Lecture 22
Assignment 8

• Recitation this week is optional.
Threads

• As a generic term
  – Abstraction for program execution
    • Current point of execution.
    • Call stack.
    • Contents of memory.
  – Smallest unit of processing that can be scheduled by an operating system.

• Multithreading
  – Program running two or more threads concurrently.
    • Separate points of execution.
    • Separate call stacks.
    • Shared memory.
Rise of Threads

• In the beginning was the command line...
  – Neal Stephenson

• First computers were batch uniprocessors
  – Ran one program at a time.
  – Non-interactive
  – No need for multithreading.
Time Division

• Computers getting faster/cheaper gave rise to interactive computing with GUIs.
  – More than one program at a time.
  – Processor time division
    • Operating system rapidly switched between separate processes.
      – Resource sharing.
      – Not quite threads
        » Separate processes (i.e., programs)
        » No memory sharing.
      – Illusion of private resource.
Threads For Time Division

• Threads extend OS mechanisms for process-level time division to within a program.
  – Program needs/wants to make progress on two or more tasks.
    • Just as two separate processes want to share the processor to make “simultaneous” progress.
    • But as part of same program need to share memory and coordinate actions.

• Example: GUI
  – Want GUI to remain responsive when CPU heavy task is occurs.
  – lec22.v1
Parallel Programming

• Multicore increases performance only if we can find ways to parallelize our task.
  – Sometimes very easy.
    • No data or logic dependencies.
    • Aggregating data operations.
  – Sometimes very hard.
Java Threads

• Two basic mechanisms:
  – Extend Thread
    • Override run()
    • Call start()
      – Creates a new thread.
      – Executes run() in that thread.
    • lec22.v2
      – Note: only works once.
  – Implement Runnable
    • Provide reference to runnable object to Thread constructor.
    • Call start() on thread object.