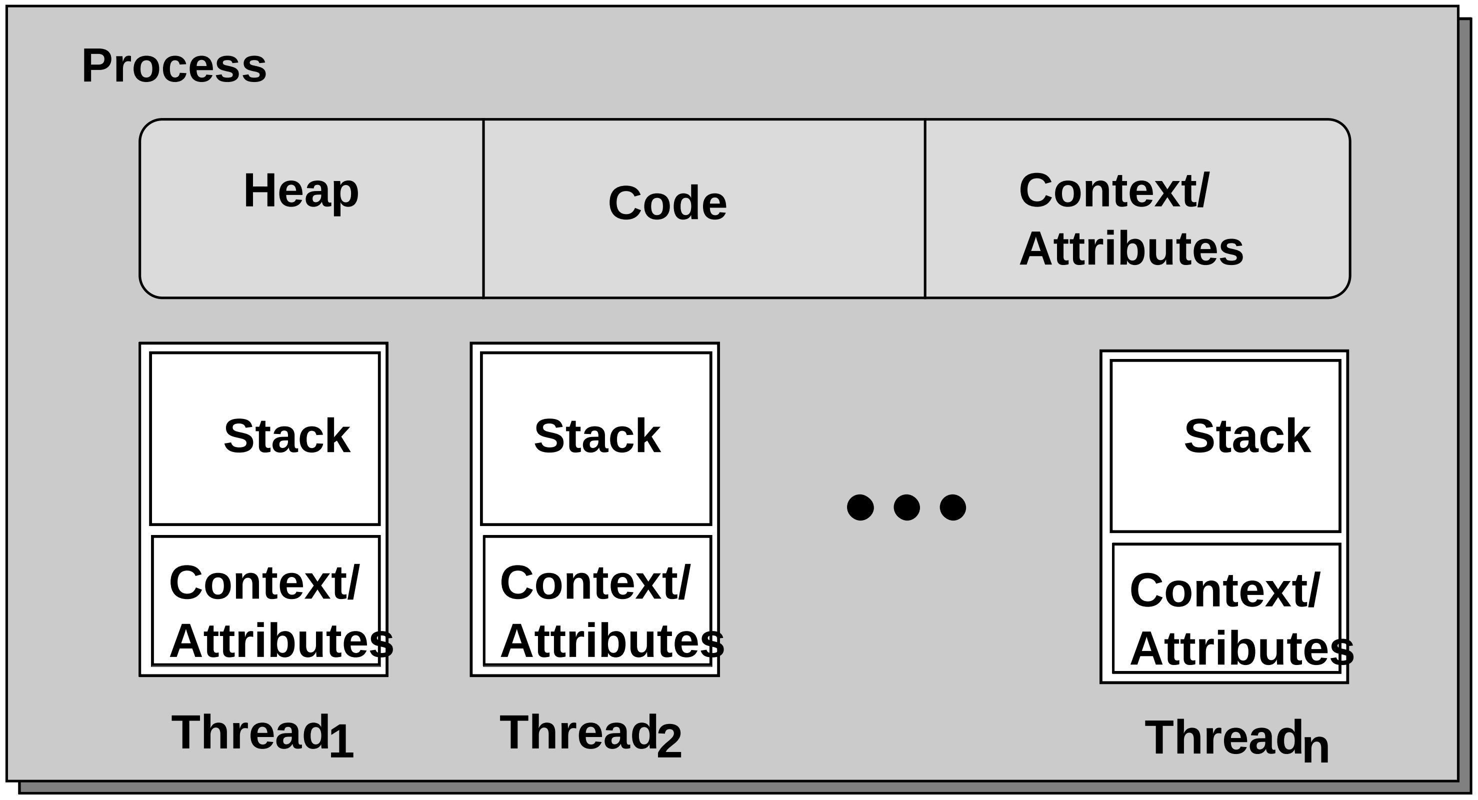
C340 Concurrency: Concurrency in Java

Wolfgang Emmerich

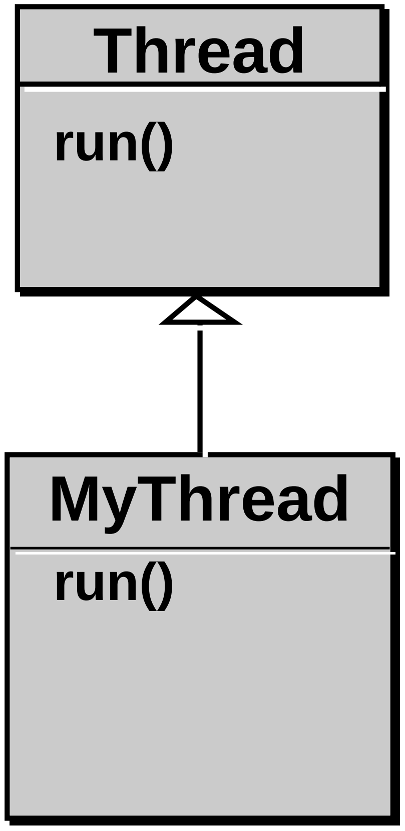
Mark Levene

Threads and OS Processes



- ***OS process provides protected address space.***
- ***Many threads may execute within space.***
- ***Each thread: stack & context (saved registers).***

Threads using Inheritance

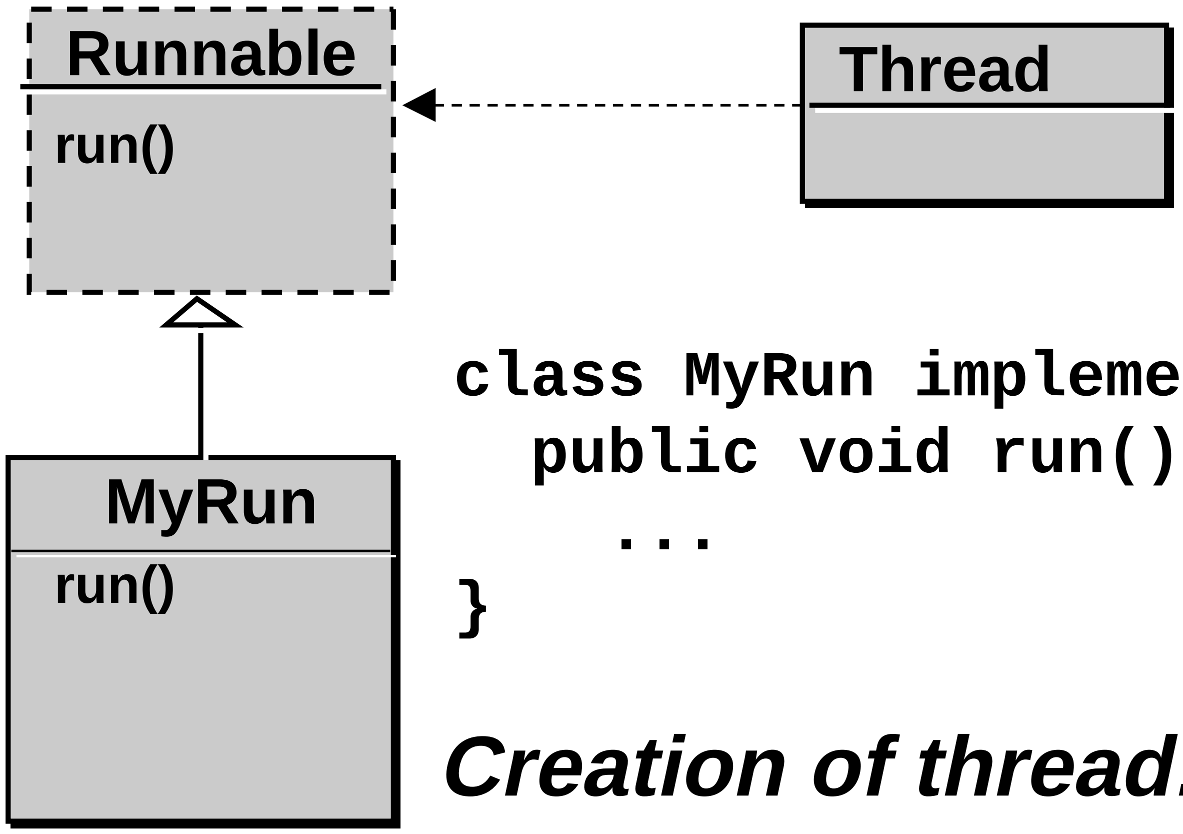


```
class MyThread extends Thread {
  public void run() {
    ...
}
```

Creation of thread:

```
MyThread t=new MyThread();
```

Threads implementing Interfaces



```
class MyRun implements Runnable {
  public void run() {
    ...
}
```

Creation of thread:

```
Thread t=new Thread(new MyRun);
```

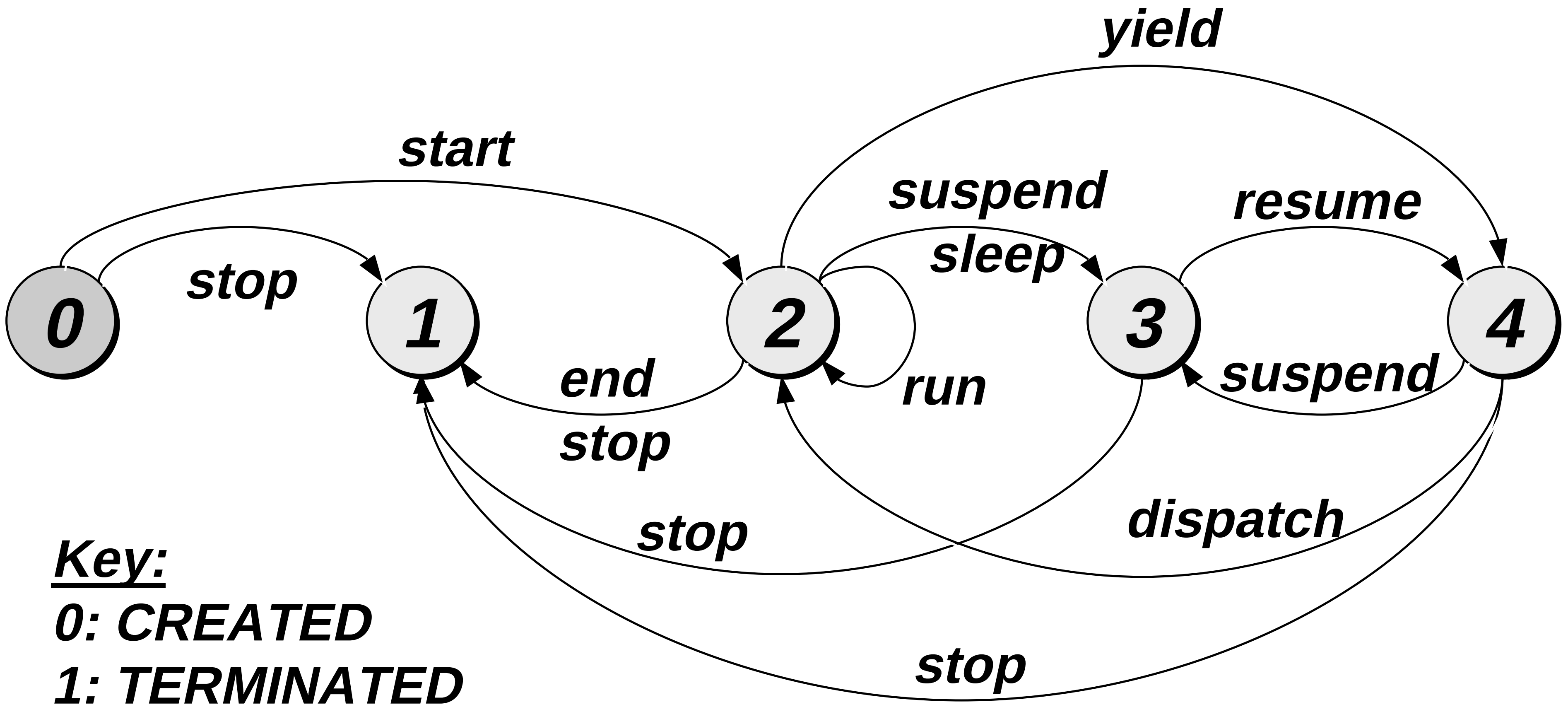
Thread Lifecycle

- ***Started by start() which invokes run()***
- ***Terminated when***
  - ***run() returns or***
  - ***explicitly terminated by stop().***
- ***A started thread may be***
  - ***running or***
  - ***runnable (waiting to be scheduled)***
- ***Thread gives up processor using yield().***
- ***A thread may be suspended by suspend()***
- ***If Suspended gets runnable by resume().***
- ***sleep() suspends for a given time and then resumes***

FSP Model of Java Thread Lifecycle

```
THREAD = CREATED,
CREATED = ( start -> RUNNING
          | stop -> TERMINATED),
RUNNING = ( {suspend,sleep}-> NON_RUNNABLE
          | yield -> RUNNABLE
          |{stop, end} ->TERMINATED
          | run -> RUNNING),
RUNNABLE= ( suspend -> NON_RUNNABLE
          | dispatch -> RUNNING
          | stop -> TERMINATED),
NON_RUNNABLE = ( resume ->RUNNABLE
               | stop ->  TERMINATED),
TERMINATED = STOP.
```

LTS of Java Thread Lifecycle



Key:

0: CREATED

1: TERMINATED

2: RUNNING

3: NON_RUNNABLE

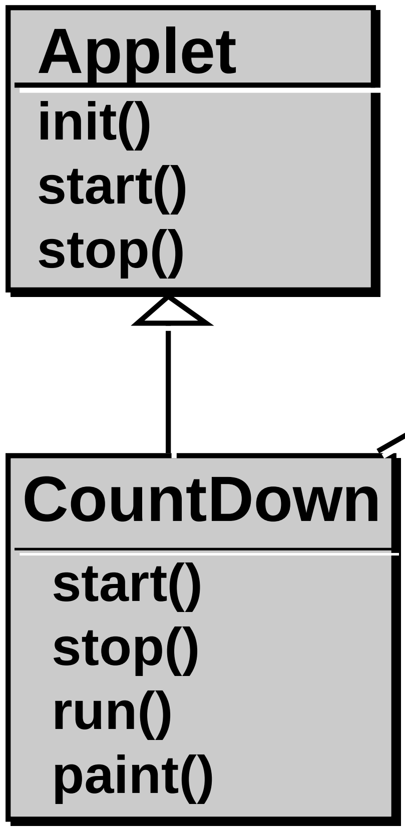
4: RUNNABLE

Example: CountDown Timer

- Demo: CountDown
- *FSP of CountDown:*

```
COUNTDOWN (N=3) = COUNTDOWN[N],
COUNTDOWN[i:0..N] =
 ( when(i>0) tick->COUNTDOWN[i-1]
 | when(i==0) beep->STOP
 ).
```

CountDown Timer - Class diagram



`Runnable` *is an interface*

CountDown Timer - Java class

```
import java.awt.*;                //windows toolkit
import java.applet.*;             //applet support
public class CountDown extends Applet implements Runnable{
  int counter; Thread cd;
  public void start() {     // create thread
   counter = 60;  cd = new Thread(this); cd.start();
  }
  public void stop() { cd = null;}
  public void run() {      // executed by Thread
   while (counter>0 && cd!=null) {
    try{Thread.sleep(1000);}
    catch (InterruptedException e){}
    --counter;  repaint(); //update screen
   }
  }
  public void paint(Graphics g) {
    if (counter>0)
      g.drawString(String.valueOf(counter),25,75);
    else g.drawstring(“Bang”, 10, 50);
  }
}
```

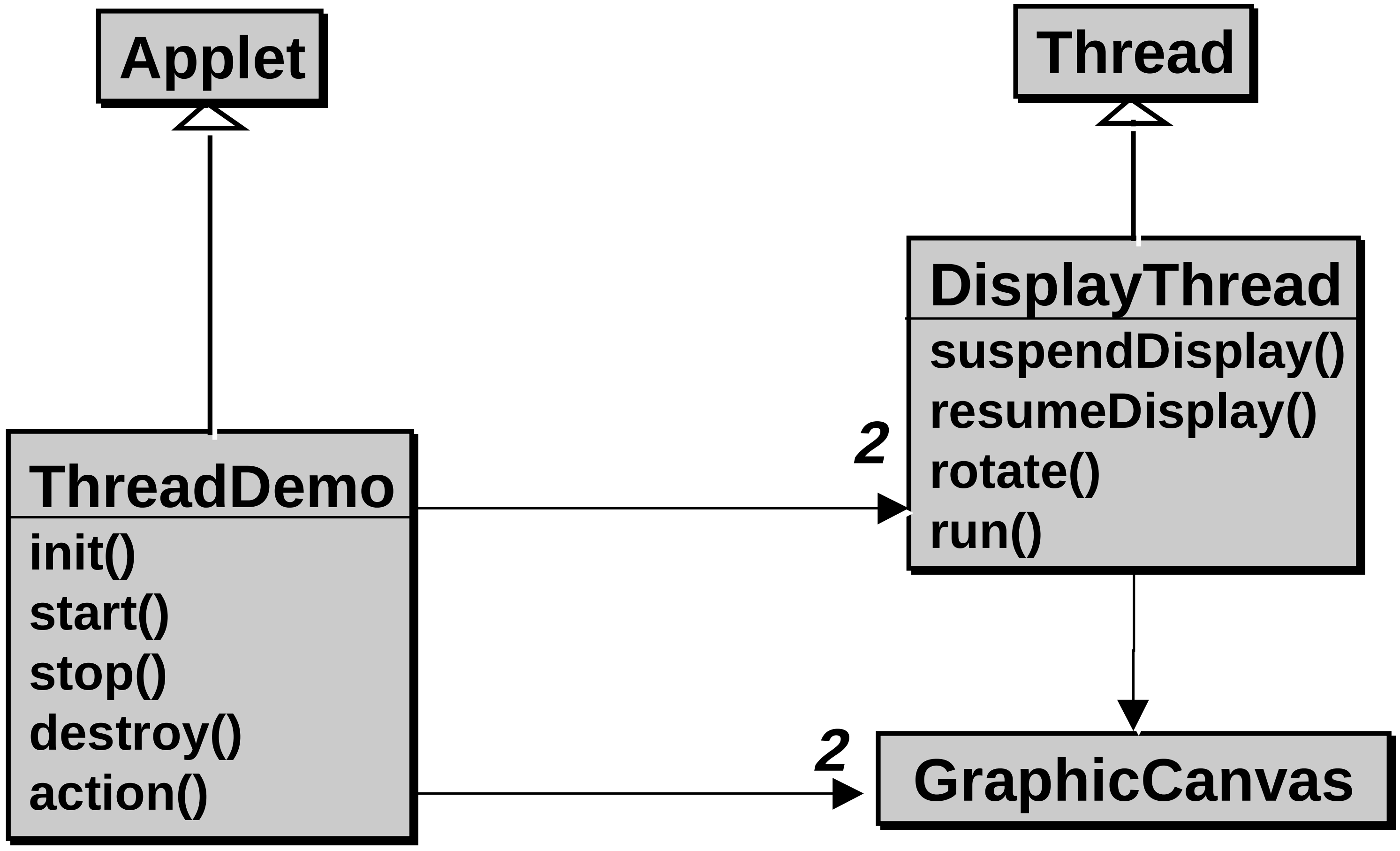
Concurrent Threads

- ***Parallel composition operator ||***
- ***Implemented by creation of several new thread objects***
- ***Example: ThreadDemo***
- ***Creates two thread objects that execute concurrently***

FSP Spec of Thread Demo

```
DISPLAY_THREAD = SUSPENDED,
SUSPENDED = ( resume->RUNNING ),
RUNNING =   ( rotate->RUNNING
            | suspend->SUSPENDED
            ).
||THREAD_DEMO =
   (a:DISPLAY_THREAD||b:DISPLAY_THREAD).
```

Class Diagram of ThreadDemo



Summary

- ***Threads vs. operating system processes***
- ***Threads through class inheritance / interface implementation***
- ***Thread lifecycle***
- ***Concurrent threads by creating new thread objects***
- ***Class diagrams***
- ***Next: Java Thread Programming Lab***