Ruby 101

Advanced Ruby: Progress Bar

Advanced Ruby:

File Loads:

short_files ... short time

large files ... large time

Needs:

inform the user that something is happening

display a progress bar in the notice area

Advanced Ruby:

How to do this?

Two threads

1) File Loading Thread
2) Display Thread

Thread 1 -> Runs as fast as it can
Thread 2 -> update the progress bar

a) File Reading
b) File Parsing

Advanced Ruby:

Model Needs:

Percent of file loaded
Percent of file parsed

Keep track of File object
Keep track of progress through log_entries

Advanced Ruby:

model.rb

```
class LogFile
attr_accessor :file_name, :file_path, :log_entries,
              :directory, :directory_index,
              :log_entry_index, :list_start,
              :sort_filter, :parse_percent

    def initialize
        cd "./"
        @log_entries = Array.new
        @sort_filter = SortFilter.new
        @parse_percent = 0.0
        @actual_file = nil
    end
```

Advanced Ruby:

model.rb

```
class LogFile
...
    def file_initialized?
        @actual_file != nil
    end

    def file_percent_loaded
        @actual_file.pos.to_f / @actual_file.size.to_f
    end

    def clear_file
        @actual_file = nil
    end
```

Advanced Ruby:

```
def load_file
    begin
        if File.file?(@file_path + @directory.entries[@directory_index])
            @file_name = @directory.entries[@directory_index]
            @actual_file = File.new(@file_path + @file_name)
            @file_name = @directory.entries[@directory_index]
            log_array = @actual_file.readlines
            @parse_percent = 0.0
            log_array.each_with_index do |log, index|
                @log_entries[index] = LogEntry.new log
                @parse_percent = index.to_f / log_array.count.to_f
            end
            @log_entry_index = 0
            @list_start = 0
            true
        else
            ...
```

Advanced Ruby:

View needs:

Basic Progress Bar Display
(Label + ****)

Advanced Ruby:

view.rb

```
class BasicView
...
    def progress_bar percent, label = "Loading"
        set_cursor $stdin.winsize[0], 1
        label_line = label + " ["
        print label_line
        length = percent * ($stdin.winsize[1] -
                                        label_line.length + 1)
        bar = "*" * length
        print bar + "\e[K"
        set_cursor $stdin.winsize[0], $stdin.winsize[1]
        print "]"
    end
```

Advanced Ruby:

Controller needs:

- Create two threads during file loads:
  - load_thread
  - display_thread
- Communicate back to controller type of file (file or directory)

Advanced Ruby:

controller.rb

```
def file_dialog_select
    # create queue to store symbols, :file or :directory
    load_response = Queue.new
    # create a thread to load file data
    load_thread = Thread.new do
        begin
            # store response in queue
            load_response << @log_file.select_directory_or_load_file
        rescue NotAnApacheAccessLog
            @log_file.clear_file
            @current_view.notice "File does not conform to Access Log
pattern"
        rescue NoFileAccess, NoDirAccess
            @log_file.clear_file
            @current_view.notice "File or Directory Access Not Permitted"
        end
    end
    ...
```

Advanced Ruby:

```
def file_dialog_select
    ...
    display_thread = Thread.new do
        #if the load_thread is working ok
        while load_thread.status != nil && load_thread.status != false do
            # load_thread has initialized the @actual_file
            if @log_file.file_initialized?
                if @log_file.file_percent_loaded != 1
                    @current_view.progress_bar @log_file.file_percent_loaded,
                                                "File Loading"
                else
                    # display a new progress bar after file is loaded
                    # displaying percent of parsing
                    @current_view.progress_bar @log_file.parse_percent,
                                                "File Parsing"
                end
            end
            Thread.pass
        end
    end
    ...
```

Advanced Ruby:

```
def file_dialog_select
    ...
    # Join threads to ensure processing is complete
    display_thread.join
    load_thread.join
    #if successful there is an object in the queue
    if !load_response.empty?
        case load_response.pop
            when :directory
                @current_view.update @log_file
            when :file
                @current_view = LogListView.new
                @current_view.display @log_file
        end
    end
end
```

Advanced Ruby:

controller.rb

```
def apply_sort_filter
    # Create a thread to load the file data
    load_thread = Thread.new do
        @log_file.log_entries = []
        @log_file.clear_file
        @log_file.select_directory_or_load_file
    end
    ...
```

Advanced Ruby:

controller.rb

```
def apply_sort_filter
    ...
    # Create a thread to display progress
    display_thread = Thread.new do
        while load_thread.status != nil && load_thread.status != false do
            if @log_file.file_initialized?
                if @log_file.file_percent_loaded != 1
                    @current_view.progress_bar @log_file.file_percent_loaded,
                                                "File Loading"
                else
                    @current_view.progress_bar @log_file.parse_percent,
                                                "File Parsing"
                end
            end
            Thread.pass
        end
    end
```

Advanced Ruby:

controller.rb

```
def apply_sort_filter
     ...
     # join the threads to ensure processing is complete
     display_thread.join
     load_thread.join

     # apply the filters and sorting
     begin
          @log_file.sort_filter.apply_selections @log_file
          @current_view = LogListView.new
          @current_view.display @log_file
     rescue IPAddr::InvalidAddressError
          @current_view.notice "Please input a valid IP address"
     rescue InvalidDate
          @current_view.notice "Please input a date and time 'MM-DD HH:MM:SS'"
     end
end
```