

Threads II: Synchronization

Tom Kelliher, CS 245

Oct. 13, 2004

1 Administrivia

Announcements

Assignment

Read Chapter 6.

From Last Time

Introduction to threads.

Outline

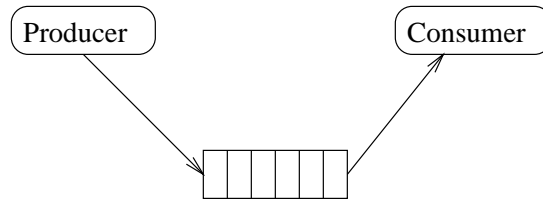
1. Thread Synchronization
2. Lab

Coming Up

Game2D case study.

2 Thread Synchronization

1. The Producer/Consumer problem:



- (a) Producer and Consumer work at different rates.
- (b) Fixed amount of buffering (queuing) between them.
- (c) With no synchronization, items can “disappear” or be multiply consumed.

2. A solution: the monitor.

- (a) Lock and condition variables.
- (b) A thread must secure the lock before executing within the monitor.
- (c) Once in the monitor, a thread checks its condition variables:
 - i. Producer: queue full.
 - ii. Consumer: queue empty.

If the condition is false, the thread waits — exits the monitor and sleeps. When it awakes, it **must** recheck its condition.

- (d) Once a thread has modified state within the monitor, it will notify waiting threads, allowing them to recheck their conditions. (Must first re-obtain lock.)

3. Example:

```
class Queue
{
```

```

private int val;
private boolean full = false;
private boolean empty = true;

public Queue()
{
    val = 0;
}

public synchronized void put(int v)
{
    while (full)
        try
        {
            wait();
        }
        catch (Exception e)
        {
        }

    val = v;
    full = true;
    empty = false;
    notify();
}

public synchronized int get()
{
    while (empty)
        try
        {
            wait();
        }
        catch (Exception e)
        {
        }

    full = false;
    empty = true;
    notify();
    return val;
}
}

```

3 Lab