Ruby 101

Intermediate Ruby:Sorting Part 2

Intermediate Ruby:

How to call a method ... from a symbol

```
.send :method_name, arguments
```

```
class MyClass
      def my_number
            8
      end
end

my_instance = MyClass.new

my_instance.send :my_number => 8
```

Intermediate Ruby:

Our Sorting Algorithm:

Handle sorting by
- Time
- IP
- File Size

Intermediate Ruby:

model.rb

```
def apply_selections log_file
    #first sort the file
    if @field_selection[0] != 0
        if @field_selection[1]==0
            #sort by selected symbol asc
            log_file.log_entries.sort! do |entry_a, entry_b|
                entry_a.send(@field_list[0][1][@field_selection[0]]).to_i <=>
                    entry_b.send(@field_list[0][1][@field_selection[0]]).to_i
            end
        else
            #sort by selected symbol desc
            log_file.log_entries.sort! do |entry_a, entry_b|
                entry_b.send(@field_list[0][1][@field_selection[0]]).to_i <=>
                    entry_a.send(@field_list[0][1][@field_selection[0]]).to_i
             end
        end
    end
...
```

Intermediate Ruby:

Filter Algorithm:

separate out parts of date, then display only matching lines
match ip addresses and ip ranges
match requests that contain the search string

Intermediate Ruby:

Filter Algorithm: Date and Time

```
#apply the time stamp filter
if @field_list[2][1] != "" && @field_list[2][1] != nil
    #apply a time stamp filter
    regex = /(..)[\/-](..)\s(..):(..):(..)/
    matches = @field_list[2][1].match regex
    if matches != nil
        if matches[1] != "**"
            log_file.log_entries.select! do | entry |
                entry.time_stamp.month == matches[1].to_i
            end
        end
        if matches[2] != "**"
                log_file.log_entries.select! do | entry |
                    entry.time_stamp.day == matches[2].to_i
              end
        end
...
```

Intermediate Ruby:

Filter Algorithm: Date and Time

```
...
        if matches[3] != "**"
            log_file.log_entries.select! do | entry |
                entry.time_stamp.hour == matches[3].to_i
            end
        end
        if matches[4] != "**"
            log_file.log_entries.select! do | entry |
                entry.time_stamp.min == matches[4].to_i
            end
        end
        if matches[5] != "**"
            log_file.log_entries.select! do | entry |
                entry.time_stamp.sec == matches[5].to_i
            end
        end
    end
end
```

Intermediate Ruby:

Filter Algorithm: IP Address

```
...
# apply the ip address filter
if @field_list[3][1] != "" && @field_list[3][1] != nil
    #apply an ip filter
    ip_address_range = IPAddr.new(field_list[3][1])
    log_file.log_entries.select! do | entry |
        ip_address_range.include? entry.ip_address
    end
end
```

Intermediate Ruby:

Filter Algorithm: Request

```
...
    # apply the request filter
    if @field_list[4][1] != "" && @field_list[4][1] != nil
        log_file.log_entries.select! do | entry |
            entry.request.include? field_list[4][1]
        end
    end
end
```

Intermediate Ruby:

Controller Changes:

Handle a carriage return in the SortFilterView

Action to apply the filter

```
def parse_input user_input
    case user_input
        when "\n" , "\r"
            #carriage return / new line received
            case @current_view.class.to_s
                    when "FileDialogView"
                    file_dialog_select
                when "SortFilterView"
                    apply_sort_filter
            end
    ...
```

Intermediate Ruby:

Controller Action

```
def apply_sort_filter
    @log_file.log_entries = []
    @log_file.select_directory_or_load_file
    @log_file.sort_filter.apply_selections @log_file
    @current_view = LogListView.new
    @current_view.display @log_file
end
```

Intermediate Ruby:

Try it yourself:

Continue testing the program. Make a note of any problems you find. In the next lesson we will define and handle exceptions that should correct many of these kinds of problems. Think about how you might correct some of these issues.