Section 3

Threading and Locking
Definitions

What is a thread?
What is a thread?

A single flow of control with a process
Definitions

What is a thread?

A single flow of control with a process

Why use threads?
Definitions

What is a thread?

A single flow of control with a process

Why use threads?

Exploit latency, concurrency

Event-driven software
Dangers

What could go wrong?
Dangers

What could go wrong?

Race Conditions
Dangers

What could go wrong?

Race Conditions

How do we fix this?
Dangers

What could go wrong?

Race Conditions

How do we fix this?

Locking
What is a lock?
What is a lock?

Serializes access to some critical region of code or data

Used to enforce mutual exclusion concurrency control
What is a lock?

- Serializes access to some critical region of code or data
- Used to enforce mutual exclusion concurrency control

Locks need help from hardware
What is a lock?

- Serializes access to some critical region of code or data
- Used to enforce mutual exclusion concurrency control
- Locks need help from hardware

Different kinds of locks
Dangers

What could go wrong?
Dangers

What could go wrong?

Impacts Performance
Dangers

What could go wrong?

Impacts Performance

Hard to debug
Dangers

What could go wrong?

Impacts Performance

Hard to debug

Deadlocks
Dangers

What could go wrong?

Impacts Performance

Hard to debug

Deadlocks

How do we fix this?
Dangers

What could go wrong?

Impacts Performance

Hard to debug

Deadlocks

How do we fix this?

Punt (if collisions aren’t our problem)
Dangers

What could go wrong?

- Impacts Performance
- Hard to debug
- Deadlocks

How do we fix this?

- Punt (if collisions aren’t our problem)
- Resource hierarchy/Conductor/Chandy-Misra
Using Threads

Creation

Termination

Detachment and Joining

Self and Equal
Using Threads

Thread (class)

Runnable (interface)

ForkJoin

ThreadPools/Executor Services
Project 0 Review
Project 1 Suggestions

John’s Suggestion

One thread for listening (waiting for server probes)

One thread for keyboard input that handles server communication initiated by keyboard input

One thread for periodic re-registration
Project 1 Reminders

John’s Suggestion

One thread for listening (waiting for server probes)
One thread for keyboard input that handles server communication initiated by keyboard input
One thread for periodic re-registration

Remember

Terminate cleanly
You need to re-register
Modularity
Questions?

Java:

http://docs.oracle.com/javase/tutorial/essential/concurrency/index.html

http://docs.oracle.com/cd/E13150_01/jrockit_jvm/jrockit/geninfo/diagnos/thread_basics.html

C/Unix:


http://www.yolinux.com/TUTORIALS/LinuxTutorialPosixThreads.html