Wyrd v1.0 User Manual

Paul J. Pelzl

June 22, 2005

"Because you're tired of waiting for your bloated calendar program to start up."

Contents

1 Introduction **2**

2 Installation **2**

3 Quick Start **3**

3.1 Overview 3
3.2 Navigation 3
3.3 Editing Reminders 3
3.4 Viewing Reminders 4
3.5 Searching for Reminders 4
3.6 Other Commands 4

4 Advanced Configuration **4**

4.1 `wyrdrc` Syntax 5
  4.1.1 Including Other Rcfiles 5
  4.1.2 Setting Configuration Variables 5
  4.1.3 Creating Key Bindings 5
  4.1.4 Removing Key Bindings 6
  4.1.5 Setting the Color Scheme 6
4.2 Configuration Variables 6
4.3 Calendar Operations 7
4.4 Colorable Objects 9

5 Licensing **10**

6 Acknowledgments **11**

7 Contact info **11**

8 Miscellaneous **11**

1 Introduction

Wyrd is a text-based front-end to Remind, a sophisticated calendar and alarm program available from Roaring Penguin Software, Inc.[1] Wyrd serves two purposes:

1. It displays reminders in a day-calendar view suitable for visualizing your calendar at a glance.
2. It makes creating and editing reminders fast and easy. However, Wyrd does not hide Remind's textfile programmability, for this is what makes Remind a truly powerful calendaring system.

Wyrd also requires only a fraction of the resources of most calendar programs available today.

2 Installation

This section describes how to install Wyrd by compiling from source. Volunteers have pre-packaged Wyrd for several popular operating systems, so you may be able to save yourself some time by installing from those packages. Please check the Wyrd website[2] for up-to-date package information.

Wyrd is designed to be portable to most Unix-like operating systems, including GNU/Linux, *BSD, and Mac OS X. Before installing Wyrd, your system must have the following software installed:

- OCaml[3] $\geq$ 3.08
- the ncurses library[4] (and development headers)
- Remind [5] $\geq$ 3.0.23
- GNU make[6]
- standard Unix utilities such as `cat`, `sort`, `cal`, and `less`

Wyrd may be compiled by executing the following at the root of the source tree:

```
./configure
make
```

After compiling, become root and execute

```
make install
```

to complete the installation process. The `make` command here should correspond to GNU make; on some systems (particularly *BSD), you may need to use `gmake`.

---

[1] http://www.roaringpenguin.com/penguin/open_source_remind.php

[2] http://www.eecs.umich.edu/~pelzlpj/wyrd

[3] http://caml.inria.fr

[4] http://www.gnu.org/software/ncurses/ncurses.html

[5] http://www.roaringpenguin.com/penguin/open_source_remind.php

[6] http://www.gnu.org/software/make/

3 Quick Start

This section describes how to use Wyrd in its default configuration. After familiarizing yourself with the basic operations as outlined in this section, you may wish to consult Section 4 to see how Wyrd can be configured to better fit your needs.

3.1 Overview

Before attemping to use Wyrd, learn how to use Remind. Wyrd makes no attempt to hide the details of Remind programming from the user.

You can launch Wyrd using the default reminder file by executing `wyrd`. If desired, a different reminder file may be selected by executing `wyrd <filename>`.

At the top of the window is a short (incomplete) list of keybindings.

The left window displays a scrollable day schedule view, with reminders highlighted in various colors. If the `DURATION` specifier is used for a reminder, the highlighted area is rendered with an appropriate size. Overlapping reminders are rendered using one of four different indentation levels so that all reminders are at least partially visible.

The upper right window displays a month calendar, with the color of each day representing the number of reminders it contains. The colors range across shades of white to blue to magenta as the number of reminders increases. The selected date is highlighted in cyan.

The lower right window displays a list of the untimed reminders falling on the selected date.

The bottom window displays the full text of the `MSG` for the reminder or reminders that are currently selected.

3.2 Navigation

Action	Keypress
scroll up and down the schedule	`<up>`, `<down>` or `k`, `j`
jump back or forward by a day	`<pageup>`, `<pagedown>` or `4`, `6` or `<`, `>` or `H`, `L`
jump back or forward by a week	`8`, `2` or `[`, `]` or `K`, `J`
jump back or forward by a month	{, }
jump to current date and time	`<home>`
switch between schedule and untimed reminders window	`<left>`, `<right>` or `h`, `l`
zoom in on the schedule	`z`
scroll the description window up and down	`d`, `D`

Notice that if you have a numeric keypad, the {`4, 6, 8, 2`} keys will let you move directionally in the month calendar view at the upper-right of the screen. Similarly, {`H, J, K, L`} will cause directional calendar movement using the standard mapping from `vi(1)`.

3.3 Editing Reminders

Note: By default, Wyrd is configured to use the Vim editor for modifying your reminder files. To use a different editor, see Section 4.

If you select a timeslot in the schedule view, then hit `'t'`, you will begin creating a new timed reminder. Wyrd will open up your reminder file in your favorite editor and move the cursor to the end of the file, where a new reminder template has been created. The template has the selected date and time filled in, so in many cases you will only need to fill in a `MSG` value.

Similarly, hitting `'u'` will begin creating an untimed reminder. `'w'` will create a weekly timed reminder, and `'W'` will create a weekly untimed reminder; `'m'` will create a monthly timed reminder, and `'M'` will create a monthly untimed reminder.

`'T'` and `'U'` also create timed and untimed reminders (respectively), but first will provide a selection dialog for you to choose which reminder file you want to add this reminder to. The set of reminder files is determined by scanning the `INCLUDE` lines in your default reminder file.

If you select a reminder (either timed or untimed) and hit `<return>`, you will begin editing that reminder. Wyrd will open up the appropriate reminders file in your editor and move the cursor to the appropriate `REM` line.

If you select a timeslot that contains multiple overlapping reminders, Wyrd will provide a dialog that allows you to select the desired reminder.

3.4 Viewing Reminders

Aside from viewing reminders as they fall in the schedule, you can press `'r'` to view all reminders triggered on the selected date in a `less(1)` window. Similarly, `'R'` will view all reminders triggered on or after the selected date (all non-expired reminders are triggered).

3.5 Searching for Reminders

Wyrd allows you to search for reminders with `MSG` values that match a search string. Press `'/'` to start entering a (case insensitive) regular expression. After the expression has been entered, press `<return>` and Wyrd will locate the next reminder that matches the regexp. Press `'n'` to repeat the same search. Entry of a search string may be cancelled with `<esc>`.

The regular expression syntax is Emacs-compatible.

Note: Sorry, there is no ”search backward” function. The search function requires the use of `"remind -n"`, which operates only forward in time.

3.6 Other Commands

You can exit Wyrd by pressing `'Q'`. If the screen is corrupted for some reason, hit `'Ctrl-L'` to refresh the display.

4 Advanced Configuration

Wyrd reads a run-configuration textfile (generally `/etc/wyrdrc` or `/usr/local/etc/wyrdrc`) to determine key bindings, color schemes, and many other settings. You can create a personalized configuration file in `$HOME/.wyrdrc`, and select settings that match your usage patterns. The recommended procedure is to “include” the `wyrdrc` file provided with Wyrd (see Section 4.1.1), and add or remove settings as desired.

4.1 `wyrdrc` Syntax

You may notice that the `wyrdrc` syntax is similar to the syntax used in the configuration file for the Mutt email client (muttrc).

Within the `wyrdrc` file, strings should be enclosed in double quotes (`"`). A double quote character inside a string may be represented by `\"` . The backslash character must be represented by doubling it (`\\`).

4.1.1 Including Other Rcfiles

Syntax: `include` *`filename_string`*

This syntax can be used to include one run-configuration file within another. This command could be used to load the default `wyrdrc` file (probably found in `/etc/wyrdrc` or `/usr/local/etc/wyrdrc`) within your personalized rcfile, `~/.wyrdrc`. The filename string should be enclosed in quotes.

4.1.2 Setting Configuration Variables

Syntax: `set` *`variable=value_string`*

A number of configuration variables can be set using this syntax; check Section 4.2 to see a list. The variables are unquoted, but the values should be quoted strings.

4.1.3 Creating Key Bindings

Syntax: `bind` *`key_identifier operation`*

This command will bind a keypress to execute a calendar operation. The various operations, which should not be enclosed in quotes, may be found in Section 4.3. Key identifiers may be specified by strings that represent a single keypress, for example `"m"` (quotes included). The key may be prefixed with `"\\C"` or `"\\M"` to represent Control or Meta (Alt) modifiers, respectively; note that the backslash must be doubled. A number of special keys lack single-character representations, so the following strings may be used to represent them:

- `"<esc>"`
- `"<tab>"`
- `"<enter>"`
- `"<return>"`
- `"<insert>"`
- `"<home>"`
- `"<end>"`
- `"<pageup>"`
- `"<pagedown>"`

- `"<space>"`
- `"<left>"`
- `"<right>"`
- `"<up>"`
- `"<down>"`
- `"<f1>"` to `"<f12>"`

Due to differences between various terminal emulators, this key identifier syntax may not be adequate to describe every keypress. As a workaround, Wyrd will also accept key identifiers in octal notation. As an example, you could use `\024` (do *not* enclose it in quotes) to represent Ctrl-T.

Multiple keys may be bound to the same operation, if desired.

4.1.4 Removing Key Bindings

Syntax: `unbind` *`key_identifier`*

This command will remove all bindings associated with the key identifier. The key identifiers should be defined using the syntax described in the previous section.

4.1.5 Setting the Color Scheme

Syntax: `color` *`object foreground background`*

This command will apply the specified foreground and background colors to the appropriate object. A list of colorable objects is provided in Section 4.4. Wyrd will recognize the following color keywords: `black, red, green, yellow, blue, magenta, cyan, white`.

4.2 Configuration Variables

The following configuration variables may be set as described in Section 4.1.2:

- `reminders_file`
  Controls which Remind file Wyrd will operate on. The default is ~/.reminders .
- `edit_old_command`
  Controls the command used to edit a pre-existing reminder. The special characters `'%f'` and `'%n'` will be substituted with a filename to edit and a line number to navigate to within that file.
- `edit_new_command`
  Controls the command used to edit a new reminder. The special character `'%f'` will be substituted with a filename to edit. Ideally, this command should move the cursor to the last line of the file, where the new reminder template is created.

- `timed_template`
  Controls the format of the `REM` line created when editing a new timed reminder. The following character substitutions will be made: `'%M'` - month name, `'%d'` - day of the month, `'%y'` - year, `'%h'` - hour, `'%m'` - minute, `'%w'` - weekday name.

- `untimed_template`
  Controls the format of the `REM` line created when editing a new untimed reminder. The substitution syntax is the same as for `timed_template`.

- `template`*N*
  Controls the format of a generic user-defined `REM` line template; *N* may range from 0 to 9. The substitution syntax is the same as for `timed_template`.

- `busy_level1`
  An integer value specifying the maximum number of reminders in a day, colored using the color scheme for `calendar_level1`.

- `busy_level2`
  Same as above, using the `calendar_level2` color scheme.

- `busy_level3`
  Same as above, using the `calendar_level2` color scheme rendered in bold.

- `busy_level4`
  Same as above, using the `calendar_level3` color scheme. Any day with more reminders than this will be rendered using the `calendar_level3` color scheme rendered in bold.

- `week_starts_monday`
  A boolean value (`"true"` or `"false"`) that determines the first day of the week.

- `schedule_12_hour`
  A boolean value that determines whether the timed reminders window is drawn using 12- or 24-hour time.

- `selection_12_hour`
  A boolean value that determines whether the selection information is drawn with 12- or 24-hour time.

- `status_12_hour`
  A boolean value that determines whether the current time is drawn using a 12- or 24-hour clock.

- `description_12_hour`
  A boolean value that determines whether reminder start and end times are drawn using 12- or 24-hour time in the description window.

For maximum usefulness, `busy_level1` < `busy_level2` < `busy_level3` < `busy_level4`.

4.3 Calendar Operations

Every Wyrd operation can be made available to the interface using the syntax described in Section 4.1.3. The following is a list of every available operation.

- `scroll_up`
  move the cursor up one element
- `scroll_down`
  move the cursor down one element
- `next_day`
  jump ahead one day
- `previous_day`
  jump backward one day
- `next_week`
  jump ahead one week
- `previous_week`
  jump backward one week
- `next_month`
  jump ahead one month
- `previous_month`
  jump backward one month
- `home`
  jump to the current date and time
- `zoom`
  zoom in on the day schedule view (this operation is cyclic)
- `edit`
  edit the selected reminder
- `scroll_description_up`
  scroll the description window contents up (when possible)
- `scroll_description_down`
  scroll the description window contents down (when possible)
- `new_timed`
  create a new timed reminder
- `new_timed_dialog`
  same as previous, with a reminder file selection dialog
- `new_untimed`
  create a new untimed reminder
- `new_untimed_dialog`
  same as previous, with a reminder file selection dialog

- `new_template`*N*
  create a new user-defined reminder using `template`*N*, where *N* may range from 0 to 9
- `new_template`*N*`_dialog`
  same as previous, with a reminder file selection dialog
- `switch_window`
  switch between the day schedule window on the left, and the untimed reminder window on the right
- `begin_search`
  begin entering a search string
- `search_next`
  search for the next occurrence of the search string
- `view_remind`
  view the output of `remind` for the selected date
- `view_remind_all`
  view the output of `remind` for the selected date, triggering all non-expired reminders
- `refresh`
  refresh the display
- `quit`
  exit Wyrd
- `entry_complete`
  signal completion of search string entry
- `entry_backspace`
  delete the last character of the search string
- `entry_cancel`
  cancel entry of a search string

4.4 Colorable Objects

Each of Wyrd's on-screen elements may be colored by the color scheme of your choice, using the syntax defined in Section 4.1.5. The following is a list of all colorable objects.

- `help`
  the help bar at the top of the screen
- `timed_default`
  an empty timeslot in the day-schedule window
- `timed_reminder1`
  a nonempty timeslot in the day-schedule window, indented to level 1
- `timed_reminder2`
  a nonempty timeslot in the day-schedule window, indented to level 2

- `timed_reminder3`
  a nonempty timeslot in the day-schedule window, indented to level 3
- `timed_reminder4`
  a nonempty timeslot in the day-schedule window, indented to level 4
- `untimed_reminder`
  an entry in the untimed reminders window
- `timed_date`
  the vertical date strip at the left side of the screen
- `selection_info`
  the line providing date/time for the current selection
- `description`
  the reminder description window
- `status`
  the bottom bar providing current date and time
- `calendar_labels`
  the month and weekday labels in the calendar window
- `calendar_level1`
  calendar days with low activity
- `calendar_level2`
  calendar days with medium activity
- `calendar_level3`
  calendar days with high activity
- `calendar_today`
  the current day in the calendar window
- `left_divider`
  the vertical line to the left of the timed reminders window
- `right_divider`
  the vertical and horizontal lines to the right of the timed reminders window

5 Licensing

Wyrd is Free Software; you can redistribute it and/or modify it under the terms of the GNU General Public License (GPL), Version 2, as published by the Free Software Foundation. You should have received a copy of the GPL along with this program, in the file `'COPYING'`.

6 Acknowledgments

Thanks, of course, to David Skoll for writing such a powerful reminder system. Thanks also to Nicolas George, who wrote the OCaml curses bindings used within Wyrd.

7 Contact info

Wyrd author: Paul Pelzl `<pelzlpj@eecs.umich.edu>`
Wyrd website: `http://www.eecs.umich.edu/~pelzlpj/wyrd`

Feel free to contact me if you have bugs, feature requests, patches, etc. I would also welcome volunteers interested in packaging Wyrd for various platforms.

Wyrd is developed with the aid of the excellent GNU Arch RCS[7]. Interested developers are advised to track Wyrd development via my public archive:

```
        pelzlpj@eecs.umich.edu--2005 \
                  http://www-personal.engin.umich.edu/~pelzlpj/tla/2005
```

Primary development may be found on branch `wyrd--main` .

Wyrd uses tla's "configs" support. After getting a copy of the source, run `"tla build-config dist.arch"` in the project tree root to grab the extra packages that Wyrd depends on.

8 Miscellaneous

"Wyrd is a concept in ancient Anglo-saxon and Nordic cultures roughly corresponding to fate." – *Wikipedia*

---

[7] http://www.gnu.org/software/gnu-arch/