

CSSE 220

Multithreading

Checkout *Multithreading* project from SVN

Joe Armstrong,
Programming in Erlang

THE WORLD IS CONCURRENT

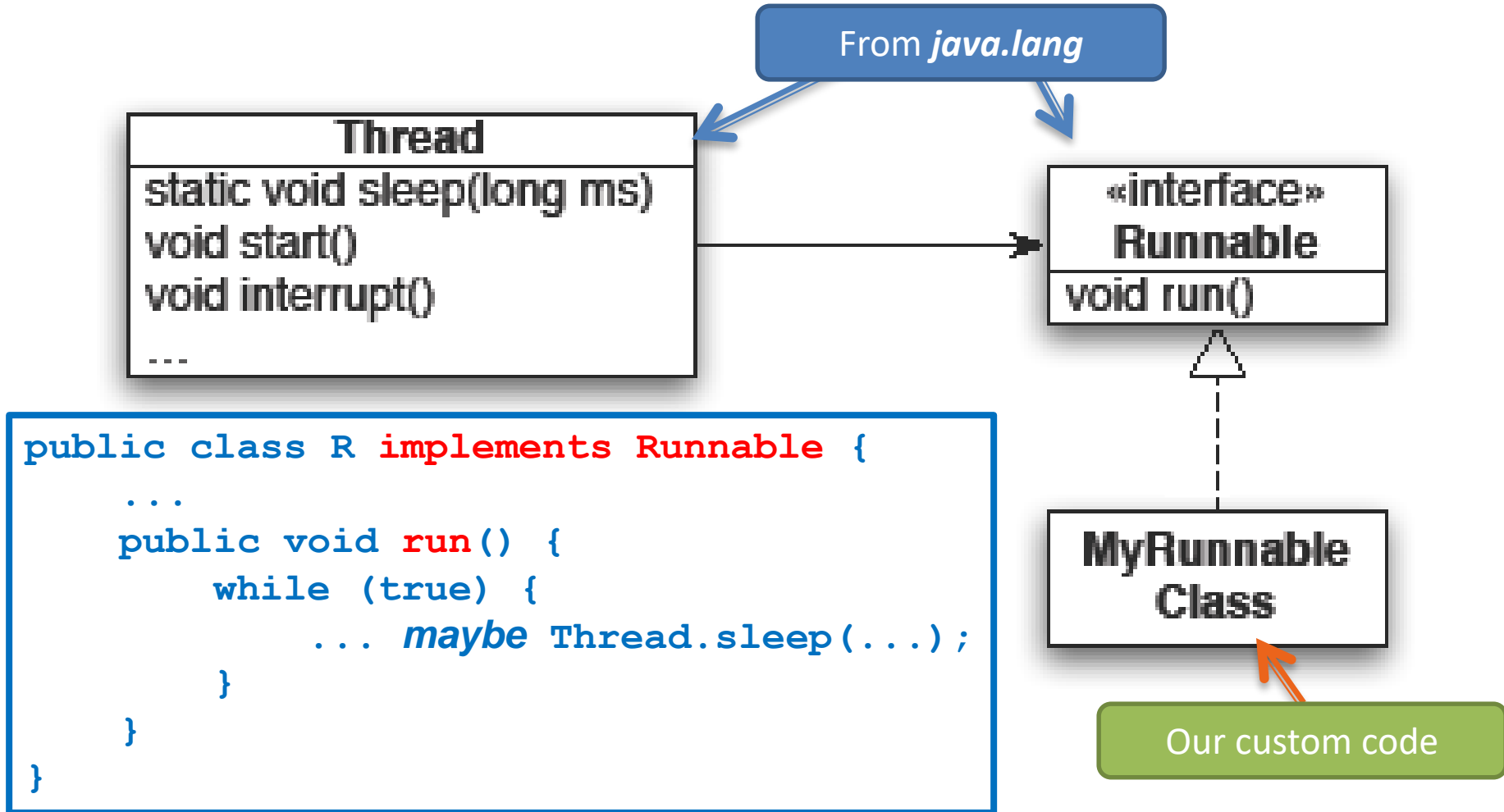
Multithreading

- A technique to:
 - Run multiple pieces of code “simultaneously” on a single machine

Time → Slices	1	2	3	4	5	6	7	8	9	10	11	12	13	14
running thread 1	█	█	□	█	□	□	□	█	□	█	□	□	█	█
running thread 2	□	□	█	□	█	█	█	□	█	□	█	█	□	□

- Run different parts of a program on different processor cores

Running Our Own Code Concurrently



Wherever you want to start the Thread:
`new Thread(object of type R).start();`

Animation with Threads

- Example 1: A single object
 - “Animate” it with button clicks
 - Animate it with a Timer

```
Timer timer = new Timer(50, animatorButton);  
timer.start();
```

- Animate it by using a thread

```
public class R implements Runnable {  
    ...  
    public void run() {  
        while (true) {  
            ... maybe Thread.sleep(...);  
        }  
    }  
}
```

Wherever you want to start the Thread:

```
new Thread(object of type R).start();
```

Animation with Threads

- Example 2: Multiple objects
 - Use separate thread for each object's "brain"
 - Another thread asks Java to update the GUI



Other Uses for Threads

- Web servers: many users connecting
- Desktop applications:
 - layout, spellchecking, auto-save, ...
- Scientific computing
- Weather forecasting
- ...

Caution!

- What if one thread is in the middle of performing an action when its time slice ends?
- What if a second thread's action interferes with the first's action?
- See bank example in today's project

Optional: For a way to fix this, see Big Java Section 20.4

Work time

Be sure everyone is getting a chance to drive.

TEAM PROJECT