# Lisp in Summer Projects Submission

| | |
|---|---|
| **Submission Date** | 2013-10-24 14:42:45 |
| **Full Name** | Nehemiah Paramore |
| | |
| | |
| | |
| | |
| | |
| **Country** | United States |
| **Project Name** | easy-timer |
| **Type of software** | other |
| **General category** | utility |
| **LISP dialect** | Emacs Lisp |
| **GitHub URL** | https://github.com/nehemiahap/easy-timer/blob/master/easy-timer.el |
| **Did you start this project?** | Yes, all the code is written by me |
| **Project Description** | I want to describe my project in this form. |
| **Purpose** | This project is an accidental re-implementation of tea-time.el. It was intended to allow the user to easily set a countdown timer, and pause/resume the timer if needed. The pause/resume functionality is not included in tea-time.el. |
| **Function** | The code allows the user to set a countdown timer, display the time in the emacs message buffer, and pause/play the timer easily. When the timer runs out, it switched the emacs focus to the message window, so the user is notified without significant disruption. |
| **Motivation** | I wrote this because I wasn't satisfied with the extend-ability of the timer utilities already written. I was inspired by org-mode's timer function, but then simplified it greatly. |
| **Audience** | I am the audience. This project was more of a motivation to see if I could build something that served my own needs more than the solutions already available. |
| **Methodology** | Variables<br>task-timer-default-time<br>- task-timer default time can be either a number or a time string as specified in the elisp documentation. This is the user defined default time to use for all timers. In the case |

| | |
|---|---|
| | that this variable is set to a number, it is formatted upon use to a time string with units of minutes.<br><br>task-timer<br>task-timer is used to represent several things all at once. Throughout the duration of execution, it takes the following forms:<br>- valid time string : in this case, the timer has been paused, when we unpause it, we simply pass the task-timer as the argument to the function that sets the task-timer. We then reassign task-timer to the timer that was formed.<br>- emacs timer(vector) : in this case, the timer is active. When we pause it from this state, the timer is cancelled and task-time is set to a valid time string equal to the amount of time left at the time of pause.<br>- nil : Before any timer has been set, or in the event that the timer runs out, the task-timer is set to nil. This serves as an indicator of whether we need to set, resume, or pause a timer. Because the control of the timers is relegated to one function for sake of ease, this polymorphism of the variable is useful. |
| **Conclusion** | This project had very humble goals. The project did accomplish it's functionality requirements though. One of the limitations that I'm considering expanding is the use of multiple timers. But, I'm unsure of the usefulness of this expansion. I'd also like to possibly implement buffer specific timers that pause when the buffer is idle or not selected. |
| **Build Instructions** | Put the file in your .emacs directory and load it.<br>(load "dir/to/task-timer.el")<br>Consider setting the default-task-time variable. This can be a number, or a time string. Numbers default to minute units. Valid time string : "30 min 30 sec"<br>(setq task-timer-default-time 30)<br><br>The only two functions that should be bound to keys are task-timer-display-time and task-timer-pause-resume |
| **Test Instructions** | Once the files have been loaded, simply call M-x task-timer-pause-resume or M-x task-timer-display-time. Other than that, no test suite was necessary. |
| **Execution Instructions** | Once the files have been loaded, simply call M-x task-timer-pause-resume or M-x task-timer-display-time. |
| **Describe any bugs or caveats** | This is a very simple project, so it might not be very extendable. |
| **Official** | I have read rules and have abided by them.<br>I am 18 years of age or older.<br>I am not living in Brazil, Quebec, Saudi Arabia, Cuba, Iran, Myanmar (Burma), North Korea, Sudan, or Syria. |