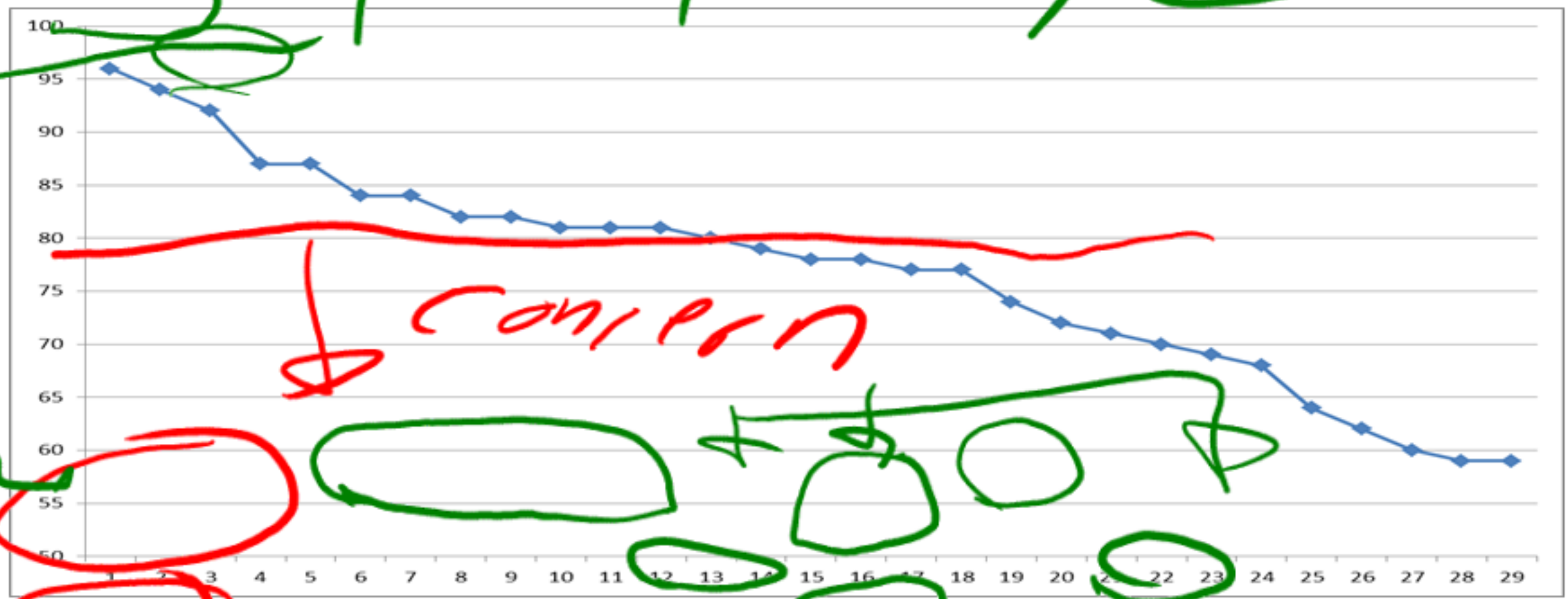
Exam Status



77%	Average	82%	89%	68%	71%	77%	65%
78%	Median	85%	100%	67%	75%	75%	69%
10%	STD	12%	22%	37%	20%	12%	27%
96.00%	Max	100.00%	100.00%	100.00%	100.00%	100.00%	125.00%
59.00%	Min	60.00%	33.33%	0.00%	33.33%	50.00%	18.75%

Starting

Lets take a look at a Java observer implementation

Our friend the crosswalk…

CW
observer

update

CWC

Subscribe
unsubscribe

Traffic
light
observer

update

Interface

TLC

Subscribe
unsubscribe

UI

State
Machine

Java Multithreading

Objectives

- Define thread
- Explain the purpose for the Runnable Interface
- Draw the representation for the stack when multiple threads are executing
- Implement a simple multi-threaded application in Java
- Draw the thread lifecycle
- Explain how threads are indicated by the design of a state machine.

Embedded Systems Example

- I want to blink a light on the ATMEGA32 board ~~once~~ per second.
  - How would I do this?

Timer that fires an interrupt every second.

Example 2: I want to blink a light at a 1 Hz rate as well as send out the A/D reading over the serial port

Timer fires once per second ⇒ AD?

⇒ Serial Comm?

25 LEDs blinking @ different rates

Java Example

- I want to write a program which will countdown once per second from n (where n is passed on the command line) to 0 before saying “Liftoff”.

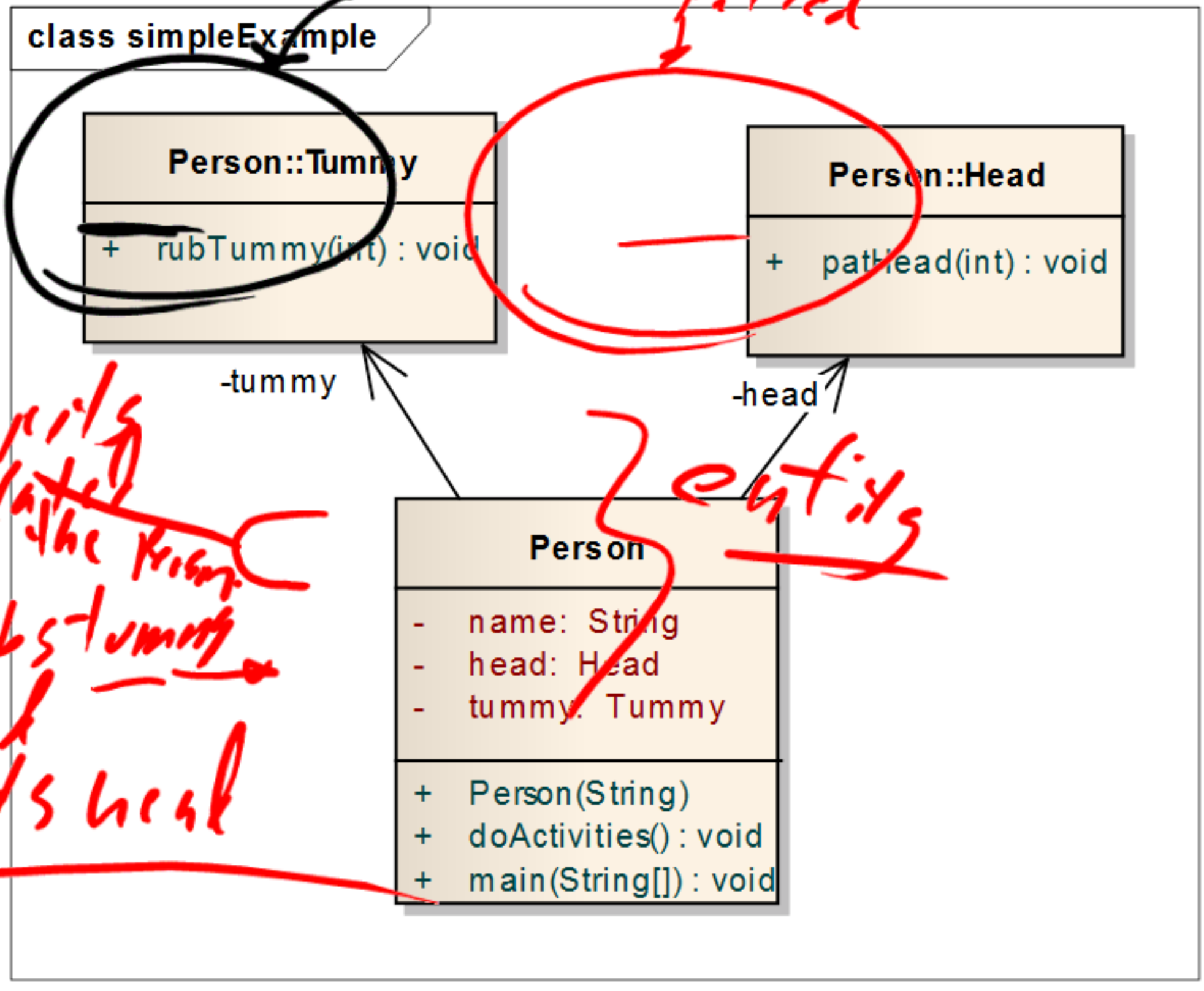
```java
/**
 * This class will provide a method that will count down from n to 0, once per
 * second.
 *
 * @author schilling
 *
 */
public class CountdownTimer {
    public static void main(String args[]) {
        // Determine what number to start at when counting.
        int loopCount = Integer.parseInt(args[0]);

        // Count down to 0.
        while (loopCount > 0) {
            System.out.println("" + loopCount + "....");
            loopCount--;
            try {
                Thread.sleep(1000);
            } catch (InterruptedException e) {
                // No used / supported
            }
        }
        System.out.println("Liftoff");
    }
}
```

Rub Your Tummy Lyrics

- Rub your tummy, just like this.
  Rub it all day long!
  Rub your tummy, rub it hard,
  While we sing our song.
  Rub, rub, rubba, rub
  Rub, rub, rubba, rub
  Rub, rub, rubba, rub
  Rub, rub, rubba, rub

- Rub
  Pay and sing along.
  Pat your head, just like this.
  Pat it all day lon!
  Pat your head, but not too hard,
  And sing our silly song
  Pat Pat Patta Pat
  Pat Pat Patta Pat
  Pat Pat Patta Pat
  Pat Pat Patta Pat
  Pat

UML Class Diagram



Lets look at the code

- What are the problems with this code?

Problems

Does lox in a row the switches ⇒ Not multitasking.

Rub tummy Pat head 3 Simultaneously

Things we Care About

- Thread Class
  - A thread objects represents a thread of execution in a program.
- Runnable Interface
  - interface is implemented by those classes whose instances are supposed to be executed in a different thread.
  - has only one method 'run'
    - which takes no arguments

Java Thread Interface:

What's wrong with this diagram?

"a main" for the thread.

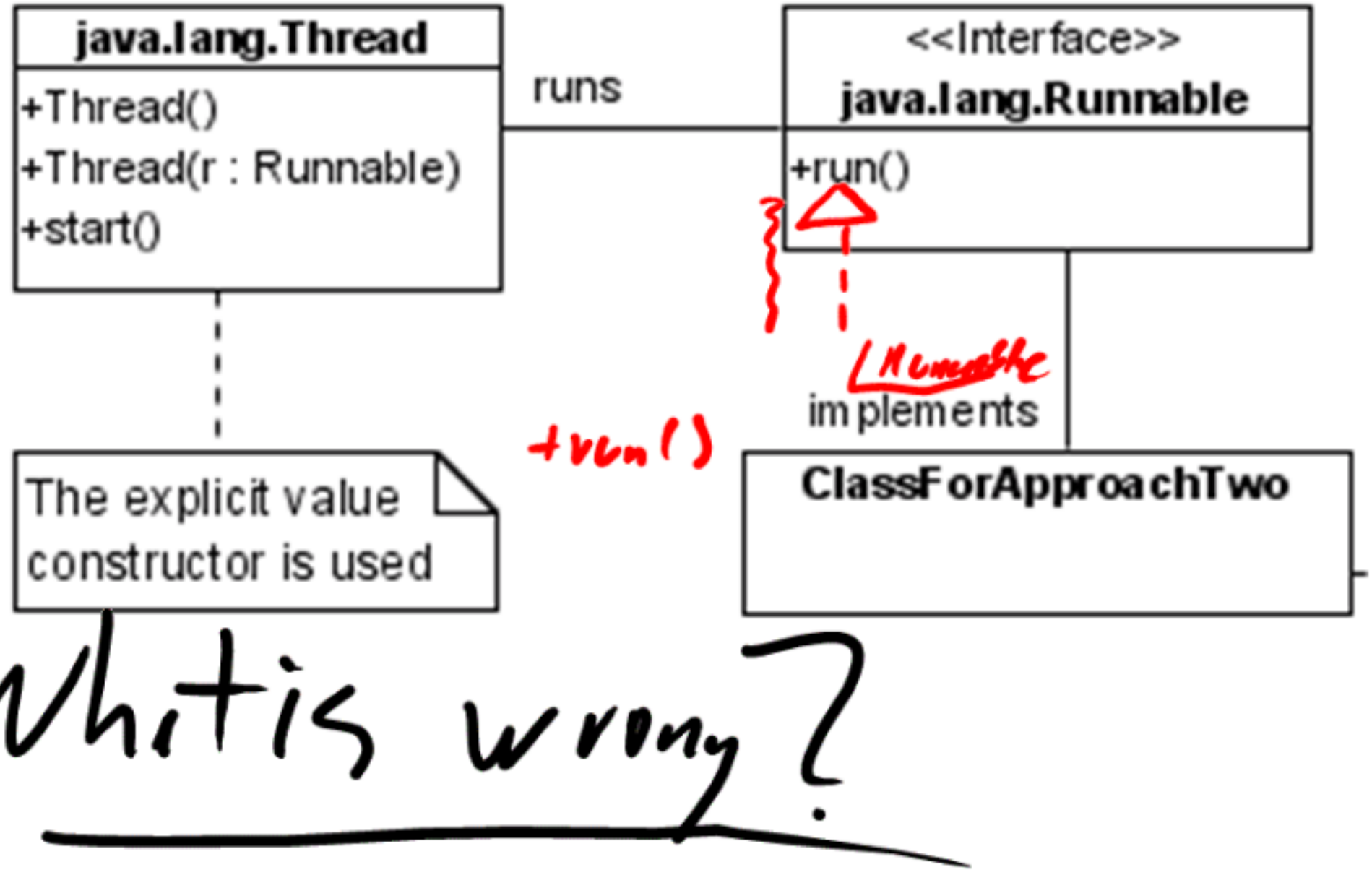
Keeps track of PC / stack

Defines how we run a thread

Causes a thread to (r.main) running calling its run.

Java Thread Interface:

What's wrong with this diagram?



How does a multi-threaded application start up?

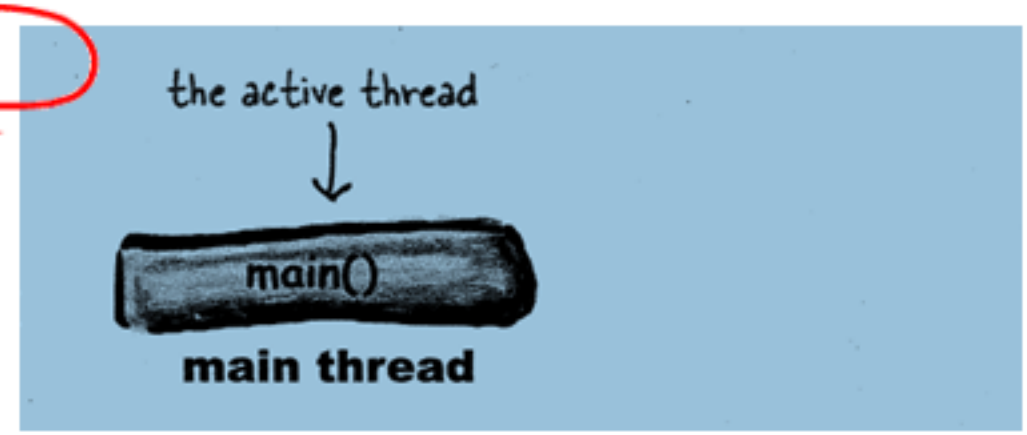
⇒ Starts @ main

↳ main will spawn new threads.

How does a multi-threaded application start up?

- JVM calls the main method

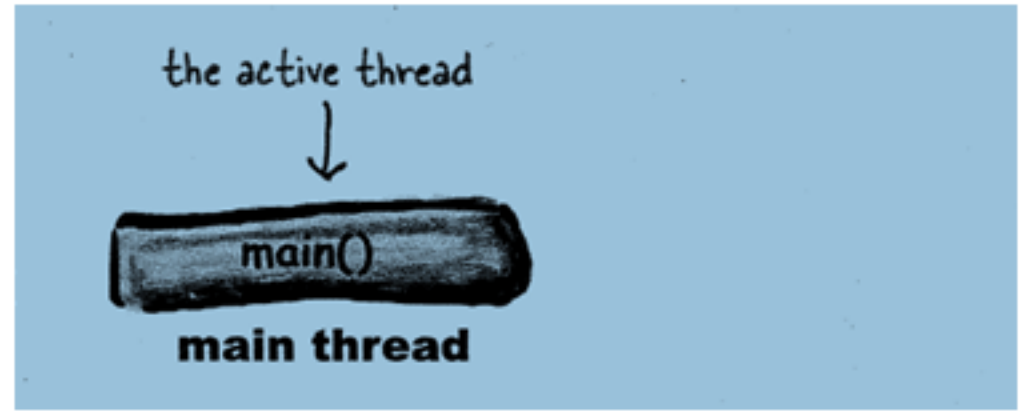
main
gets
invoked.



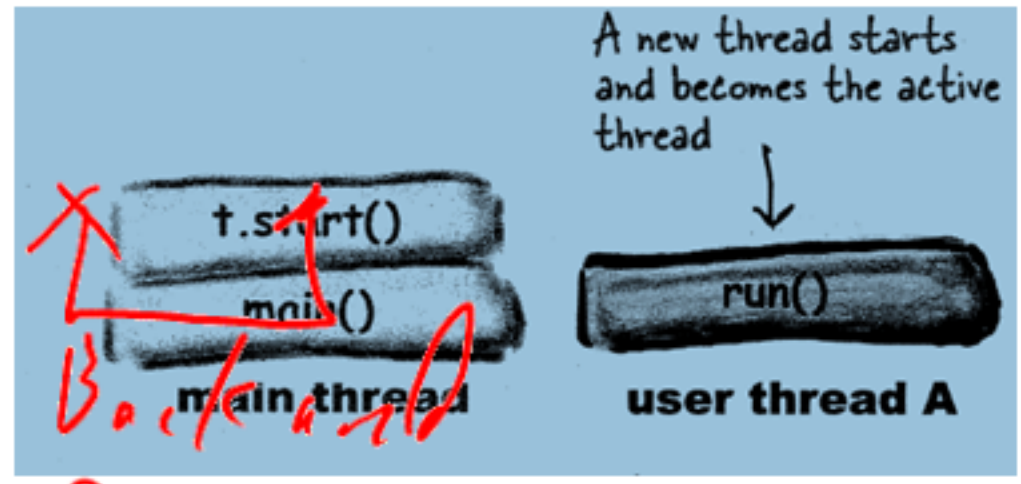
Single
thread
process.

How does a multi-threaded application start up?

- JVM calls the main method



- Main spawns a new thread



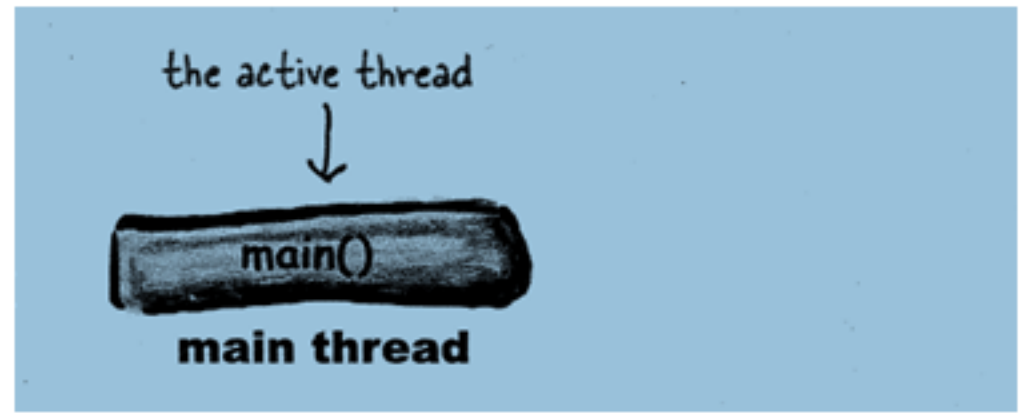
causes the run of the thread to be executed.

Backend

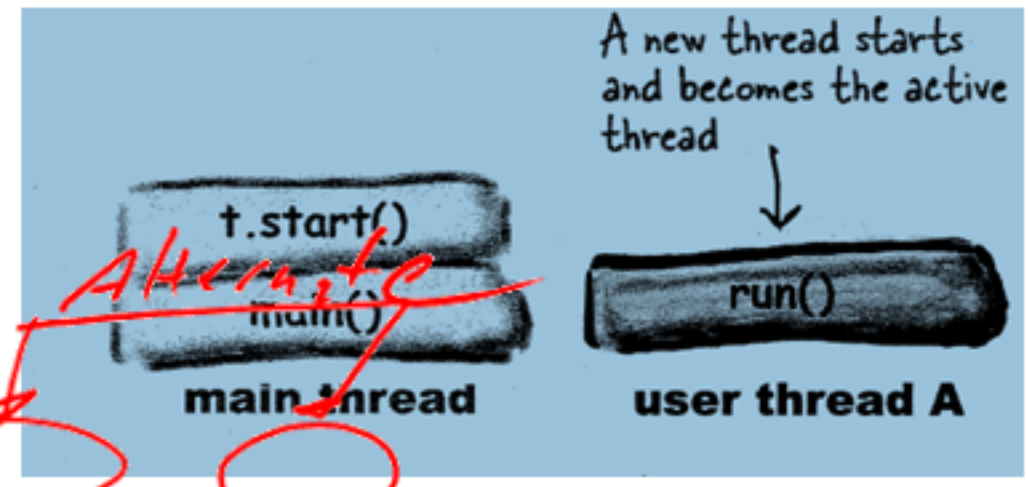
fifth

How does a multi-threaded application start up?

- JVM calls the main method

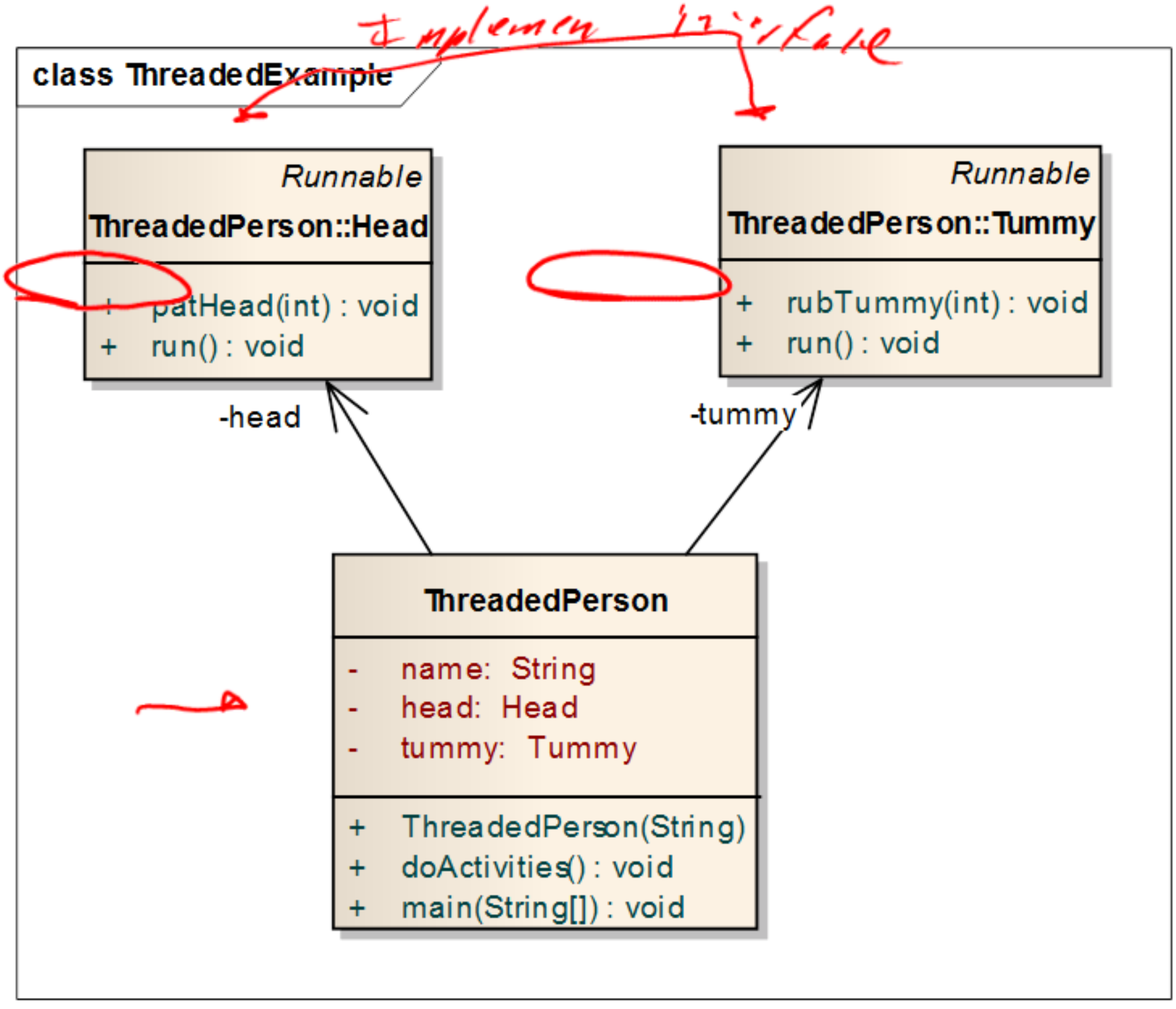


- Main spawns a new thread



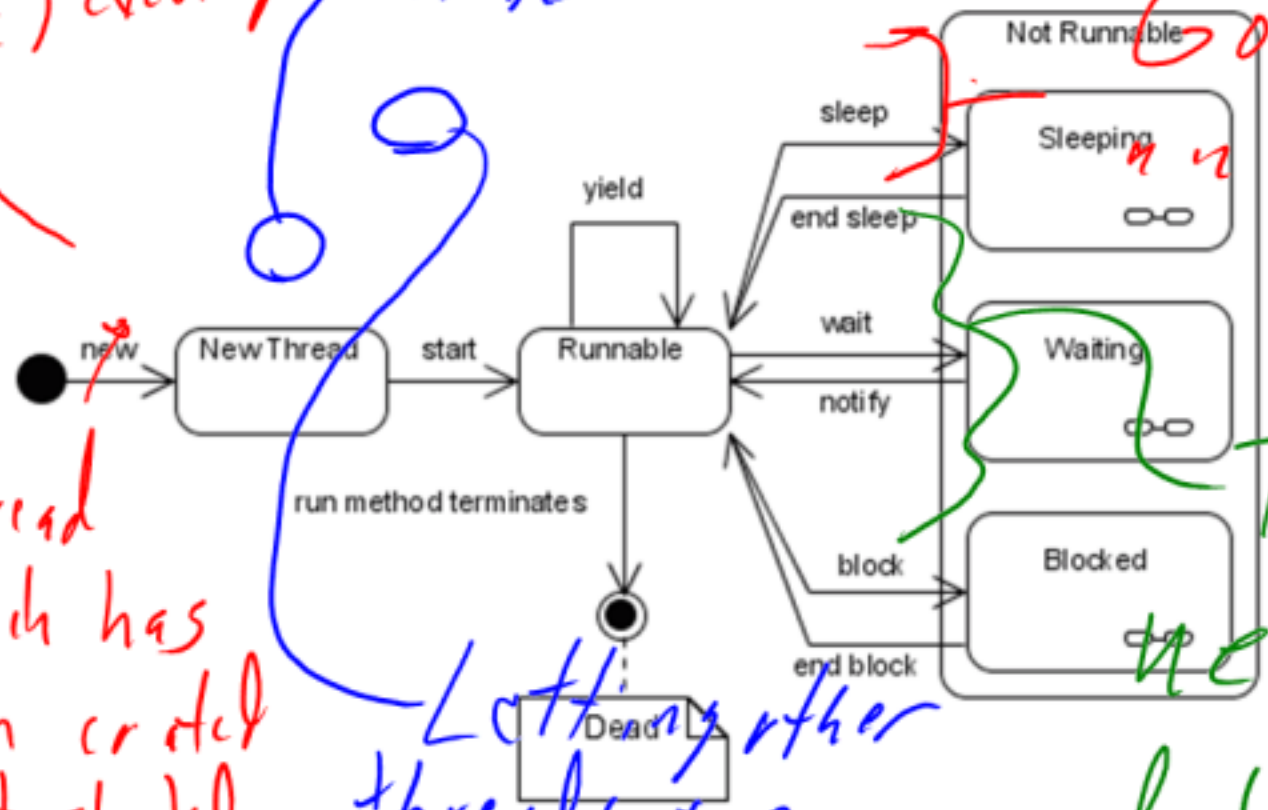
- JVM switches back and forth between threads

Rub Tummy and Pat Head Multithreaded style

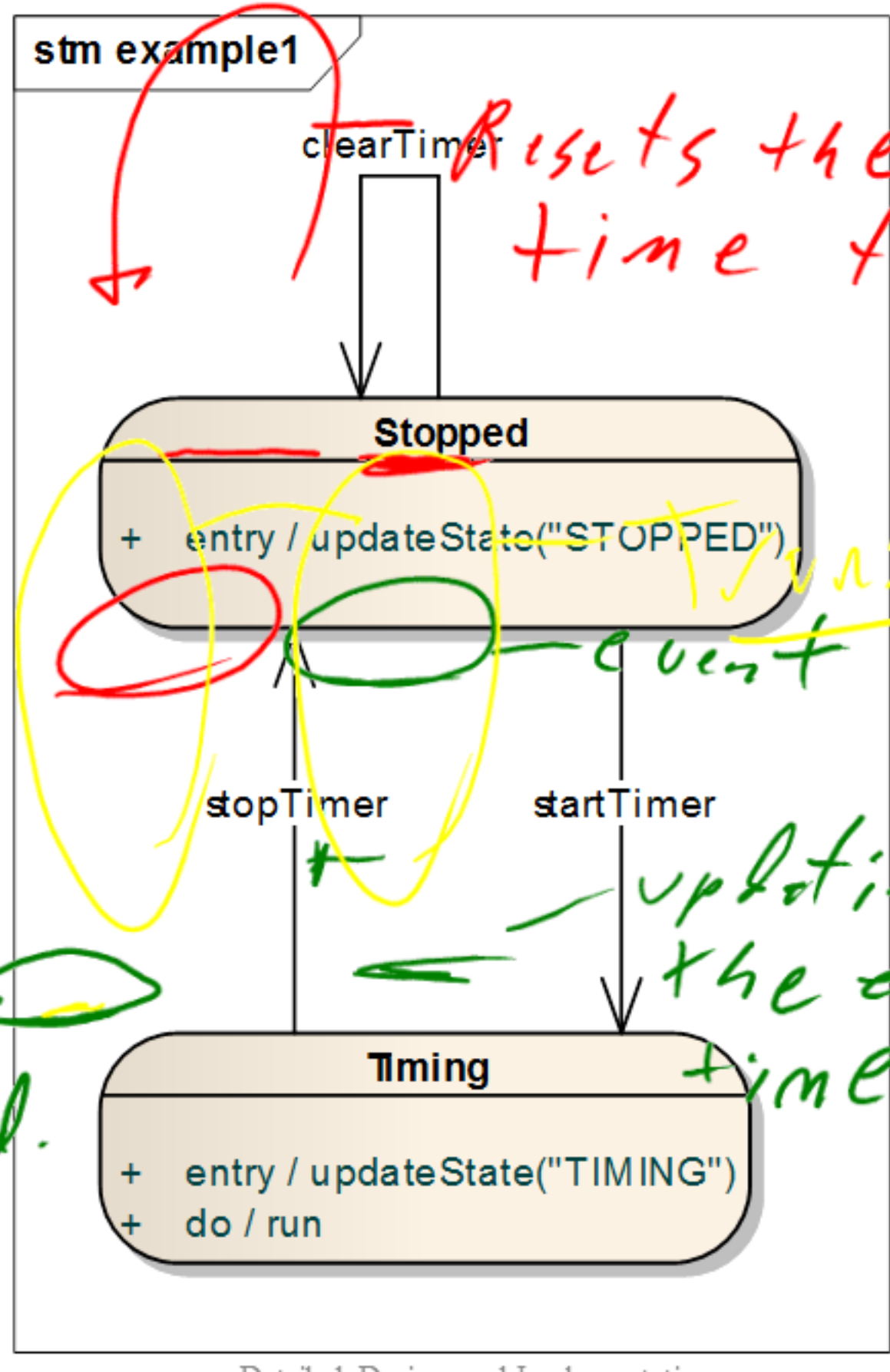


Lets look at the code

The thread lifecycle



A simple stopwatch



class example1

«interface,observer»
StopwatchObserverInterface

+ updateTime(int) : void
+ updateState(String) : void

«interface»
StopwatchSubjectInterface

+ Subscribe(StopwatchObserverInterface) : void
+ unsubscribe, subjectInterface(StopwatchObserverInterface) : void

-observers

JFrame

«Real Observer»
StopWatchUI

- serialVersionUID: long = 1L {readOnly}
- timeDisplay: JTextField
- state: JTextField
- ticker: TimeTicker

+ updateTime(int) : void
+ StopWatchUI()
+ main(String[]) : void
+ updateState(String) : void

-ticker

Runnable

«Real Subject»
TimeTicker

- keepRunning: boolean = true
- timing: boolean = false
- tmrThread: Thread
- time: int
- observers: List<StopwatchObserverInterface> = new LinkedList<...

+ updateTime(int) : void
+ updateState(String) : void
+ run() : void
+ clearTimer() : void
+ startTimer() : void
+ stopTimer() : void
+ Subscribe(StopwatchObserverInterface) : void
+ ubsubscribe(StopwatchObserverInterface) : void