Operating System Simulator (CMSC 312)

Simulator built in Java using JavaFX for GUI. Also used some bare-bones HTML and Javascript to randomly generate program files.

How to run the Operating Simulator

Simply double click the `cmsc312-operating-system-simulator.jar` file to view the operating system.

The Operating System's clock ticks once every 90 millis seconds, so that as a user, you can see what is going on in the simulator.

Commands

- proc - Prints a list of the currently running processes and runtime information
- load <program name> - Load the specified program into the simulation.
- exe <number of cycles> - Run the simulator for the given number of cycles.
- exe - Run the simulator continuously. You can use "exe 0" to pause the simulation.

- reset - Clear all programs and reset the clock to 0.
- clear - Clear the text area.
- mem - Print the amount of used memory to the screen.
- exit - Closes the simulation.

Programs (Memory usage)

- ide (200 KB)
- paint (212 KB)
- textedit (66 KB)
- web_browser (255 KB)

Basic flow of the operating system

Each run through the operating system's `loop`:

- removes any processes in the ready queue to see if they are marked `EXIT`
- checks for any processes from the waiting queue that can fit in the space available into the ready queue, and brings the process into the ready queue
- detect for interrupts or preemptions, switching processes as necessary
- we can then safely advance the clock (doing so elsewhere would prevent us from checking for interrupts in the step above)
- if there is still a `current process`, we run the next command,

switching the process’s status as neccessary.
- after reaching a specific quantum of time, we signal an interrupt to mark that we should move on to the next process.

OS Components

Scheduler

The scheduler takes in processes and stores them in the ready queue until it is full. If a process does not fit into the ready queue it is stored in the waiting queue.

The scheduler uses a round robin approach. It takes each process and executes it for a certain time quantum (10 cycles).

When there is room in the ready queue the scheduler uses a first fit algorithm to find the first process that can fit into the available space.

IO and Interrupts

The IO command causes an interrupt, which pauses the process and puts it back at the end of the ready queue.

The process get a random amount of time to wait for IO, during which it is blocked and will be skipped in the round robin.

After the time to wait for IO runs out the process interrupts the current running process to resume running.

Git Repository

https://github.com/StevenmHernandez/cmsc312-operating-system-simulator