# Scheduler User Manual

## INPUTS

* **Planner (.m file)** 
  + This is a list of the task for each pc.
  + This is a shared file for all PC.
  + I not sure how this file is shared.
    - I think it may be through an SVN update.
    - Please check.
* **Track file (.mat file)**
  + The is found in the.\Track\ directory
  + Each day will be log as a .mat with the day date.
  + This is generate the first time from the planner file once.
  + Once it be generated it is updated as of when the programs start/stop.

## FUTURE WORK

* Try and make this completely generic.
  + Can this be a bit more like the windows scheduler.
  + delay the action trigger etc.
* Make the tracker viewable?
* Dos Task Manger
* Share Program Manager. (mindowSchedular)
* Take the dos classdef from work
  + Add it to the dos task manager stuff.

## EXAMPLE PLANNER FILE

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **PC\_Name** | **ProgramName** | **MacroName** | **StartTime** | **Type** |
| BRYAN\_PC | URL\_Download | BritishBulls\_ALLSTATUS | 20:00:00 | WeekDaysOnly |
| MT | URL\_Download | ADVFN\_URL\_CompanyInfo | 20:50:00 | WeekDaysOnly |
| MediaPc | URL\_Download | SharePrice\_Summary | 19:00:00 | WeekDaysOnly |
| BRYAN\_PC | WebPageDecoder | ADVFN\_ProcessDay | 00:00:01 | WeekDaysOnly |
| BRYAN\_PC | WebPageDecoder | BB\_HIST\_Decode | 00:00:01 | WeekDaysOnly |

## mSchedular

### Description:

A very basic tool uses to scheduler the timing of task.

* You should be able to add a task and scheduler for a time.
* You should also be able to add a task to a queue of events.
* You should be able to display the status of all queued a running events.

## mTaskManger

### Description:

A very basic control use to start/stop and monitor tasks.

You should be able to start a dos command and track it progress.

Useful functions:

* RunTask
  + INPUTS
    - Execute name.
    - Number of time to run if fail.
  + Run a dos program with definable inputs
  + It should return a process if. (PID)
* GetTaskStaus
  + INPUTS
    - (PID) – Process ID
  + Should return:
    - Status – NotPreset, InProgress, Complete.
* KillTask
  + INPUTS
* GetProcessInformation
  + This should be a list of running processes and status information.
  + This is different from the GetProcessInformation from the mDosTaskManager as it will only contain processes executed from the matlab processes.

## mDosTaskManager

Just a wramper around only the dos commands.

* Remote system
  + This can manage process from a remote system.
* Consider using the class you use for work.
* It could log all the dos commands to a file.
* It should be able to do a process kill.
* GetProcessInformation
  + This should be a list of running processes and status information.
  + This is different from the GetProcessInformation from the mDosTaskManager as it will only contain processes executed from the matlab processes.

## DOS\_Command\_Logger

Very basic class which replaced the dos or system command. But this allow you to log into a file the commands sent to the dos console.

### Current Implementation

* Log to command window only.

### Key Future Requirements

* It should be able to log to the same file even if it’s a different exe.
* Commands should be available to clear the log file and save the log file,
* You should be able to check how big the log current is?
* The date and time the log was created.
* Maybe use the properties2store class as the may work better across different exe. However the problem may be when you change the file the other exe may not know of the change and overwrite. So you may want to check the file value before ovwerwritting them in this class.
* It would be great if all commands from all programs appear in a DOS terminal (i.e a matlab GUI). In this GUI we could filer on specific programs etc.
* Can you put a listener on a file changing? (This could prompt and update on the properites 2 store function.

QUEUE.M CLASS??????? To manage to the time,

### Concept

.

Single GUI in matlab which has all commands and filterable.

Multiple DOS Terminal for each program

>> DOS

KillTask example.exe

## Installation Checks

Maybe have a piece of m code to do all these checks.

* Check that the Y drive is mapped and you can run the .exe

## Scheduler Class UML model

**TASKMASTER.m ?**

**SchedulleTask AddTaskToQueue GetTaskStatus GetRunningTasks GetCompletdTasks GetPlanned Taks**

This file could be predefined. (A sort of settings file)

The setting file could be changes remotely by a master program

TASK LIST ON FILE SYSTEM

BritishBulls Run at 12.00 pm Yahoo Run at 12:30 pm

LOGIC

**mTaskManager.m**

LOGIC

**RunTask GetTaskStatus KillTask GetRunningProcess GetProcessHistory**

**mSchedular.m**

**Sends DOS command to Shell**

**This is DOS WRAPPER around the fundamental dos commands. Not fancy is on this layer just a wrapper around available task manager style commands**

**mDosTaskManager.m**

LOGIC

LOGIC

**RunTask KillTask GetRunningProcess**

**DOS\_Command\_Logger.m**

**DOS Shell**

Example.exe

Kill Example.exe

**DOS LOG FILE**

**MATLAB COMMAND WINDOW**

Example.exe

Kill Example.exe

**Example.exe Kill Example.exe**

**This layer is to extend the *tracking* capability of the dos layer and also to allow only the matlab exe to be tracked. All other task running on the terminal will be *ignored*.**