# CS1007: Object Oriented Design and Programming in Java

Lecture #24

April 18

Shlomo Hershkop

*shlomo@cs.columbia.edu*

## Announcements

- No class on Thursday….chance to catch up with work/reading
- Start to think about next homework (will talk about it today)
- Next week is last week of classes
  - Next Tuesday will cover advanced topics
  - Next Thursday will do review etc

## Outline

- More on threads
- Locking
- Synchronization
- Thread safe code
- Animation examples
  - Sorting animation
  - Ping pong homework
- Reading: Chapter 9

## Next homework

- theory homework on things we’ve been discussing regarding objects/threads
- Programming ping pong game!
  - Instead of maze game, I want you to practice graphics/threading

## Random stuff

- Anyone used the JScrollPane ??
- What is a viewport ?

## Random II

- Any ideas on how to catch key presses in a GUI program ??

## KeyListener

- Register a key listener
- KeyTyped
  - Generally the value of what was pressed
- KeyPressed
  - How the value was entered

## A shortcut

- JButton jub = new JButtton(“Undo”);
- jub.setMnemonic(KeyEvent.VK_U);
- This will fire the button when the letter u is pressed!

# Reminder

- Lets look at last classes code
  - How is the thread started ?
  - Where is the logic?
  - How does it end ?

# Extensions

- Lets move the action to a start button

- What has to be done ?

# Code

- Basedon last classes, code

```
JButton start = new JButton("Start");
start.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent evt) {
                  new Thread(){
                        public void run() {
…
```

# Problems

- So we can launch the progress bar with a button

- What happens if I press it a few times ?

# Thread synchronization

- When threads can share a common object
  - Might conflict

- Example have a program to get order information off the web
  - Shopping cart
- Have another program to process orders
  - Delivery
- Orders are places in a list which can grow

# Shopping cart

-

```
//while list is full, sleep

if ( ! List.isFull()){
     List.add(new Order (..) );
}
```

# Delivery

- //while list is empty …sleep

```
while(!List.isEmpty()) {

Order proc = List.getFirst();
…
}
```

- What can go wrong ?

# Race problem

- While one thread is trying to add an Order
- The other might be removing it

- Many times, the specific test computer might be the perfect speed not to have a problem

# So how do you manage ?

- Same problem with launching the progress bar with a button

- Any suggestions for managing the progress bar updates

## Easy locking

- Synchronized keyword on method
- Can test for some condition and call:
  - wait();
- Once done with our work (i.e. we didn’t spin wait, call:
  - notifyAll();

## Multiple locks

- Remember if you want to grab multiple locks,

- Dining philosopher problem

# Something to think about

- Anyone know what a computer cluster is ?
- New CPU's will have multiple cores
- Which means what for your threaded code?

- Imagine a shared variable
- If each CPU has its own memory locations (for speed)
- One thread (using locks) might correctly update the variable, but old copy might exist on other CPU

# volatile

- Keyword
- Tells system that this variable might change, so not to store any copies elsewhere

- public volatile int groupcount;

---

- Anyone play computer games ?

- How do threads relate to computer games ?

# Code examples

- Let me launch the
- Sort animation1
- Lets look at the code
- Sort animation 2
- Lets look at the code

# Next homework

- Take some basic code and create a
- PING PONG GAME

# Sample code

- I’ve dug this up on the web

- Will post on website

- Can you make the game work correctly?
- Move it to swing ?
- Keep the score
- Ability to add a second ball ?
  - What does it involve?
  - When does the game end ?
- Magic block ?
- Color ?
- Sound ?

# Homework

- Any general questions on the homework ?

- Thanks, and please catch up over the weekend…
  - Read chapter 9

- Homework 4 will be posted by Friday noon-ish (or earlier)