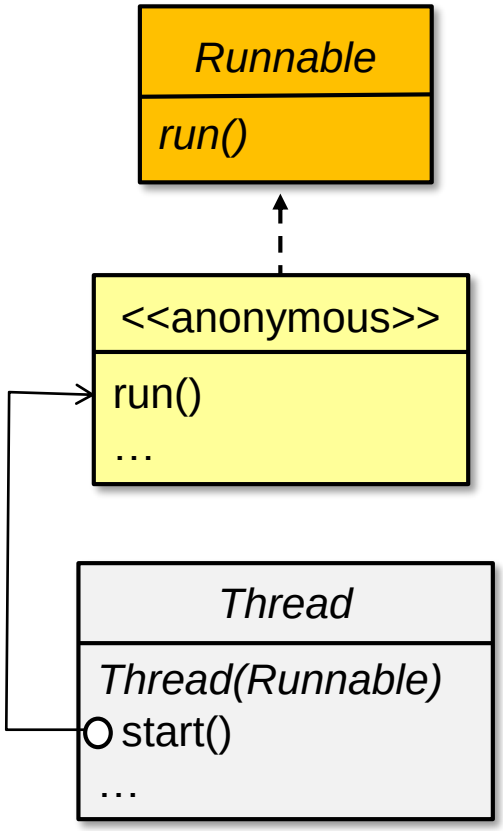

```
public class GCDRunnable
        implements Runnable {
  public void run() {
    // code to run goes here
  }
}

Runnable gCDRunnable =
  new GCDRunnable();
new Thread(gCDRunnable).start();
```

Pass that runnable to a new thread object & start it

Ways of Giving Code to Java Threads

- Java threads *must* be given code to run, e.g.
  1. Extend the Thread class
  2. Implement the Runnable interface

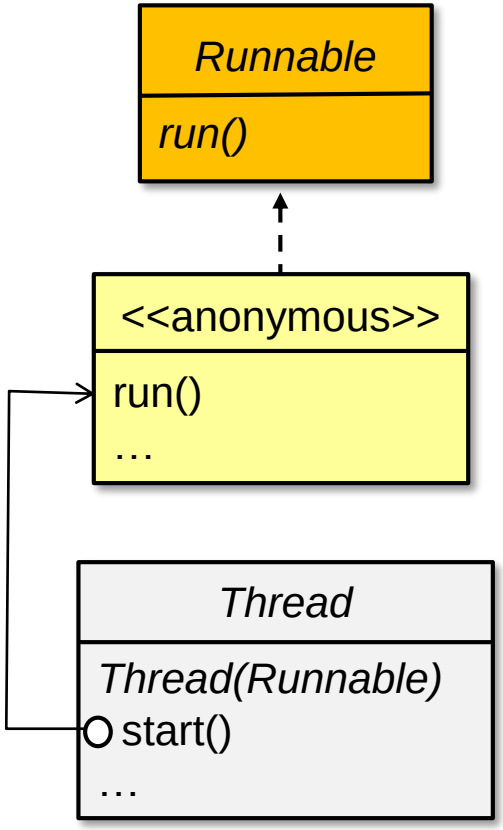



```
new Thread(new Runnable() {
    public void run(){
      // code to run goes here
  }
}).start();
```

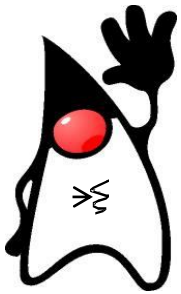
Create & start a thread by using an anonymous inner class as the runnable

Ways of Giving Code to Java Threads

- Java threads *must* be given code to run, e.g.
  1. Extend the Thread class
  2. Implement the Runnable interface



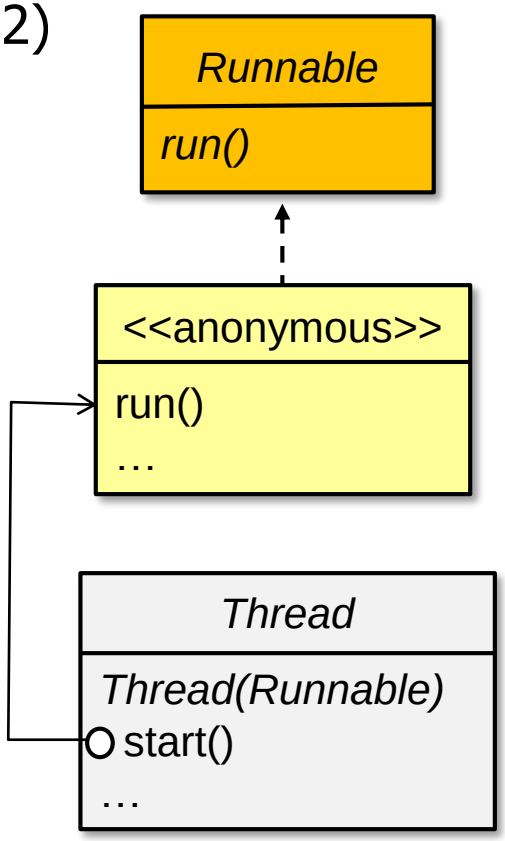

```
new Thread(new Runnable() {
    public void run(){
      // code to run goes here
  }
}).start();
```



This anonymous inner class idiom is used extensively in older Java & Android code but is tedious to program..

Ways of Giving Code to Java Threads

- Java threads *must* be given code to run, e.g.
  1. Extend the Thread class
  2. Implement the Runnable interface
  3. Use Java 8 lambda expressions (variant of #2)



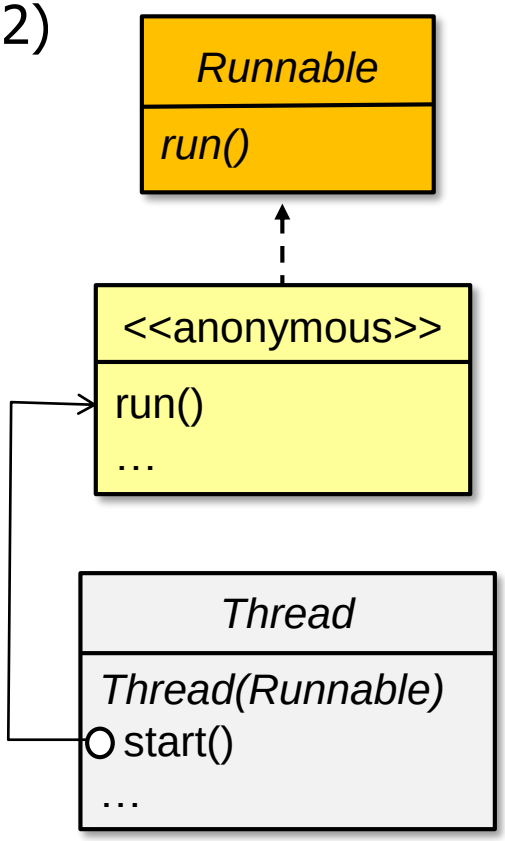

```
new Thread(() -> {
        // code to run goes here
}).start();
```

A lambda expression is an unnamed block of code (with optional parameters) that can be passed around & executed later

See www.drdobbs.com/jvm/lambda-expressions-in-java-8/240166764

Ways of Giving Code to Java Threads

- Java threads *must* be given code to run, e.g.
  1. Extend the Thread class
  2. Implement the Runnable interface
  3. Use Java 8 lambda expressions (variant of #2)

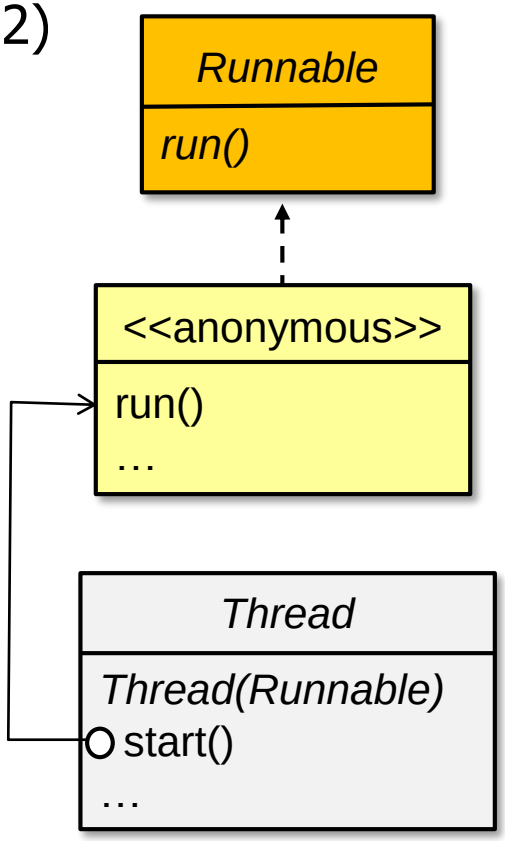



```
new Thread(() -> {
        // code to run goes here
}).start();
```

This approach is unwieldy if the code to run is long, complex, or needs to be used multiple times!

Ways of Giving Code to Java Threads

- Java threads *must* be given code to run, e.g.
  1. Extend the Thread class
  2. Implement the Runnable interface
  3. Use Java 8 lambda expressions (variant of #2)




```
Runnable r = () -> {
   // code to run goes here
};

new Thread(r).start();
```

You can therefore store the runnable in a variable & pass it to the Thread constructor

End of Java Thread: Ways of Giving Code to a Thread