Ruby 101

Intermediate Ruby: Reading a Log File

Intermediate Ruby:

Selecting a directory entry:Model Needs

1) a method to apply the current selection, whether a file or a directory
2) a method to load a file into memory

Intermediate Ruby:

```
class LogFile
...
    def select_directory_or_load_file
        if cd(@file_path +
            @directory.entries[@directory_index] + "/")
            :directory
        else
            if load_file
                :file
            end

        end

    end
...
```

Ruby 101

Intermediate Ruby:

```
class LogFile
...
    def load_file
        if File.file?(@file_path +
                                @directory.entries[@directory_index])
            @file_name = @directory.entries[@directory_index]
            @log_entries = IO.readlines(@file_path + @file_name)
            @log_entry_index = 0
            @list_start = 0
            true
        else
            false
        end
    end
...
```

Intermediate Ruby:

View Needs:

1) A method to display file contents, highlighting the current line.
2) An update method, in case the user moved the cursor.
3) Code to trim lines, if they are too long.

Ruby 101

Intermediate Ruby:

```
class LogListView
    ...
    quittable?
        true
    end

    def display log_file
        clear_display
        set_cursor
        print red(center(log_file.file_name)) + "\n"
        update log_file
    end
```

Ruby 101

Intermediate Ruby:

```
def update log_file
    set_cursor 2,1
    log_file.log_entries.each_with_index do |entry, index|
        if index < log_file.list_start
            next
        end
        if index > log_file.list_start + $stdin.winsize[0] - 3
            break
        end
        entry = entry.chomp.gsub("\t", "     ").slice(0,
                                                $stdin.winsize[1])
        entry = red(entry) if index == log_file.log_entry_index
        print "\e[K" + entry + "\n"
    end
    print "\e[J"
    set_cursor $stdin.winsize[0], 1
    print red("Type 'q' to exit, up/down to move, 's' to sort or
filter");
end
```

Intermediate Ruby:

Controller Needs:

1) An action for directory entry selection
2) An action to handle moving up and down in a file
3) code to call these action

Intermediate Ruby:

```
def parse_input user_input
    case user_input
        when "\n" , "\r"
            #change controller likely to happen
            case @current_view.class.to_s
                when "FileDialogView"
                    file_dialog_select
            end
        when "\e[A"
        #up button ... update the view with an up action
           case @current_view.class.to_s
                when "FileDialogView"
                    file_dialog_move -1
                when "LogListView"
                    log_list_move -1
            end
...
```

Intermediate Ruby:

```
...
        when "\e[B"
            #down
                case @current_view.class.to_s
                    when "FileDialogView"
                        file_dialog_move 1
                    when "LogListView"
                        log_list_move 1
            end
...
```

Intermediate Ruby:

```
def file_dialog_select
    case @log_file.select_directory_or_load_file
        when :directory
            @current_view.update @log_file
        when :file
            @current_view = LogListView.new
            @current_view.display @log_file
    end
end
```

Intermediate Ruby:

```
def log_list_move increment
    @log_file.log_entry_index += increment
    if @log_file.log_entry_index < @log_file.list_start
        @log_file.list_start = @log_file.log_entry_index -
                                        $stdin.winsize[0] + 3
    elsif @log_file.log_entry_index > @log_file.list_start +
                                            $stdin.winsize[0] -
3
        @log_file.list_start = @log_file.log_entry_index
    end
    @current_view.update @log_file
end
```