CS2103 PROJECT MANUAL

Command line task manager

He Haocong

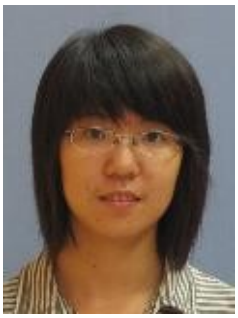
U099121H

Liu Jialong

U099122U

Wang Xiangyu

U099120W



Zhou Biyan

U094837M

November 1, 2010

Contents

1 Introduction

abc abc

2 User Guide

2.1 Quick Start

This section introduces you the minimum amount of commands to get you started.

1. Start taskManager shell

   On Mac OS and GNU Linux (referred to as "*nix" later), taskManager can be started from shell by commands:

   ```
   $ cd the/folder/containing/taskManager
   $ ./taskManager [1]
   ```

   On Microsoft Windows, taskManager can be launched in command prompt as well, or simply by double clicking.

   Once taskManager is launched, you will see a prompt like "> _", and you will start typing commands!

2. Add some tasks

   ```
   $ ./taskManager
   > add "Sample task 1"
   > add "Sample task 2"
   TaskManager:  This task is highly similiar to some existing
   task, do you really want to add it?  y
   ```

   To add a task, simply use the `add` command followed by the description of the task in a pair of quotation marks.

   If no error message is shown, the task is successfully added. TaskManager may prompt for confirmation if the task to be added is highly similar to some existing task(s) to help prevent people forget adding tasks, which is the case in the example above.

3. List the existing tasks

[1] *nix version can be run at any directory after installing taskManger – "make install". See details in section "Compilation and Installation".

```
> ls
1 Sample task 1
2 Sample task 2
```

To see the existing tasks, use `ls` command. By default, the taskManager shows the serial number and the description of the tasks.

4. Mark a task as finished

```
> finish 2
> ls
1 Sample task 1
2 f Sample task 2
```

To finish an existing task, use the finish command followed by the serial number of the task to finish. Notice that for finished task, a "f" is shown between serial number and task description.

5. Remove task(s)

```
> ls
1 f Sample task 1
2 Sample task 2
3 Sample task 3
> rm 1 2
TaskManager:  Do you really want to remove these tasks permanently?
y
> ls
3 Sample task 3
```

To remove a existing task, use `rm` command followed by the serial number(s) of the task to remove. TaskManager will prompt for confirmation when removing tasks.

6. Exit from taskManager

```
> exit
```

To quit from taskManager, use `exit` command. All changes to the existing tasks will be automatically saved.

2.2 More commands

2.2.1 read, import and export

TaskManager stores the tasks in an XML file which is by default located at `~/record.xml` on *nix, and `%USERPROFILE%\record.xml` on Windows.

TaskManager also supports importing and exporting. This is done by `read`, `import` and `export` commands.

1. `read`

   `read` command reads an xml file, list all the tasks it contains without affecting the current task list.

   This is helpful when you only want to peek the content of an xml file without really importing it.

   ```
   > ls
   1 Sample task 2
   > read thisweek.xml [2]
   1 f CS2103 midTerm Sep 29 06:30 - 07:30 pm MPSH 1B
   2 f CS3230 midTerm Oct 15 06:00 pm
   3 f CS3241 midTerm Oct 07 lecture
   4 f CS3244 midTerm Oct 04 lecture
   5 f ST2132 midTerm Oct 08 LT33 12:15 - 1:30 pm
   > ls 1 Sample task 2
   ```

2. `import`

   `import` command is similar to `read` command. It reads the content of the xml file and appends all the tasks in it to current task list.

   ```
   > ls
   1 Sample task 2
   > import mytasks.xml
   1 Sample task 2 [3]
   2 f CS2103 midTerm Sep 29 06:30 - 07:30 pm MPSH 1B
   3 f CS3230 midTerm Oct 15 06:00 pm
   4 f CS3241 midTerm Oct 07 lecture
   5 f CS3244 midTerm Oct 04 lecture
   6 f ST2132 midTerm Oct 08 LT33 12:15 - 1:30 pm
   ```

3. `export`

   `export` command exports current task list to an xml file or html file.

   ```
   > export sampletasks.xml [4]
   > exit
   ```

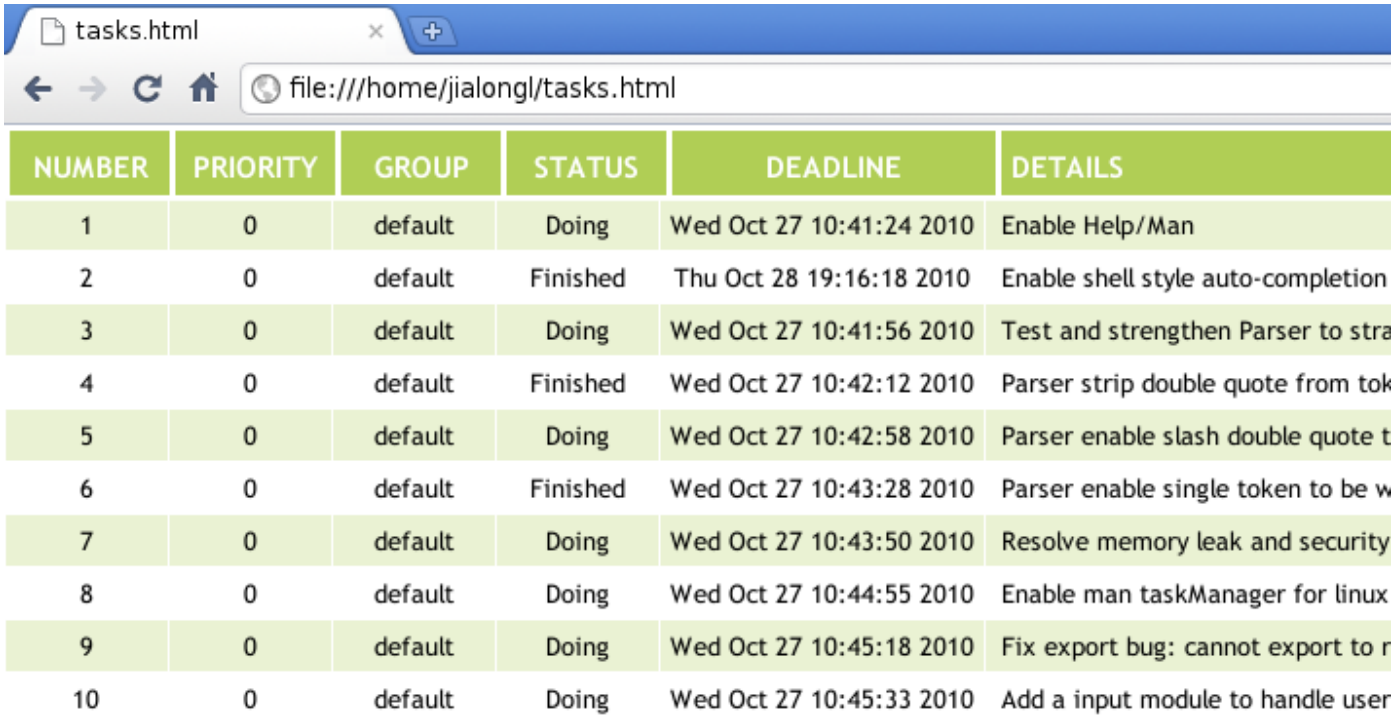
[2] The file name is not quoted. If the file name contains space, please quote it with a pair of quotation marks.

[3] Task 1 is still in task list. Importing tasks will not erase existing tasks.

```
$ cat sampletasks.xml
<taskList>
<task>
<serialNumber>1</serialNumber>
<deadline>1288473083</deadline>
<priority>0</priority>
<description>Sample task 2</description>
<group>default</group>
<isFinished>0</isFinished>
</task>
</taskList>
```

`export` can also be used to generate an html file which is more visually pleasant in your favourate browser.

```
> export -html sampletasks.html
> exit
```



NUMBER	PRIORITY	GROUP	STATUS	DEADLINE	DETAILS
1	0	default	Doing	Wed Oct 27 10:41:24 2010	Enable Help/Man
2	0	default	Finished	Thu Oct 28 19:16:18 2010	Enable shell style auto-completion
3	0	default	Doing	Wed Oct 27 10:41:56 2010	Test and strengthen Parser to stra
4	0	default	Finished	Wed Oct 27 10:42:12 2010	Parser strip double quote from tok
5	0	default	Doing	Wed Oct 27 10:42:58 2010	Parser enable slash double quote t
6	0	default	Finished	Wed Oct 27 10:43:28 2010	Parser enable single token to be w
7	0	default	Doing	Wed Oct 27 10:43:50 2010	Resolve memory leak and security
8	0	default	Doing	Wed Oct 27 10:44:55 2010	Enable man taskManager for linux
9	0	default	Doing	Wed Oct 27 10:45:18 2010	Fix export bug: cannot export to r
10	0	default	Doing	Wed Oct 27 10:45:33 2010	Add a input module to handle user

Figure 1: tasks exported as webpage [5]

[4] Currently `export` does not support environmental variables in path. E.g. export ~/abc.xml will not export the file to /home/username/.

[5] Page may not render correctly in IE 6 or its earlier versions.

2.2.2 task

```
   > task 1
Number:  1 Deadline:  Sun Oct 31 05:11:23 2010
Priority:  0 Status:  Doing
Group:  default
Details:
Sample task 2
```
[6]

To show detail information of a task, use `task` command followed by serial number of the task.

2.2.3 pri

```
   > pri 1 10
> task 1
Number:  1 Deadline:  Sun Oct 31 05:11:23 2010
Priority:  10 Status:  Doing
Group:  default
Details:
Sample task 2
```

To change the priority of a task, use `pri` command followed by the serial number of a task and its new priority. Priority is typically a number between -20 and 20. By default, the priority of a newly added task is 0.

2.2.4 edit

```
   > edit 1 -d "Sample task 3" -p 12 -t 1d
```
[7]
```
 -g SampleGroup -f yes
> task 1
Number:  1 Deadline:  Mon Nov 1 05:47:59 2010
Priority:  12 Status:  Finished
Group:  SampleGroup
Details:
Sample task 3
```

To edit a task, `edit` command is used like this: `edit <task serial number> -d <new description> -p <new priority> -t <new deadline> -g <new group> -f <finished or not>`

[6] Adding tasks with detailed information is covered in chapter 3. In this example, default values are shown.

Only serial number is compulsory. Besides, to finish a task, `finish 1` is equivalent to `edit 1 -f yes`.

2.2.5 undo

> `undo` [8]
Undo the last command.

2.2.6 redo

> `redo`
Redo the last undo. If more than one undo command executed, redo the next latest one.

2.3 Using Options

Like `edit`, some of the commands come with options to support more functionality. In this chapter they are introduced in great detail.

2.3.1 add

Use -t option to add a task with a deadline: > `add "some task" -t 3d2h`

taskManager support 3 types of time format: Plus format: <number:a>d <number:b:cMeans a days b hours c minutes later. i.e. current time plus a day b hours c minutes. None of them is compulsory e.g. 3d2h means 3 days 2 hours from the moment the command is executed. By format: b:a:b:c:dMeans before c hour d minutes on the b-th day of a-th week from now. None of them is compulsory e.g. b0w5d means by friday this week. b2w3d8h means by wednesday 8:00am, 2 weeks later. b1w means by start of next week, i.e. by the end of this week. b2d means by the end of tomorrow. b0d22h means by 10:00pm today. b10d means by start of the 10-th day from now, i.e. in 10 days including today.

Unix timestamp: number of seconds since 1/1/1900 00:00:00

Use -p option to add a task with a priority >`add "some important task" -p 20`

[7] -t 1d means setting the deadline to be 1 day later. Time format that taskManager accepts is discussed in section 3.1

[8] The undo command has no effect on commands like ls, export, tui and undo.

Use -g option to specify a group for a task > `add "the task with group"`[9] `-g SampleGroup`

Options are not compulsory. Different options can be used together. For example `add "some task" -p 10 -g "special task" -t 4d`.

2.3.2 ls

Use -s to sort the task list

> ls -s "deadline priority"

A more general format is:

> ls -s "keyword 1>keyword 2>..."

The listed tasks will be sorted by keyword 1 then keyword 2 ...

Available search keywords are: deadline priority serialnumber. Prefix of a keyword is also acceptable. e.g. -s "p" will sort the list by priority e.g.

> ls -s "p d"

1 Sample task 1. This also has high priorty 10 Sun Oct 31 06:49:09 2010 2 Sample task 2. This has high priority 10 Mon Nov 1 06:54:42 2010 3 Sample task 3. This is the latest 0 Tue Nov 2 06:54:37 2010

Use -k to filter tasks with a keyword

> ls -k *Sam?le*task

? means can be replaced with any character. * means can be replaced with any number of ?s. (including 0) keyword is case insensitive

E.g. "This is a sample with a important task" will match *Sam?le*task by letting first * to be "This is a ", ? to be 'p' and second "*" to be " with a important " "samqleTask" will also match *sam?le*task by letting both * to be empty string and ? to be 'q'

Use -f to show/hide finished tasks

> ls -f yes

Show only finished tasks

> ls -f N

Show only doing tasks

Use -g to show tasks of a specific group

> ls -g SampleTask

Show only tasks from SampleTask group

Notice: Different options can be used together. When more then one restrictive options are there, conjuction of these restrictions are used. e.g. ls -g SampleTask -f y will show finished tasks from SampleTask group.

[9] If group name contains spaces, use a pair of quotation marks to quote it.

2.3.3 rm

Use -g option to remove a group of tasks:

`rm -g SampleTask` removes the entire SampleTask group.

`rm` can be used to remove a list of tasks as well. e.g. `rm 1 2 3` removes tasks 1, 2 and 3.

Notice: Commands like finish, rm, export, etc. do not support all task-selective options like -g -k -f. Executing these commands on a selected task set can be done with command piping, which is discussed in section 2.5.1.

2.4 Text Based Interactive User Interface

blah blah blah

2.5 Advanced Usages

2.5.1 command piping

> ls — rm

TaskManager support command piping for most commands. Though it is a bit different from traditional linux/unix pipe. Piping in taskManager is done with symbol '—'. When a few taskManager commands are piped together, they are executed one by one from left to right. If one command has output, it is then passed to the next command as input. Only output of the last command is shown.

Examples:

a) finish all tasks

> ls — finish

b) remove all finished tasks

> ls -f yes — rm

c) import from a file and replace corrent task list

> ls — rm — import newTasks.xml

d) import all CS2103 group tasks from a file

> read newTasks.xml — ls -g CS2103 — add

e) export all CS2103 related tasks to a html file

> ls -k *CS2103* — export -html cs2103tasks.html

f) show detail of CS2103 tasks, sort by priority

> ls -g CS2103 — sort "pri" — task

2.5.2 command mapping

> map "ls" "ls -f no"

TaskManager support custom command mapping/alias. General format of map is like this:

> map "command>" "command>" > map "of aliased command>" "of original command>"

A simple mapping is like the previous example. This maps "ls" to "ls -f no", which means hide finished tasks when listing. To have another command which shows all tasks, do this:

> map "lsa" "ls" > map "ls" "ls -f no"

Notice: The order of mapping matters as commands are executed one by one. Reversing the order of these two mapping will not work.

More complex mapping makes use of $ symbol. There are two kind of $ symbols: $0 Means all characters from surrent position $1, $2, $3 .... Means one token

E.g.

> map "tomorrow $1" "add $1 -t 1d" > tomorrow "Finish user guide"

The latter command will be parsed as > add "Finish user guide" -t 1d and a new task will be added with the deadline to be 1 days later

> map "do $1 at $2" "add $1 -t $2" > do "Laundry" at 4h

The latter command will be parsed as > add "Laundry" -t 4h and a new task called Laundry will be added with the deadline to be 4hours later

> map "ls $0" "ls -f no $0" > ls > ls -g cs2103

The second command will be parsed as > ls -f no and will list out all unfinished tasks. The third command will be parsed as > ls -f no -g cs2103 and will list out all unfinished cs2103 tasks.

Notice: TUI uses ls to retrieve tasks. Mapping ls to something else will affect behaviour of TUI.

2.5.3 taskManager script

Task manager commands can be saved in a single script file and be executed using run command.

$ cat tmscript ls map "ls" "ls -f no" ls

$ ./taskManager

Task Manager V 0.1 exit>to quit. help>for more instructions ==========
> run tmscript

1 f Sample task 1. This also has high priorty 2 Sample task 2. This has high priority 3 Sample task 3. This is the latest

2 Sample task 2. This has high priority 3 Sample task 3. This is the latest

Notice: The first 3 tasks are the result of the first ls in the script. The last 2 tasks are the result of the second ls in the script. Because "ls" is mapped

to "ls -f no", finished tasks are not shown by the second ls. TaskManager scripts are pure text files. ".txt" suffix is omitted here. In windows, suffix is hidden by default. To show it, check corresponding checkbox in tools ->folder options. Windows may not know how to open file with no suffix.

2.5.4 startup script

By default, taskManager executes a special script everytime when it is started. This script is located at:

~/.tmrc <linux/mac os>%USERPROFILE%\tmrc.txt <Windows>

This file can be editted to include customized setting.

E.g.

a) To switch to the interactive user interface by default, add this line into tmrc tui

b) To save a backup file when taskManager is started, add this line into tmrc export /tmp/backupTasklist.xml

c) To show tasks when taskManager is started, add this line into tmrc ls

d) To remove finished tasks when taskManager is started, add this line into tmrc ls -f yes — rm

e) To run a script with all self-defined mappings when taskManager is started, add this line into tmrc run /home/myusername/mymappings

5.5 Talk to taskManager

For all inputs that cannot be recognized by taskManager as a command, it will be treated as natural language sentense. TaskManager will try its best to recognize it and give correct response.

E.g.

> what do I do today?

All tasks due today will be listed out.

2.6 Compilation and Installation

2.6.1 Microsoft Windows

On Windows, taskManager can be built with Visual Studio in the following steps:

1. Start Visual Studio with "C++ Development Settings".
2. Create a win32 console project.
3. Drag all source files into the solution folder. Files should be automatically categorized into header files and source files.

4. Edit project configuration, under general, set character set to be multi-byte characters.

5. Edit project configuration, under linker, add pdcurses.lib to additional libraries and add the folder containing pdcurses.lib to library search directories.

6. Copy pdcurses.dll to %WINDIR%\system32\or the directory your executable will be generated.

7. Build the solution.

To build taskManager with TUI, pdcurses library is needed. It is free and can be downloaded here: http://sourceforge.net/projects/pdcurses/files/

Edit the project property to include the pdcurses.lib and in the linker options.

2.6.2 Unix-like Operating Systems

In Unix-like operating systems like GNU Linux and Mac OS X, start a shell, change directory to taskManager's source folder and type:

```
$ make
$ sudo make install
```

da "make install" is optional. It just makes taskManager available system wide, which copys the executable and man pages to cooresponding directories.

The text UI is built by default, which requires ncurses library. It ships with most Linux distributions and Mac OS. If not, it can be installed with the package manager (apt-get, yum, pacman on various Linux distributions and port on Mac).

3 Developer Guide

ahah