LinuxAcademy.com

Ruby 101

Intermediate Ruby: Sorting Part 1

Sorting:

Time	-> Time objects
IP Address	-> String
File Size	-> String

Problem

String comparisons are not the same as numberic.

```
"1000" > "999"   => false

"1000".to_i > "999".to_i => true

"192.168.0.1".to_i => 192
```

We could try converting them all to integers:

Would take a little more code.
Would take more work later if we wanted to filter subnets.

IPAddr class:

Provided by 'ipaddr' library

require 'ipaddr'

ip_address = IPAddr.new "192.168.0.1"

IP addresses can be compared, they can all convert .to_i
Provides methods for checking subnets.

model.rb

```
require 'ipaddr'
...
    def set_properties match_data
        @ip_address = IPAddr.new match_data[1]
...
```

view.rb

```
...
    def update log_file
...
        "\e[16G" + entry.ip_address.to_s +
```

Sort/Filter Object: Data Structure

- Sort:
    - Time
    - IP Address
    - File Size
- Directions:
    - Ascending
    - Descending

- Filters:
    - IP Address
    - Request
    - Time

model.rb

```ruby
class SortFilter
    attr_accessor :field_list, :field_name_index,
        :field_selection
    def initialize
        @field_list = [
            [:sort_by, [:none, :time_stamp, :ip_address, :file_size]],
            [:sort_direction, [:asc, :desc]],
            [:time_stamp],
            [:ip_address],
            [:request]]
        @field_name_index = 0
        @field_selection = [0,0]
   end
end
```

model.rb

```
…
class LogFile
    attr_accessor :file_name, :file_path,
        :log_entries, :directory, :directory_index,
        :log_entry_index, :list_start, :sort_filter
    ...
    def initialize
               cd "./"
               @log_entries = Array.new
               @sort_filter = SortFilter.new
         end
...
```

View: Displaying the Sort/Filter Screen with class SortFilterView

```
def quittable?
      false
end
def display sort_filter
      clear_display
      set_cursor
      print red(center("Sort and Filter"))
      update sort_filter
end
```

View: Displaying the Sort/Filter Screen with class SortFilterView

```
def update sort_filter
	set_cursor 2,1
	#----Loop through outer field_list array [a] => the fields----#
	sort_filter.field_list.each_with_index do |field_name, index|
		#---- if this is nil or String then it's an input field and not a choice box-----
		if field_name[1] != nil && field_name[1].class != String
			#----Display the choice box----#
			label = field_name[0].to_s.gsub(/_/, " ").upcase + ":"
			label = red(label) if index == sort_filter.field_name_index
			puts label
			field_name[1].each_with_index do | option, opt_index |
				option = red(option) if opt_index ==
								sort_filter.field_selection[index]
				puts "\e[K" + option.to_s
			end
			print "\e[K\n\e[K\n"
```

View: Displaying the Sort/Filter Screen with class SortFilterView

```
def update sort_filter
...
        else
                #-----Display the input field----#
                #These are typed input fields
                input = ""
                input = field_name[1] if field_name[1] != nil
                row = "Show only records where #{field_name[0].to_s.gsub(/_/, " ").upcase}
contains: #{input}"
                row = red(row) if index == sort_filter.field_name_index
                puts "\e[K" + row
            end
        end
        print "\e[J"
        set_cursor $stdin.winsize[0], 1
        print red("Esc to return, Move up/down to select, Tab to change focus, Return to Apply")
end
```

View: Minor Changes to instruction line

Escape key for exiting

```
print red("Esc to exit; up/down to move; return to select")

print red("Esc to exit, up/down to move, 's' to sort or filter")
```

Controller Changes:

- parse tab characters
- escape for exit, not 'q'
- new actions:
    - open up the sort/filter display
    - move cursor from one field to another
    - move cursor through field choices
    - type into input fields
    - leave the sort filter

Controller Changes:

```
def run
    ...
    if @current_view.quittable? && user_input == "\e"
    ...
end
def parse_input user_input
    ...
    when "\e[A"
        #up button ... update the view with a move action
        case @current_view.class.to_s
            when "FileDialogView"
                file_dialog_move -1
            when "LogListView"
                log_list_move -1
            when "SortFilterView"
                move_filter_selection -1
        end
```

```
...
when "\e[B"
    #down
    case @current_view.class.to_s
        when "FileDialogView"
            file_dialog_move 1
        when "LogListView"
            log_list_move 1
        when "SortFilterView"
            move_filter_selection 1
     end
when "\t"
    case @current_view.class.to_s
        when "SortFilterView"
            move_filter_field 1
    end
when "\e[D", "\e[C"
...
```

Controller Actions:

```
def sort_select
    @current_view = SortFilterView.new
    @current_view.display @log_file.sort_filter
end

def escape_sort_filter
    @current_view = LogListView.new
    @current_view.display @log_file
end
```

Controller Actions:

```
def move_filter_field increment
    @log_file.sort_filter.field_name_index += increment
    if @log_file.sort_filter.field_name_index >=
                          @log_file.sort_filter.field_list.length
        @log_file.sort_filter.field_name_index = 0
    end
    @current_view.update @log_file.sort_filter
end
```

```
def move_filter_selection increment
    current_field = @log_file.sort_filter.field_name_index
    field_list = @log_file.sort_filter.field_list

    if field_list[current_field][1] != nil && field_list[current_field]
                                                  [1].class != String
        @log_file.sort_filter.field_selection[current_field] += increment
        if @log_file.sort_filter.field_selection[current_field] >=
                                       field_list[current_field][1].length
            @log_file.sort_filter.field_selection[current_field] =
                                       field_list[current_field][1].length - 1
        end
        @log_file.sort_filter.field_selection[current_field] = 0 if
                    @log_file.sort_filter.field_selection[current_field] < 0
        @current_view.update @log_file.sort_filter
    end
end
```

Ruby 101

Intermediate Ruby:

```
def input_filter_field user_input
    current_field = @log_file.sort_filter.field_name_index
    if @log_file.sort_filter.field_list[current_field][1] == nil
        @log_file.sort_filter.field_list[current_field][1] = user_input
    elsif @log_file.sort_filter.field_list[current_field][1].class == String
        if user_input == "\u007F"
            @log_file.sort_filter.field_list[current_field][1].gsub! /.$/, ""
        else
            @log_file.sort_filter.field_list[current_field][1] += user_input
        end
    end
    @current_view.update @log_file.sort_filter
end
```