TaskPaper

User's Guide

Welcome

Before reading this guide, you may want to watch TaskPaper's screencast. The screencast is an online movie that demonstrates TaskPaper's most important features. When you need more detail use this guide. If you have questions please visit TaskPaper's user forums.

- User forums link
  http://www.hogbaysoftware.com/forums
- TaskPaper FAQs page
  http://www.hogbaysoftware.com/products/taskpaper/faq
- TaskPaper screencast link
  http://www.hogbaysoftware.com/products/taskpaper

TaskPaper’s Story

From 2001 to 2007 I created and worked on another program called Mori. My goal was to create the perfect information manager. It was fast, it was flexible, it was full featured, but it never really worked well for me.

Instead of using Mori I would find myself writing my notes and to-do's in simple text files on my Desktop. As you might guess this was a little frustrating. I tried to use Mori, but I kept going back to my text files. Eventually I sold Mori, and continued to keep my to-do lists in text files.

At about that time a number of well-designed to-do list applications such as OmniFocus and Things were gaining popularity. But for me they still felt very much like Mori—they did everything that I wanted them to do, but for some reason I still preferred my simple text files.

However, text files aren’t perfect. My to-do list text files were always messy. Being free to make a mess is important to me, but without any structure I got overwhelmed when my lists were long.

To get more organized I started adding the simplest structure that I could think of to my lists. For each project, I typed the project name and ended that line with a colon. For each task, I indent it under its project and started the line with a dash followed by a space.

```
project 1:
    - task 1
    - task 2
    - task 3
```

I typed everything else in free form and called those lines notes.

That small amount of structure made a big difference. I could still be as messy as I wanted to be, but I always had this simple structure to fall back on. My lists now had an overall structure, even if some parts were still messy.

I continued to tweak the system and eventually turned it into TaskPaper. TaskPaper has grown beyond my simple text files, but at its core it remains a simple system for organizing your to-do list in a text file.

— Jesse

Working with TaskPaper

Remember, it's just text.

TaskPaper does many cool things, but in the end you are just editing a text file. If you know how to type, you already know most of what you need to effectively use TaskPaper.

Getting Started

TaskPaper knows about four things: **projects**, **tasks**, **notes**, and **tags**. As you type, these items are auto-formatted so that your lists are easier to read.

- To create a **project**, type a line ending with a colon:

  ```
  Groceries:
  ```

- To create a **task**, type a line starting with a dash followed by a space:

  ```
  - Milk
  ```

- To create a **note**, type any line that isn't recognized as a project or task:

  ```
  This is a note that I added.
  ```

- To create a **tag**, type the @ symbol followed by a name. Tags can optionally have a value in parentheses after the tag name:

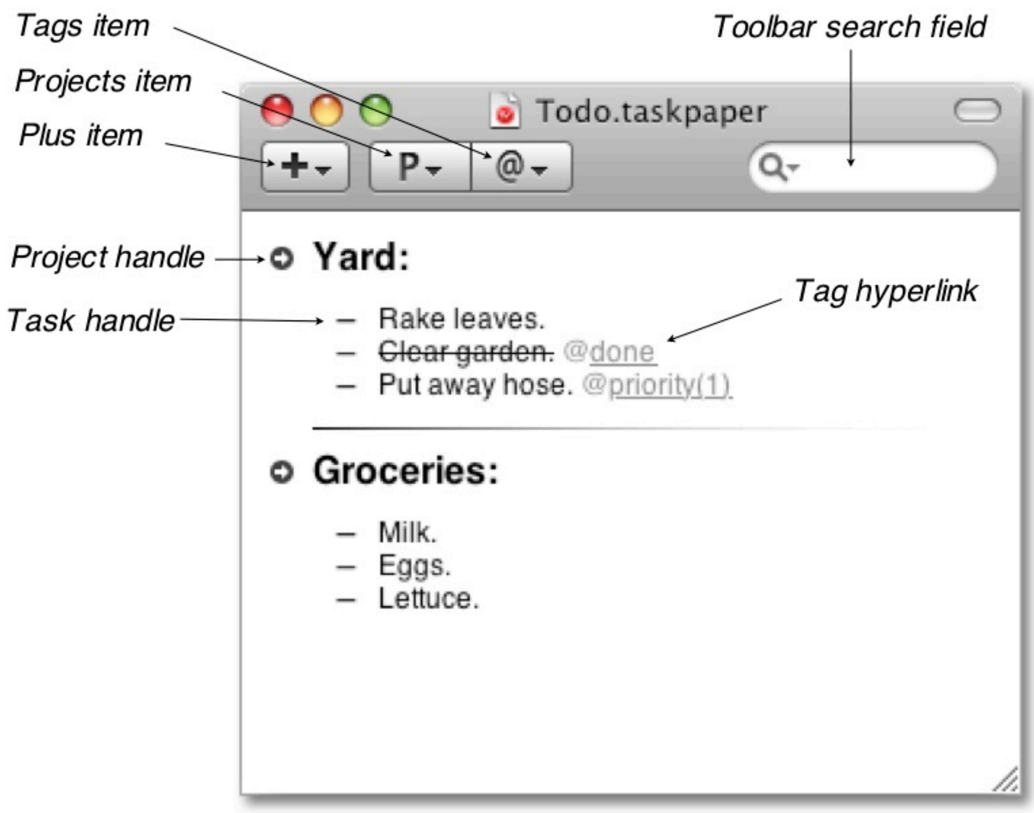
  ```
  @priority(1)
  ```

Here's an example TaskPaper file:

```
Yard:
    - Rake leaves
    - Clear garden @done
    - Put away hose @priority(1)
Groceries:
    - Milk
    - Eggs
    - Lettuce
```

You use these basic parts in any way that you like. TaskPaper doesn't force a particular system on you; it provides the basic to-do list elements and then you use them as you see fit.

TaskPaper’s Main Window



Using TaskPaper Lists

You now know how to create lists of your projects, tasks, and notes. It’s certainly simple, but so far it’s not much different than typing your to-do lists in a word processor. So let’s learn what else TaskPaper does to make working with your lists easier.

Adding new entries

TaskPaper makes adding new entries easier in two ways:

- To add an entry with your mouse, click the Plus (**+**) toolbar item. Press the keyboard shortcuts, listed in that popup menu, for an even faster approach.
- To use TaskPaper’s auto-formatted lists, press Enter when on a task line—the next line will automatically be formatted as a new task at the same indentation level.

Creating outlines

TaskPaper recognizes outline structures in your lists. Outlines are best illustrated with an example:

```
Project 1:
    - Task 1
            - Task 2
            - Task 3
    - Task 4
```

Indentation (use of tabs) is used to determine the outline structure. In this example `Project 1` is the direct parent of `Task 1` and `Task 4`. `Task 1` also has two subtasks, `Task 2` and `Task 3`.

These relationships are used by some of TaskPaper's commands. For example if you move Task 1 by dragging and dropping, its two subtasks will be moved along with it.

To unindent a line with the keyboard, press Shift+Tab.

Moving entries

TaskPaper provides four ways to move entries.

- To drag and drop, click and drag on the entry's handle.
- To copy and paste, select the entry's text then copy (or cut) just like in a word processor.
- To quickly move an entry to any project, choose Entry > Move to Project.
- To move an entry by one line, press Command+Control+Up or Down Arrow.

Tagging entries

TaskPaper provides five ways to add tags to entries.

- To create a new tag, type the @ symbol followed by the tag name.
- To begin tag auto-complete, type the `@` symbol.
- To apply an existing tag, choose Entry > Tag With.
- To apply `@done`, click on the entry's handle, or press Command-D.
- To apply `@today`, press Command-Y.

Completing tasks

TaskPaper provides three options for dealing with completed tasks.

- To cross out an entry, apply the `@done` tag. Type `@done` or use Command-D. You can also use the mouse to click the task handle.
- To delete an entry and its subentries, press Command-K.
- To move `@done` entries to the “Archive” project, press Command-Shift-D.

Filtering TaskPaper Lists

TaskPaper’s toolbar search field allows you to filter your entries. Matching entries remain visible and fully editable; everything else is hidden. TaskPaper’s query language is quite powerful, but most of that power can be ignored until you need it.

- To search, type your search terms into the toolbar search field. You can quickly jump to the search field by pressing Command+Shift+F.
- To cancel your search, clear the toolbar search field or press Command-[.
- To learn more advanced search features, read “*Searching in TaskPaper*”.

Filtering entries by project

TaskPaper makes it easy to focus on a single project.

- To focus on a project, click the arrow handle next to the project’s name. This automatically begins a search that filters everything else out. Or press Command-[.
- To navigate between projects, click the Projects (**P**) toolbar item and select a new project. Press Command-L to switch between projects without using the mouse.

Filtering entries by tag

TaskPaper makes it easy to filter entries by tag.

- To filter entries by a tag, click on that tag’s hyperlink (underline portion).
- To filter entries by a tag and value combination, click on the value part of the tag’s hyperlink.
- To switch between different tag filters, click the Tags (**@**) toolbar item. Use Command-Shift-L to switch between tag filters.

Using Tabs

You can use tabs in a window view a single document with several different search views.

- Open a new tab by pressing Command+T or choosing Window > New Tab View
- Close a tab by pressing Command+W (current tab) or clicking the X in the tab.
- The tab title is set by the active search or current Project location of the insertion point.

TaskPaper Tips & Tricks

This section lists TaskPaper tips & tricks that are not essential to using TaskPaper, but can be useful to lean.

How I Use TaskPaper

First I turn off most of the “visual interface” so that I have a very simple-looking text window.

To use TaskPaper effectively in this minimal mode:

1. Hide extra features like the window toolbar, the projects list, and tab views.
2. Learn the Command-L shortcut to switch between projects.
3. Learn the Command-Shift-L shortcut to switch between tag views.
4. Learn the Command-Shift-Return shortcut to show the Quick Entry Window.
5. Learn the Command-Command (press it twice quickly) keyboard shortcut to access the toolbar search field when the toolbar is hidden.

For each big thing that I am working on I create a project. For example, I have a project for each of the applications that I’m working on, books that I want to read, and website work that I need to do. Right now I have about 30 projects overall.

I also have a default “Inbox” project at the top of my list. That’s where I put new tasks that don’t fit into an existing project. Later, when I have time, I’ll go back and figure out which project they fit in, or I’ll give them their own project.

I then add related tasks and notes to my projects. For well-defined projects (like a book list or grocery list) I just keep a simple list of tasks. For less-defined projects (like a software application) I also have many messy notes mixed in with the tasks.

When I complete a task, most of the time I delete it (Command-K) right away. For tasks that I need a record of, I’ll tag them `@done` and then I’ll press Command-Shift-D to send them to the “Archive” project.

I don’t use tags very often, except for the `@done` tag and sometimes the `@today` tag. Other people use tags quite extensively, especially people who are following the “Getting Things Done” (GTD) style system. TaskPaper’s tags work well for assigning GTD contexts to your tasks.

Quick Entry Window

The Quick Entry Window makes it possible to add tasks to a project without having to first navigate to that project. It even works when you are working in another application such as Mail or Safari.

- To show the Quick Entry Window, press Command-Shift-Return. (TaskPaper must be running, but it doesn't need to be the front application)
- To save and close the Quick Entry Window, press Command-Shift-Return again.
- To close and not save the Quick Entry Window, press Escape.

Projects List

The Projects List displays a separate list containing only your projects..

- To show the projects list, choose View > Show Projects List.
- To focus on a project, click the project name in the Projects List.
- To move a task to a project, drag it from the main view to a project in the Projects List.

Views

Views allow you to keep multiple views of your to-do list. Each view maintains its own search, selection, and scrollbar state.

- To create a new View, choose Window > New View.
- To open a link in a new View, press Command while clicking on the link.

Service Menu Items

Service menu items provide similar functionally to the quick entry window. They allow you to add new tasks to your list without switching to TaskPaper. To use the service menu:

1. Select text in any application that you want to make into a task.
2. Choose Application Name > Services > TaskPaper > New Entry and a new task will be created from your selected text and inserted into your lists Inbox project.
3. To assigned keyboard shortcuts to service menu items, and in general clean up your services menu I recommend "*Service Scrubber*".

Themes

Themes specify the fonts, colors, and other settings that TaskPaper uses to format and display your lists. Themes can be customized in two ways. The recommended way is to use TaskPaper's Preference pane to change theme values. For more control, you can modify the theme file directly, but this process isn't well documented, and your custom theme files aren't likely to work in future versions of TaskPaper.

Change theme values

To change theme values from the Preference pane:

1. Choose TaskPaper > Preferences to open Preferences.
2. Check "Change theme colors & fonts" and click the "Choose…" button.
3. Select values in the themes override sheet. Your lists are updated immediately.

Create your own theme file

To create your own theme file (possibly a headache-inducing process):

1. Choose "Open Themes Folder" from the Theme Options popup in Preferences.
2. Duplicate the "Standard" theme, and make sure that your copy shares the same "taskpapertheme" filename extension.
3. Restart TaskPaper so that your new theme is visible in Theme Options.
4. Edit your theme using a text editor. To reload your theme, select it again in Theme Options.
5. If you have questions please post them in TaskPaper's user forum.

Searching in TaskPaper

There are two ways to search in TaskPaper. You can use the standard Find & Replace method that you know from working with word processors. Alternatively, you can use TaskPaper's toolbar search field which allows you to search all TaskPaper entries at once.

Find & Replace

Find & Replace works just like it does in a standard word processor. Press Command-F to show the Find & Replace panel. Press Command-G to find the next match. This is the technique that I use most; it's fast, familiar, and very effective.

Toolbar Search Field

The toolbar search field gives you the option to search all of your entries at once. Matching entries are displayed in text view and remain fully editable.

- To search for a word or phrase, type the phrase into the search field. All entries containing the phrase will be matched.

Besides simple phrase searches, TaskPaper also has a query language that allows your searches to be more specific and powerful.

While you don't need to use the query language, sometimes its syntax can interfere with a basic phrase search. You'll notice that the phrase turns red in color, or has some of its words highlighted in bold. In such cases, enclose your search in quotes. For example, try searching for:

```
this is not what I want
```

Note that the word "`not`" displays in bold. This means TaskPaper is trying to run a logical search—if that's not what you want, just enclose the search in quotes like this:

```
"this is not what I want"
```

TaskPaper will act as expected, matching entries that contain the phrase "`this is not what I want`".

TaskPaper's query language

Now let's look at TaskPaper's advanced query language. We'll start with some examples:

```
project = Inbox
```

Matches all entries that are in the Inbox project.

```
project = Inbox and not @done
```

Matches all entries that are in the Inbox and that are not tagged with `@done`.

```
project Inbox and not @done and (@priority > 1 or @today)
```

Matches all entries that are in the Inbox and that are not tagged with `@done` and that have a `@priority` tag with a value greater than 1 or are that are tagged with `@today`.

```
@today+d
```

Matches all entries tagged with @today and shows their descendants. So for instance if a note entry is indented under another entry that is tagged with @today, then adding `+d` will also show that note entry.

TaskPaper's query language syntax

The query language syntax follows this basic pattern:

```
<attribute> <relation> <search term>
```

In the first query language example, "`project = Inbox`", follows the pattern exactly. "`project`" is the attribute, "`=`" is the relation, and "`Inbox`" is the search term. But, as you might have noticed above, if you just type "`project Inbox`" or even just "`Inbox`" that works too. What's happening in that case?

A search like "`Inbox`" works because you don't have to fully specify each search. If you leave out part of the search, TaskPaper will anticipate your meaning. If you don't provide an attribute, TaskPaper will assume the default "`line`" attribute. If you don't provide a relation, TaskPaper will assume the default "`contains`" relation. Because of these defaults the following four searches are all equivalent:

```
line contains Inbox
line Inbox
contains Inbox
Inbox
```

Attributes

Each entry has built-in attributes that you can use in your searches as shown in this table:

`type`	The entry's type. This attribute will have the value project, task, or note. For example search for "`type = task`" to see just your tasks. Because project is a keyword when searching for `type =` "`project`" you need to enclose "`project`" in quotes.
`line`	The entry's entire line of text. For example you can use "`line contains joe`" to match all entries that contain the text "`joe`".

`content`	The entry’s text content. This is similar to the `line` attribute, except it doesn’t include the entry formatting, trailing tags, or the end of line character. The search “`content = Inbox`” will match any of these lines:
	
	`Inbox @done`
	`Inbox:`
	`- Inbox`

`level`	The number of tabs at the beginning of the entry’s line. For example you can use “`level = 0`” to only match entries that are not indented.
`parent`	The text content of the entry’s parent in the outline. For example, use “`parent contains subproject`” to match all entries whose direct parent contains the text “`subproject`”.
`project`	The text content of any of the entry’s enclosing projects. The entry will match if any of its containing projects match. For example, use “`project = Inbox`” to match all entries contained in the Inbox project.
`index`	The entry’s index in list of siblings. For example, use “`index = 0`” to find all entries that are first in their parents list of children.
`uniqueid`	The entry’s unique id. For example, use “`uniqueid = 0`” to find the entry with id 0 if it exists. The `uniqueid` attribute isn’t generally useful, but is useful if you want to find certain entries via applescript and then build a search to display only those entries.

Tags

You can also use tags as search attributes.

If your search only contains a tag value, then the search will match any entry that has that tag. It doesn’t make sense to do a search that only consists of a built-in attribute like `line`, but it does make sense to do a search that consists of only a tag.

For example the search:

```
@done
```

Will match all entries that have the `@done` tag.

You can also search for the values that are associated with tags. For example you might have the tag `@priority(1)` in one of your entries. To match that entry use:

```
@priority = 1
```

That will match all entries that have a `@priority` tag with a value of `1`.

Relations

Relations define the relationship between the attribute and search term that must be met for a successful match. If no relation is specified, then `contains` is assumed. In TaskPaper all values are string values, so > and < relations work on the alphabetical order of the values that are being compared:

=	Is true if the values are equal after stripping off white-space.
==	Is true if the values are exactly character-by-character equal.
>	Is true if the attribute is alphabetically greater-than the search term.
<	Is true if the attribute is as alphabetically less-than the search term.
`contains`	Is true if the attribute case-insensitive contains the search term.
`matches`	Is true if the attribute matches the `regular expression` defined by the search term.

Because of alphabetical sort ordering, some tag value formats work better than others. For example, if you want to have `@due` tags and give them date values, then you should use a date format that sorts correctly in alphabetical order. The recommended date format for TaskPaper is YYYY-MM-DD, where YYYY's are for year, MM's are for month, and DD's are for day.

Search terms

Most of the time you can just type the search term and be done with it. However, if you want to search for something that has special meaning in TaskPaper's query language, then you need to first enclose the search term in double quotes. For example, to find all entries that contain an equal sign, your search must enclose the search term in quotes like this:

```
line contains “=”
```

Unless the search term contains a keyword you do not need to enclose the search term in quotes. This is true even if the search term has multiple words. For example this search is perfectly valid:

```
line contains my search terms and not @done
```

Matches entries that have the text “my search terms” and are not tagged with `@done`.

Logical operators

You can combine TaskPaper's basic search patterns with logical `and`, `or`, and `not` operators.

```
one and two
```

Matches entries that contain both the text `one` and the text `two`.

```
not @done
```

Matches entries that do not have an `@done` tag.

```
project Inbox and not @done and (@priority > 1 or @today)
```

Matches all entries that are part of the Inbox project, and that are not tagged with `@done`, and that have an `@priority` tag with a value greater than 1 or are tagged with `@today`.

TaskPaper Scripting

This content is useful for people who want to write software code that integrates with TaskPaper. If you are not a programmer, and don't care how TaskPaper is built, you can safely skip this chapter.

AppleScript Support

For advanced use, AppleScript support allows you to automate TaskPaper and to integrate it with other applications. For example, this script moves all entries that are tagged `@done` in the current project to the end of it's list:

```
tell application "TaskPaper"
    tell front document
        set p to containing project of selected entry
        set done_entry_ids to {}
        repeat with each in entries of p
            if exists (tag named "done" of each) then
                set done_entry_ids to done_entry_ids & id of each
            end if
        end repeat
        repeat with each in done_entry_ids
            move entry id each to end of entries of p
        end repeat
    end tell
end tell
```

To view TaskPaper's full scripting dictionary open OS X's default "Script Editor" application, and then choose File > Open Dictionary and choose TaskPaper's dictionary.

Running Scripts

Your TaskPaper scripts can be accessed from a global OS X menu or with a keyboard shortcut:

- To activate the global AppleScript menu, open AppleScript Editor.app, open Preferences and select the 'General' tab. Check the option 'Show script menu in menu bar' and look for a new script icon on the right-hand side of your menu bar.
- To add a script to the OS X script menu choose Script Menu > Open Scripts Folder and place your script there. You can also place scripts in a subfolder if you wish.
- Use System Preferences > Keyboard & Mouse > Keyboard Shortcuts to assign a keyboard shortcut to your script (you may need to add it as a Service first), or use a utility like FastScripts.

TaskPaper’s File Format

TaskPaper’s file format is fairly simple. Here’s how TaskPaper reads a file:

- Files are expected to use the UTF-8 encoding and use ‘\n’ to separate lines.
- A task is a line that begins with a hyphen followed by a space ('- ') which can optionally be prefixed (i.e indented) with tabs or spaces. A task can have zero or more tags anywhere on the line (not just trailing at the end).
- A project is a line that isn't a task and ends with a colon (':'), or a colon (':\n') followed by a newline. Tags can exist after the colon, but if any non-tag text is present, then it won’t be recognized as a project.
- A note is any line that doesn't match the task or project rules.
- Indentation level (with tabs, not spaces) defines ownership. For instance, if you indent one task under another task, then it is considered a subtask. Tasks and notes own all objects that are indented underneath them. Empty lines are ignored when calculating ownership.
- A tag has the form "@tag", i.e. it starts with an "at" character ("@"), followed by a run of non-whitespace characters. A tag can optionally have a value assigned to it. The value syntax immediately follows the tag word (no whitespace between) and is enclosed by parentheses: '(' and ')'. The value text inside can have whitespace, but no newlines. Here is an example of a tag with a value: @tag(tag's value)

Acknowledgments

This section gets tricky. My memory isn't good, and I don't keep many records. I won't list individuals (except thanks Laurel, Jaeda, Genki, Kimchi, and even DD) because I just can't keep track, but if you notice any project that I have forgotten to list here please let me know.

TaskPaper users

Thanks to everyone who has helped test and design version 2.0. It's been a fun process with many twists and turns. TaskPaper 2.0 is a much better application because of all the feedback and discussion that's taken place in the TaskPaper user forums. Thank you!

Software frameworks

Blocks Plugin Framework

Blocks is the foundation for all Hog Bay Software applications. It is open source, undocumented, and I welcome others to try it out. Blocks is heavily influenced by the Eclipse project.

Link: http://github.com/jessegrosjean

Cocoa Script Menu Framework

The code used to construct and display the script menu in Hog Bay Software applications is based on code from the Cocoa Script Menu project that is developed under the MPL 1.1 license. Blocks modifications can be found here.

Link: http://jay.tuley.name/archives/2005/10/18/Cocoa-Script-Menu-Revised-1.01

Sparkle Update Framework

The code used to find and download new software updates is based on the Sparkle software update project. Blocks modifications can be found here.

Link: http://sparkle.andymatuschak.org/

Shortcut Recorder Framework

The code used to capture and respond to keyboard shortcuts in TaskPaper 2.0's preferences is from the shortcut-recorder project.

Link: http://code.google.com/p/shortcutrecorder/

Additional help

Thanks to www.devon-technologies.com for sharing some code that makes it possible to use TaskPaper's menu items in the services menu without making TaskPaper the foreground application as a result.