CS 321 Lecture Notes

Orion Sky Lawlor
Department of Computer Science, University of Alaska at Fairbanks
http://lawlor.cs.uaf.edu/ olawlor olawlor@acm.org

1 2005/01/24: Flow of Control

We looked at several different ways of answering the question: What happens next?

1.1 Flows of Control

A `process` is a self-contained running program. It consists of code (executable machine binary code), memory (at least a heap and stack), and the processor registers. Processes can only be created by the OS, because only the OS can change the memory layout.

A `thread` is just a special kind of process that shares all its memory with another thread. Two threads in the same process share the same code and heap, but have separate stacks and processor registers. Threads can be created and maintained by the OS (called "kernel threads"), or by regular user code (called "user (or userland) threads").

1.2 Straight-Line Execution

The CPU normally executes (machine) code one statement after another, and only changes what it's doing when it executes branch or call instructions. This sort of regular sequential processing is how the vast majority of code runs.

Processes running sequential code are the oldest kind of flow of control. The code just runs line-by-line, and if a particular input is not available, the code stops, or "blocks" until the input becomes available. This kind of code is nice because it's obvious exactly what order things will happen in.

1.3 Switch-Style Code

However, not every program can be easily expressed as a sequential series of actions. For example, in a GUI program, what happens next is up to the user.

A common idiom for this kind of code is a big "loop and switch" statement. The program receives some sort of code identifying exactly what happened, and the switch statement jumps to the code to handle it appropriately. For example, in X windows, XNextEvent returns an XEvent structure containing a field named "type". If "type" is "ButtonPress", the user pressed a mouse button, and the XEvent structure contains the x,y coordinates. If "type" is "KeyPress", the XEvent structure contains which key was pressed. A program that cares about both mouse clicks and key presses would have a "case" statement for both ButtonPress and KeyPress.

1.4 Don't Call Me, I'll Call You

Windows programs normally also use a big switch statement, but instead of having the program ask the operating system for the next event, when an event happens Windows calls a routine inside the program. The routine is called a "Window Procedure",

or “WndProc”, and it gets passed a type code for what happened and two integers containing event-specific data.

The Java GUI also uses a similar “I’ll call you” to control, but it calls a specific method of the appropriate class. For example, when the mouse button is pressed in a window, Java calls the “mousePressed” method of the appropriate MouseListener.

Next lecture, we’ll look at another way control can be transferred in and out of a program—interrupts.