NAME

pcal − generate PostScript (or HTML) calendars

SYNOPSIS

pcal [**−e**|**−f** *cal*] [**−o** *file*] [**−l** | **−p**] [**−P** [letter | legal | a4 | tabloid]] [**−j** | **−J**] [**−m** | **−M**] [**−g** *weekday*[*−weekday*]|all|holiday] [**−O** *weekday*[*−weekday*]|all|holiday] [**−G** *weekday*[*−weekday*]|all|holiday] [**−b** *weekday*[*−weekday*]|all|holiday] [**−s** [*day_numerics_color*][*/empty_day_box_fill_color*]] [**−F** *day*] [**−A**|**−E**] [**−X** *xtrans*] [**−Y** *ytrans*] [**−x** *xscale*] [**−y** *yscale*] [**−t** [*title_font*][*/size*]] [**−d** [*day_font*][*/size*]] [**−n** [*text_font*][*/size*]] [**−L** *footer_str*] [**−C** *footer_str*] [**−R** *footer_str*] [**−N** *notes_str*] [**−D** *symbol*] [**−U** *symbol*] [**−B**] [**−#** *n*] [**−S** | **−k** | **−K**] [**−w**] [**−I**] [**−c** | **−H**] [**−q**] [**−z** *time_zone*] [**−h** | **−u** | **−v**] [**−a** *output_language*] [**−r** [*mapping*] [**−T** [B|I|R]] [**−W** [left|center|right]] [month] [year] [nmonths]

DESCRIPTION

Pcal generates PostScript to produce landscape or portrait calendars for any month and year. The arguments **month**, **year**, and **nmonths**, if provided, should be numeric. The **month** value should be in the range 1 − 12, and the **year** value should be specified as 1 or 2 digits (in which case it will be interpreted as that year in the current century) or as the full 4-digit year. If no numeric arguments are provided, the calendar for the current month and year will be generated.

If one numeric argument is provided, it is interpreted as the **year** value, and calendars for the entire year will be generated. Otherwise, **nmonths** months, starting with **month** and **year**, will be generated.

For whole-year calendars (i.e. when the **−w** option is given), the command line arguments are interpreted somewhat differently. By default, all months in the current year are printed, starting with January. If the **month** argument alone is given, it is expected to be the desired **year** to print, and prints all of the months in the given year. If both **month** and **year** are given, then 12 consecutive months are printed starting at the given month and year. If the **month**, **year**, and **nmonths** arguments are all present, printing begins with the given month and year and **nmonths** months are printed, rounded up to the nearest multiple of 12.

The Date File (Configuration File)

By default, *pcal* simply prints an empty calendar. Its real power is in its ability to place ‘‘events’’ (and, for monthly-format PostScript calendars, Encapsulated PostScript images [e.g. photos and icons]) in appropriate days on the (PostScript or HTML) calendar, thus allowing the user to create personalized calendars. This is achieved through the use of the ‘‘date file’’, also known as the ‘‘configuration file’’.

The default date/configuration file is expected to be named *.calendar* (*pcal.dat* under MS-DOS), or *calendar* for compatibility with older versions. *Pcal* will look in several places for such a file. First, if the environment variable **PCAL_DIR** is defined, *pcal* searches the directory indicated by that variable. Next, *pcal* searches the user’s home directory (as specified by the **HOME** environment variable). If neither **PCAL_DIR** nor **HOME** is defined, *pcal* searches the current directory instead. Finally, if enabled (via the ‘SEARCH_PCAL_DIR’ flag) when *pcal* was built, the directory where the *pcal* executable resides will be checked. (This final search of the executable directory has been disabled in the version shipped with Debian.) If no date file is found, an empty calendar is printed; no error is generated.

Alternatively, the name of the date file (and, optionally, the path where it can be found) can be specified using the **−f** command-line option. See the **OPTIONS** section for more details.

Every *pcal* distribution comes with an ’examples’ directory. The ‘pcal-cfg.txt’ file that is located there contains a myriad of examples of settings that can be used in your own configuration file. Please check it out for lots of useful ideas. Furthermore, that directory contains several language/country-specific examples (including holiday and other event definitions) in various ‘calendar_xx.txt’ files, where ‘xx’ represents the 2-letter language code (e.g. ’calendar_de.txt’ is the German example file).

If a date file is found, it will be searched for lines with leading dates matching the requested month and year.

Any text following the dates found will be printed on the calendar under the appropriate day of the month. Encapsulated PostScript (EPS) images are handled similarly as described in a later subsection.

troff-style escape sequences \fB, \fI, \fP, and \fR may be used to set the font style to Bold, Italic, the previous font style, or Roman respectively. For those more familiar with HTML, <B>, <I>, </B>, and </I> may be used instead to enable/disable Bold or Italic font styles. The font style is reset to Roman after each line break.

Using the ‘include’ pre-processor directive (described in the section entitled ‘Pre-Processor Functionality’, below), other configuration files can be processed from within an existing configuration file. That is, you can ‘nest’ configuration files as needed.

Dates (essentially ‘events’) in the configuration files may be expressed in any of several formats:

- <ordinal> <day_spec> in <month_spec>{*} {<text>}
- {<ordinal>} <day_spec> <prep> <date_spec>{*} {<text>}
- <date_spec>{*} {<text>}
- <pre_defined_event>{*} {<text>}

Where:

<month_name> := first 3+ characters of name of month, or “all”

Note: *pcal* looks for names of the days of the week prior to names of months when parsing event date specifications. Furthermore, some languages (e.g. French and Finnish) have a month name whose first 3 letters are the same as the first 3 letters of one of the names of the days of the week. Because of this, the specification in such a language of any month name which collides thusly must use 4 or more letters to distinguish it from the name of the day of the week with which it ‘collides’.

<month_spec> := <month_name>, or “year”

<day_spec> := first 3+ characters of name of weekday, “day”, “weekday”, “workday”, “holiday”, “nonweekday”, “nonworkday”, “nonholiday”, “new_moon”, “first_quarter”, “full_moon”, or “last_quarter”

<ordinal> := any ordinal number (“1st”, “2nd”, etc.), “first” ... “fifth”, “last”, “odd”, “even”, or “all”

<prep> := “on”, “before”, “preceding”, “after”, “following”, “on_or_before” (“oob”), “on_or_after” (“ooa”), “nearest”, “nearest_before“, or “nearest_after“

<pre_defined_event>
:= “Christmas”, “Thanksgiving”, “Easter”, “Good_Friday”, “GEaster” (Orthodox Easter), “Gstgeorge” (Orthodox holiday), and “Gmarcus” (Orthodox holiday).

<sep> := one or more non-numeric, non-space, non-‘*’ characters

<month> := a numeric month (1-12)

<day> := day of month (1-31)

<year> := a numeric year

<text> := the text to be displayed for this event; if the text begins with the constant string “image:”, then it is interpreted as a specification of an Encapsulated PostScript (EPS) image rather than as simple text; more information on specifying EPS images is available in a later section of this document

If the **−A** option (American date formats, the default) is given:

<date_spec> := <month_name> <day> | <month><sep><day>{<sep><year>}

If the **−E** option (European date formats) is given:

<date_spec> := <day> <month_name> | <day> <month> |
<day><sep><month>{<sep><year>}

The “Notes” box (see below) uses the first of the current month as the default date. All footer strings use the first of the current month in single-month mode and the first of the starting month in whole-year mode.

Examples:

```
last Monday in May*         Memorial Day Holiday

all Fridays in Oct          Status Meeting, 11 AM
first workday in all        %−B progress report due
all Fri in all              \fBTime card due,\fP 3 PM
all Monday in all           Fiscal week %0W
−2nd workday in all         Schedule for %+B due %+2D
2nd full_moon in all        Blue Moon
Fri on_or_before all 15     Pay Day
even Fridays in year        Pay Day
183rd day of year           Mid−year (%l days left)

Tue after first Mon in Nov  Election Day (USA)

4th Thu in Nov*             Thanksgiving
Fri after 4th Thu in Nov*   Day after Thanksgiving
workday nearest 12/25*      Holiday

12/25/04*                   Christmas     # American
25.12.04*                   Christmas     # European
25. 12.*                    Christmas     # European

Dec 25*                     Christmas     # American
25 Dec*                     Christmas     # European
25. Dec*                    Christmas     # European

Fri on all 13               Avoid black cats!   # 'Friday the 13th'
```

Any non-numeric character may separate numeric dates. Holidays may be flagged by following the date immediately with ‘*’ as in the examples above; this will cause the date numerics to be printed in the color specified by the **−s** option (default = gray) and will cause the associated text (on monthly-format calendars) to be placed adjacent to the numeric date in the day box rather than below the numeric date (as is done for all non-holiday events). “Each” and “every” are accepted as synonyms for “all”, and any word may be used in place of “in”. The abbreviations “oob” and “ooa” may be used in place of the keywords “on_or_before” and “on_or_after”, respectively. “Nearest” attempts to match the specified date; if that fails, it tries the day after, then the day before, then two days after, two days before, and so forth until a match occurs.

Wildcard day names are also provided. The keyword “weekday” applies to any days which are normally printed in "logical black" - the predominant day color - on the calendar. The keyword “workday” is the same, but does not include any holidays. The keyword “holiday” includes only those days flagged as holidays. The keywords “nonweekday”, “nonworkday”, and “nonholiday” are also recognized as negations of the above. See the **CAVEATS** below for important notes on using these keywords. Moon phases may also appear as wildcards; “nm” is accepted as a synonym for “new_moon”, “1q” and “fq” for “first_quarter”, “fm” for “full_moon”, “3q” for “third_quarter”, and “lq” for “last_quarter”.

Ordinal day numbers may be used to specify dates, either relative to the month or to the year. Either words or numeric abbreviations may be used for ‘‘first’’ through ‘‘fifth’’; higher numbers must be given using the numeric equivalent (e.g. 100th). Negative ordinal numbers may even be used. For example, ‘‘−2nd’’ means ‘‘next to last’’.

‘‘Odd’’ and ‘‘even’’ do not refer to the actual date; instead, ‘‘odd’’ means ‘‘alternate, starting with the first’’, and ‘‘even’’ means ‘‘alternate, starting with the second’’. Thus, ‘‘odd Fridays in March’’ refers to the first, third, and (if present) fifth Fridays in March — not to those Fridays falling on odd dates.

‘‘All’’ refers to each individual month; ‘‘year’’ refers to the year as an entity. Thus ‘‘odd Fridays in all’’ refers to the first, third, and fifth Friday of each month, while ‘‘odd Fridays in year’’ refers to the first Friday of January and every other Friday thereafter.

‘‘Nearest’’, ‘‘nearest_before’’, and ‘‘nearest_after’’ refer to the nearest weekday or wildcard day with respect to the specified date. ‘‘Nearest_before’’ and ‘‘nearest_after’’ allow the user to specify how *pcal* is to disambiguate between two dates that are equally near: e.g., ‘‘nonweekday nearest_before [Wed.] 9/25/96’’ refers to Sunday, 9/22 while ‘‘nonweekday nearest_after 9/25/96’’ refers to Saturday, 9/28. (Note that ‘‘nearest_before’’ and ‘‘nearest_after’’ are equivalent to ‘‘nearest’’ when no such ambiguity exists: e.g., ‘‘nonweekday nearest_before [Thu.] 9/26/96’’ refers to Saturday, 9/28.)

Text in the date file may use C-like escape sequences (i.e. a ‘\’ followed by a character, 1 – 3 octal digits, or ‘x’ followed by 1 – 2 hexadecimal digits). Escaped whitespace (including **newline**) and the standard ANSI character escapes (‘\a’, ‘\b’, ‘\f’, ‘\n’, ‘\r’, ‘\t’, ‘\v’) are all replaced by a single blank.

The HTML special characters ‘<’ ‘>’ ‘"’ ‘&’ ‘ ’ and ‘&#NNN;’ (NNN = any three decimal digits) are also supported. These will be propagated intact (be sure to escape the ‘#’ in ‘&#NNN;’) if the output is specified as HTML (see the **−H** flag); otherwise they will be converted to their ASCII equivalents. This allows a common date file to be used regardless of whether the desired output format is HTML, PostScript, or Un*x *calendar(1)* (see the **−c** flag) input.

Lines in the configuration file consisting of **year ####** (where **####** is a numeric year) can be used to set the year for following entries. This assumes that the following entries do not contain a year; any date entries containing year information will set the remembered year to that year.

Lines in the configuration file consisting of **year all** (or, alternatively, **year ***) direct *pcal* to wildcard following entries against every applicable year. This assumes that the following entries do not contain a year; any date entries containing year information (or an explicit **year ####** entry) will set the remembered year to that year.

Lines in the configuration file consisting of **opt <options>** can be used to override the defaults for any command-line options except **−c**, **−e**, **−f**, **−h**, **−H**, **−u**, **−v**, **−D**, and **−U**. Any options specified in this manner are, in turn, overridden by those specified explicitly on the command line.

Lines in the configuration file consisting of **note{/<number>} <month>** can be used to place notes regarding the entire month in one of the unused blocks of the calendar. The **<month>** indicator may be either a number 1 through 12 or an alphabetic month name as described above; ‘‘note all’’ will place the associated text in the notes block for each month in the current year. **<number>** is an optional positive or negative number specifying the empty box where the associated text is to be placed. If positive, *pcal* counts forward from the first empty box; if negative, *pcal* counts backward from the last empty box. Thus, **‘‘note/1’’** places the associated text in the first empty box; **note/-3** in the third-to-last. The default is -1 if no <number> is given (last empty box, immediately preceding the small calendars on the bottom row; cf. **−S**, **−k**, and **−K**, below). You can place several notes in the same box. You can also use more than 1 box for the various monthly notes.

Lines in the configuration file consisting of **input-language XX** (where **XX** is the 2-letter specification for any of the supported languages) can be used to set the language used for interpretation of the month names and day-of-week names for the remaining event entries. This option may be specified more than once, as needed, if the language used to describe events changes within the file. For backwards compatibility, the default value for ‘input language’ if this directive is never used is ’en’ (English). Note that this directive is distinct from the specification of ’output language’ as accomplished with the **−a** option.

Comments are supported in the configuration file. Any characters following a '#' character are ignored, through the end of that line, unless the '#' character is escaped by '\'.

Deleting Events

By prepending the *'delete'* keyword to an event specification, one or more events may be deleted from a set of previously-specified events.

For example, the following lines might appear in the date file:

```
all Friday in all  Poker game
delete first Friday in all  Poker game
```

This results in an event labeled 'Poker game' on every Friday except the first Friday of the month. If you delete an entry which is marked as a holiday, the 'holiday' flag for that day will be recalculated. Any 'delete' entries which don't match any pre-existing entries are silently ignored.

Format Specifiers

Pcal allows format specifiers in both the event text and footer strings (see the **–L**, **–C**, **–R**, and **–N** options below). Each format specifier will be replaced by a corresponding string as outlined in the following table:

%a	abbreviated weekday
%A	full weekday
%b	abbreviated month name
%B	full month name
%d	day of month (1-31)
%j	day of year (1-366)
%l	days left in year (0-365)
%m	month (1-12)
%U	week number (0-53)
%W	week number (0-53)
%u	week number (1-54)
%w	week number (1-54)
%y	year w/o century (00-99)
%Y	year w/century
%%	'%' character
%o	print number as ordinal
%0	print number with leading zeroes
%+	use following month or year
%–	use previous month or year
%{+N}[DWMY]	adjust date by +N days/weeks/months/years
%{–N}[DWMY]	adjust date by –N days/weeks/months/years

Most of these are derived from the ANSI C `strftime()` function, but the **%[louwMD]** and **%[o0+–]** format specifiers are specific to *pcal*.

The **%u** specifier considers the week containing 1/1 (Jan 1st) as week 1 and the following logical Sunday (the first day of the week as printed; cf. the **–F** option below) as the start of week 2; **%U** considers the first logical Sunday as the first day of week 1. **%w** and **%W** behave like **%u** and **%U** respectively, but use the first logical Monday instead. Note that **%w** has a different meaning from `strftime()`.

The **%o** format specifier prints a number as an ordinal, with the appropriate suffix ("st", "nd", "rd", or "th" in English) appended. For example, **%od** prints the day of the month as "1st", "2nd", "3rd", etc.

Unlike `strftime()`, *pcal* defaults to printing numbers (except **%y**) without leading zeroes. If leading zeroes are desired, the '0' prefix may be used. For example, **%0j** prints the first day of year as "001".

The **%+** and **%−** format specifiers direct *pcal* to substitute the following/previous month/year in the following **[bBmyY]** specifier. For example, **%+B** prints the name of the next month.

The **%{[+−]N}[DWMY]** format specifiers do not print anything, but instead adjust the working date by ± **N**days (**D**), weeks (**W**), months (**M**), or years (**Y**). Subsequent format specifiers use the adjusted date instead of the current date. For example, **%+1M %B %Y** adjusts the date forward by one month and then prints the resulting month and year ("January 1992" in December, 1991); **%−2W %b %d** adjusts the date backward by two weeks and prints the resulting month and day ("Jul 26" on August 9).

Such date adjustments are normally cumulative; for example, **%+1Y%−1D** adjusts the date forward by one year and then backward by one day. If **%D** or **%M** is specified alone (or if **N** is zero), *pcal* restores the original date. Note that **%M** has a different meaning to the `strftime()` function.

Here's a common, useful example of an event entry for the *pcal* date file which combines the ability to adjust working dates and the ability to display ordinals. This particular example is used to display text on the birthday of a person born in 1991:

May 10 Eric's %-1991Y%oY Birthday

That entry would result in the following text being displayed on May 10, 2005:

Eric's 14th Birthday

Encapsulated PostScript (EPS) Images

For monthly PostScript calendars only, *pcal* supports the embedding of one or more EPS images (photos, icons, etc) into any given day of the month. (EPS image specifications in the *pcal* date file are ignored for yearly PostScript calendars and for all HTML calendars.)

In order to associate an image with a given event, you must add one or more entries to the date file. The event date is specified exactly as described previously for simple event text specification lines. However, instead of specifying the text associated with the event, you instead specify the EPS image filename and some additional parameters in the following format:

```
image:<EPS-image-filename> <x-scale> <y-scale> <x-delta> <y-delta>
```

Where:

<EPS-image-filename> is the filename (which can include a path) of the Encapsulated PostScript image. **Note:** The EPS image filename must be preceded by the constant text 'image:' in order to distinguish an EPS image specification from an ordinary event text specification.

<x-scale> is a scaling factor in the horizontal dimension for the EPS image. A value of 1.0 is nominal (i.e. no change to image scale). Values between 0.0 and 1.0 shrink the image in the horizontal dimension while values over 1.0 expand the image in the horizontal dimension. Generally speaking, only positive values should be used. However, in the rare case that you find that your EPS image needs to be flipped about the vertical axis (i.e. left to right), you can use a negative value to achieve this without having to tweak the actual PostScript content within the EPS image file. Use of a negative value will undoubtedly necessitate a corresponding change to the <x-delta> parameter to account for the image's relocated position that occurs when it gets flipped "left-to-right".

<y-scale> is a scaling factor in the vertical dimension for the EPS image. Values between 0.0 and 1.0 shrink the image in the vertical dimension while values over 1.0 expand the image in the vertical dimension. Note that a negative value for this parameter can be useful in the less-than-rare case that you find that your EPS image needs to be flipped about the horizontal axis (i.e. top to bottom). In such cases, you can use a negative <y-

scale> value to achieve this without having to tweak the actual PostScript content within the EPS image file. Use of a negative value will undoubtedly necessitate a corresponding change to the <y-delta> parameter to account for the image's relocated position that occurs when it gets flipped "upside down".

<x-delta> := a horizontal adjustment in typographic 'points' (i.e. 72nds of an inch) for the positioning of the EPS image. With offsets of 0 for X and Y, the image will be printed at the extreme left edge of the box for that day, just under the numerics for that day. Positive values move the image to the right and negative values move the image to the left.

<y-delta> := a vertical adjustment in typographic 'points' (i.e. 72nds of an inch) for the positioning of the EPS image. With offsets of 0 for X and Y, the image will be printed at the extreme left edge of the box for that day, just under the numerics for that day. Positive values move the image up and negative values move the image down.

Here's an example of a line from the date file that associates an EPS image with an event:

```
4th Thu in Nov*   Thanksgiving
4th Thu in Nov*   image:/eps-path/turkey.eps 1.0 1.0 0 0
```

You can place as many images as you want on a single day of the month by specifying repeated lines in the date file. For example, these lines put icons of George Washington and Abraham Lincoln on the day of the U.S. ''Presidents' Day'' holiday, along with the event text:

```
3rd Monday in Feb*   Presidents' Day
3rd Monday in Feb*   image:/eps-path/washington.eps 0.08 0.08 8 0
3rd Monday in Feb*   image:/eps-path/lincoln.eps 0.22 0.22 48 0
```

Note that the icon for Lincoln is shifted to the right by 48 typographic points so as not to overlay the first icon.

The *pcal* releases come with a single EPS sample file ('eps/recycle.eps') of the ubiquitous 'recycle' icon (3 green arrows in a triangular shape). Such an image might be used with configuration file settings like this:

```
second Sat in all RECYCLE!
second Sat in all image:/eps-path/recycle.eps 0.039 0.039 34 -9
```

In cases where you're displaying non-holiday event text (e.g. someone's birthday) and an EPS image, you'll often need to use a negative 'Y-delta' value on the EPS image specification line, in order to shift the image down so that it doesn't cover the event text, which appears just below the day's numerics for non-holiday events. (Text for holiday events appears higher up, to the right of the day's numerics, so there's usually no collision with the EPS image.)

Note: Unfortunately, most EPS images cannot be used directly by *pcal*.

Depending on the EPS image used and how it was created, you may have to remove or comment out some or all of the PostScript 'translate' commands, in order to avoid the use of illogical X-delta and Y-delta values when specifying the EPS image in your *pcal* date file. Most programs that generate EPS output (either directly or via conversion from some other graphic format) seem to have these 'translate' commands relatively early in the EPS file.

It may take some experimentation to get it just right. Preview the *pcal* output using a PostScript viewer as you tweak the PostScript commands in the EPS image file and/or the event entry in the *pcal* date file.

> **Note:** Depending upon what application you use to preview PostScript content, the monthly calendars may not show any embedded EPS images. Here's a rundown of some popular PostScript-viewing applications and whether they correctly display the embedded EPS images:

- gv (version 3.5.8) -- EPS images appear fine
- ggv (versions 2.4.0.1 and 2.6.1) -- EPS images appear fine
- older kghostview (versions 0.13.2 [KDE 3.1.4] and 0.2.0 [KDE 3.2.3 and 3.3.2]) -- EPS images DO NOT APPEAR!
- newer kghostview (version 0.2.0 [KDE 3.4.2 and 3.5.4]) -- EPS images appear fine

For converting non-EPS images (e.g. photos) to EPS format, one can use the graphical GNU Image Manipulation Program, a.k.a. 'The GIMP':

http://www.gimp.org

For icons/images in WMF format (which are popular in various 3rd-party, legacy-OS, commercial calendar programs), the 'libwmf'/'wmf2eps' library/utility is useful for generating *pcal*-capable EPS images. It can be found at this site:

http://wvware.sourceforge.net/libwmf.html

For icons/images in SVG format, the ImageMagick 'convert' utility is sometimes useful for generating *pcal*-capable EPS images. This suite of utilities (which includes other useful ImageMagick utilities like 'display' and 'identify') may already be available on your Linux distribution. If not, it can be found at this site:

http://www.imagemagick.org

For cases where ImageMagick's 'convert' utility fails to properly convert SVG-format images to EPS format, you can try the method of converting the SVG image into an intermediate format (e.g. PNG) using the 'rsvg' utility. This utility may already be available on your Linux distribution. If not, it can be found at this site:

http://librsvg.sourceforge.net/

From the PNG format, the image can often then be successfully converted to EPS format, using the above-mentioned ImageMagick 'convert' utility.

The *Open Clip Art Library* is a good source of freely-usable images (many of which are in SVG format) for decorating your events:

http://www.openclipart.org

Note: The EPS image content is not generated in the PostScript output -- only a reference to the EPS image filename is generated. From a practical standpoint, this means that normally you'll need to print/preview the PostScript output of *pcal* from the same computer/setup as that which was used to run *pcal* in the first place. If you want to generate a calendar with embedded EPS images that will later be printed/viewed on another machine which does not have access to those EPS images, you'll need to run the output through a pre-processor which will put the EPS image content into the PostScript output file. For example, assuming your initial calendar output was generated to a file named 'pcal.ps', on most GNU/Linux systems you could run this command, which uses the popular 'Ghostscript' interpreter:

```
gs -r300x300 -dBATCH -dNOPAUSE -sDEVICE=pswrite -sOutputFile=out.ps pcal.ps
```

This would generate a PostScript file named 'out.ps', at 300x300 dpi resolution, which has the actual EPS image content embedded within, allowing you to transport the 'out.ps' file to another computer for viewing/printing. Of course, the new file is substantially larger, but it's portable. Furthermore, the EPS images will be viewable even in PostScript-viewing applications (see above) which don't properly support the display of embedded (by filename only) EPS images.

Pre-Processor Functionality

Pcal supports rudimentary *cpp*-like functionality in the date file, allowing the following constructs:

- **define | undef**
- **if{{n}def} ... {elif ...}* {else ...} endif**
- **include**

Note that these are not preceded by '#' as they are in C.

Symbol names defined using these keywords (or via the **-D** option) are case-insensitive. It is not an error to **undef** an undefined symbol, nor to **define** a previously-defined one.

A symbol can be defined with just a name (e.g. "define MY_SYM") or it can take on a value (e.g. "define MY_SYM SOME_VALUE"). Use of symbol values is convenient for defining a starting date then using that symbol to reference that starting date in one or more events. For example, these definitions in the date file might be useful:

```
define semester_start 8/23   # Beginning of semester
semester_start                 Class Start
7th  day after semester_start  1st Quiz
14th day after semester_start  2nd Quiz
undef semester_start
```

Be aware that the substitution of symbol values for symbol names is not robust, so it's wise to use a symbol name that's unlikely to occur in any of your other event text. In other words, if you defined the 'semester_start' symbol in the example above as merely 'start', then you'd get the undesired effect of having the text 'Class 8/23' in your calendar on that day instead of 'Class Start'! The use of 'undef semester_start' in the above example is optional and is really only useful to prevent any unwanted symbol substitutions later on, which probably won't happen unless you poorly choose your symbol name to begin with.

An **ifdef** alone is always **false**; an **ifndef** alone is always **true**. **if** is accepted as a synonym for **ifdef**.

The name of the file in the **include** directive may optionally be surrounded by either "" or <>, both of which are ignored. If the name is not an absolute path, it is taken to be relative to the directory where the file containing the directive is located. If the string "%y" appears in the file name, it is replaced by the last two digits of the current year or, if "year all" is in effect, is expanded to all applicable years. *Pcal* is smart enough to translate **~/** to the user's home directory.

Pcal normally terminates immediately if the file specified in an **include** directive does not exist. An alternate form of the directive, **include?**, directs *pcal* to continue silently if the file does not exist or cannot be opened.

In addition to pre-processing keywords, *pcal* also accepts boolean expressions in **if{{n}def}** and **elif** directives. These expressions consist of symbol names joined by the boolean operators **!**, **&**, **^**, and **|**, in order of precedence, high to low. Parentheses may be used to alter the precedence. The synonyms **&&** and **||** are accepted for **&** and **|**. A symbol name evaluates to **true** if currently defined, **false** if not; thus:

```
ifdef A | B | C
```

...is **true** if any of the symbols A, B, and C is defined, and:

```
ifdef A & B & C
```

...is **true** if they all are. Note that **ifndef <expr>** is equivalent to **ifdef !(<expr>).**

The Moon File

If a file of the name *.moon##* (*moon##.dat* under MS-DOS), where **##** is the last two digits of the calendar year, exists in the same directory as the date file (or in the directory where *pcal* resides), *pcal* uses the information contained within to calculate the phase of the moon. If a) no such file exists, b) the **–e** flag (do not use a date file) is specified, or c) the **–z** flag (specify time zone) is specified, then *pcal* uses an algorithm to calculate the phase of the moon.

Entries in the moon file must conform to the following syntax:

If the **−A** option (American date formats, the default) is given:

```
<quarter> <month><sep><day> {<hour><sep><min>}
```

If the **−E** option (European date formats) is given:

```
<quarter> <day><sep><month> {<hour><sep><min>}
```

Where:

<quarter>	:= “nm”, “fq” or “1q”, “fm”, “3q” or “lq” (new moon, first quarter, full moon, last quarter)
<hour>	:= number 0-23 (24-hour clock)
<min>	:= number 0-59

This file must contain entries for all quarter moons in the year, in chronological order; if any errors are encountered, *pcal* will revert to using its default algorithm.

As in the date file, comments start with ‘#’ and run through the end of the given line.

The moon file may optionally contain an **opt −A** or **opt −E** line to specify the format of its own date entries independently of the format used in the date file. No other flags are legal in the moon file.

Generating PostScript Calendars Via A Web Browser Interface

PostScript-format *pcal* calendars can be generated and viewed from a web browser interface.

> **Note:** This is not to be confused with the ability to generate non-PostScript, HTML-format (using the **−H** command-line option) calendars, which is a different capability entirely.

Pcal comes with 4 files that provide this ability: ‘pcal.cgi’ (a Bourne shell script), ‘pcal.pl’ (a Perl equivalent of ‘pcal.cgi’), ‘pcal.html’, and ‘pcalw.html’.

The CGI file (either ‘pcal.cgi’ or ‘pcal.pl’) must be edited before using it. Change the definition for *‘pcal=’* (Bourne shell script) or *‘my $PCAL =’* (Perl script) to point to the location of the *pcal* executable file. Change the definition for *‘file=’* (Bourne shell script) or *‘my $FILE =’* (Perl script) to point to the location of the *pcal* ‘date file’ (e.g. ‘.calendar’), which contains the options for running *pcal*. Finally, copy the ‘pcal.cgi’ (or ‘pcal.pl’) file to the location where your web server expects to find such files (e.g. ‘/var/www/cgi-bin/’).

The ‘pcal.html’ and ‘pcalw.html’ files must also be edited. Each one has a line like this:

> <FORM ACTION="http://yourpath/cgi-bin/pcal.cgi" METHOD=GET>

That line must be edited to point to the host and location of your CGI script file (‘pcal.cgi’ or ‘pcal.pl’).

Once that’s done, point your web browser to the ‘pcal.html’ or ‘pcalw.html’ file to generate monthly/yearly PostScript calendars for viewing within your web browser.

> **Note:** Depending upon what application your web browser spawns to preview PostScript content, the monthly calendars generated via this web browser interface may not show any embedded EPS images. For a rundown of some popular PostScript-viewing applications and whether they correctly display the embedded EPS images, see the section (above) entitled ‘Encapsulated PostScript (EPS) Images’.

OPTIONS

−e Prints an empty calendar. Do not print entries from a *.calendar* file even if one exists.

−f *cal* Directs *pcal* to use the file name *cal* as the input file in place of the default *.calendar* file. Note that the search rules are different when **−f** is used. If *cal* is an absolute file name (i.e. starting with a ‘/’), then *pcal* attempts to open only that file. Otherwise, *pcal* looks for *cal* in the current directory, then in the directory indicated by the environment variable **PCAL_DIR** (if defined), and finally, if enabled (via the ‘SEARCH_PCAL_DIR’ flag) when *pcal* was built, in the directory where the *pcal* executable resides. If the given *cal* file is not found, an error results.

−o *file* Directs *pcal* to write the output to *file* instead of to stdout.

−l Causes the output to be in landscape mode (default).

Pcal predefines the symbol 'ORIENTATION_LANDSCAPE' whenever 'landscape' page orientation is enabled. This can be useful for providing alternate values in the configuration file for EPS image placement and scaling, based on the page orientation.

−p Causes the output to be in portrait mode.

Pcal predefines the symbol 'ORIENTATION_PORTRAIT' whenever 'portrait' page orientation is enabled. This can be useful for providing alternate values in the configuration file for EPS image placement and scaling, based on the page orientation.

−P Selects the paper size. The following sizes are supported:

- letter — 8.5 x 11.0 inches
- legal — 8.5 x 14.0 inches
- a4 — 210 x 297 mm
- tabloid — 11.0 x 17.0 inches

Pcal predefines one of the following symbols based on the current paper size:

- PAPERSIZE_LETTER
- PAPERSIZE_LEGAL
- PAPERSIZE_A4
- PAPERSIZE_TABLOID

These symbol definitions can be useful for providing alternate values in the configuration file for EPS image placement and scaling, based on paper size.

−j Causes the Julian date (day of year) to be printed in each calendar box.

−J Causes the Julian date and the number of days remaining in the year to be printed in each calendar box.

−m Causes moon icons to be printed on dates corresponding to new, half, and full moons (the default is that no moons are printed).

−M Causes moon icons to be printed on all dates (the default is that no moons are printed).

−g *day1*[−*day2*] **| all | holiday**
Causes all dates falling on weekday *day1* (through *day2* if specified) to be printed in the 'day numerics color' (i.e. the color specified by the **−s** option [default = gray]); **−g all** causes all weekdays (other than holidays) to be printed in the 'day numerics color'; **−g holiday** causes all holidays to be printed in 'day numerics color'. *day1* and *day2* may wrap around weekends; for example, **−g fri−sun** causes Fridays, Saturdays, and Sundays to be printed in the 'day numerics color'.

−O *day1*[−*day2*] **| all | holiday**
Similar to **−g**, but the selected days will be printed as outlined characters, using the 'day numerics color'.

−G *day1*[−*day2*] **| all | holiday**
Similar to **−g**, but the selected days will be printed in the 'day numerics color', outlined in black.

−b *day1*[−*day2*] **| all | holiday**
Similar to — you guessed it — **−g**, but the selected days will be printed in black. Since black is the default for weekdays, −b is primarily used to overriding other flags (e.g., **−g all −b sat-sun**).

Note:
The default for the above options is to print Saturdays, Sundays, and holidays in the ‘day numerics color’ and all other days in black. For backward compatibility with earlier versions of *pcal*, **–O** and **–G** alone change all non-black days to the specified color.

–s *[day_numerics_color][/empty_day_box_fill_color]*
Overrides the default value(s) for the color of the numerics for each day and/or the color of the fill used on boxes for ’empty’ days. **Note:** This option only applies to PostScript-format calendars, not to HTML-format calendars. These values may be set independently of each other. For use with non-color printers, these values should be in the range 0.0 (black) through 1.0 (white). The default values are 0.8 for day numerics and 0.9 for empty day boxes. For use with color printers, these values may optionally be specified as a set of *red*:*green*:*blue* (RGB) values, each of which must in the range 0.0 through 1.0. At least one ‘:’ must be present for these values to be recognized as RGB colors; omitted values are set to 0.0.

This option may also be set semi-permanently by altering the makefile (‘Makefile’ for most environments, ’Makefile.DOS’ for MS-DOS).

–F *day* Selects weekday *day* as the first day of the week. The given day will appear in the left-most column of the calendar. *day* may be specified either as a weekday name or, optionally, as a number in the range 0 (Sunday) through 6 (Saturday).

This option may also be set semi-permanently by altering the makefile (‘Makefile’ for most environments, ’Makefile.DOS’ for MS-DOS).

–A Directs *pcal* to use American date conventions (**mm/dd{/yy}** and **month dd**) when parsing the date file. This is the default.

This option may also be set semi-permanently by altering the makefile (‘Makefile’ for most environments, ’Makefile.DOS’ for MS-DOS).

–E Directs *pcal* to use European date conventions (**dd/mm{/yy}** and **dd month**) when parsing the date file.

This option may also be set semi-permanently by altering the makefile (‘Makefile’ for most environments, ’Makefile.DOS’ for MS-DOS).

–X *xtrans*
Specifies the x-axis translation value for positioning the output on the page. Positive values shift the output to the right. Negative values shift the output to the left.

–Y *ytrans*
Specifies the y-axis translation value for positioning the output on the page. Positive values shift the output up. Negative values shift the output down.

–x *xscale*
Specifies the x-axis scaling factor for the calendar size.

–y *yscale*
Specifies the y-axis scaling factor for the calendar size.

–t [*title_font*][**/***size*]
Specifies the name of a font to use for all the calendar heading text:

- the month name and year at the top of the calendar (for monthly-format calendars) or at the top of each month (for yearly-format calendars and for the small previous/next-month calendars [if enabled] on monthly-format calendars)
- the day-of-week names
- the footer strings (if any)
- the “Notes” box heading (if any; for monthly calendars only)

For monthly calendars only, the user may optionally specify the font size, which applies only to the main month/year heading. For example, **pcal –t Times-Roman/54** sets the font to Times-Roman and the month/year point size to 54. The font size may also be changed independently: **pcal –t /54** changes the point size to 54 without affecting the font name.

Note: For yearly calendars, any specification of font *size* is ignored.

This option may also be set semi-permanently by altering the makefile (‘Makefile’ for most environments, ’Makefile.DOS’ for MS-DOS).

–d [*date_font*][**/***size*]

Similar to the **–t** option, but selects the font and/or size used for the day numerics (the numbers inside the box for each day).

Note: For yearly calendars, any specification of font *size* is ignored.

This option may also be set semi-permanently by altering the makefile (‘Makefile’ for most environments, ’Makefile.DOS’ for MS-DOS).

–n [*text_font*][**/***size*]

Similar to the **–t** and **–d** options, but selects the font and/or size used for any ‘event’ text associated with each day and for any text in the monthly ‘‘Notes’’ box.

Note: This option applies to monthly calendars only. For yearly calendars, this option does not apply.

This option may also be set semi-permanently by altering the makefile (‘Makefile’ for most environments, ’Makefile.DOS’ for MS-DOS).

–L *string*

Causes the accompanying string to be printed as a left-justified footer. Format specifiers denoting the month and/or year may appear in the string; the appropriate values will be substituted upon printing.

–C *string*

Similar to **–L**, but causes the accompanying string to be printed as a centered footer. If the **–H** flag (generate calendar as HTML table) was specified, this string will be used as the title and heading.

–R *string*

Similar to **–L**, but causes the accompanying string to be printed as a right-justified footer.

–N *string*

Causes the accompanying string to be printed as the heading for the "Notes" box. Note, however, that *pcal* makes no attempt to ensure that it fits.

–D *symbol [value]*

Defines the named symbol and an optional value to be associated with that symbol, prior to reading the date file.

–U *symbol*

Un-defines the named symbol prior to reading the date file.

–B Causes *pcal* to leave unused calendar day boxes blank as opposed to the default behavior of filling them using the ‘empty day-box fill color’ (i.e. the color specified by the **–s** option [default = gray]).

–# *n* Causes *pcal* to print *n* copies (maximum: 100) of each output page.

–S Causes *pcal* to suppress printing the small calendars. See the **CAVEATS** section for further details.

–k Causes *pcal* to print the small calendars in the upper left corner (the default is to print them at the lower right).

−K Causes *pcal* to print the small calendar for the previous month in the upper left corner and the next month in the lower right (the default is to print both at the lower right).

−w Causes *pcal* to print a calendar for 12 consecutive months: 3 rows / 4 columns in landscape mode, 4 rows / 3 columns in portrait mode. See the **CAVEATS** section for details on the use of this option with other options.

Pcal predefines the symbol **whole_year** when the **−w** flag is in effect, allowing directives like '**ifdef whole_year**' in the configuration file.

−I Resets all parameters to the program defaults.

−c Causes *pcal* to generate a date file suitable for use as input to the Un*x *calendar(1)* utility. The normal PostScript output is suppressed.

−H Causes *pcal* to generate a calendar in HTML table format. The normal PostScript output is suppressed.

The HTML table format does not support moon graphics, Julian date information, 'day numerics' color, 'empty day' 'box fill' color, left or right footer strings (but see the **−C** flag), alternate fonts/sizes, transformation and scaling factors, or embedded EPS images.

−q This option is only valid when used in conjunction with the **−H** (generate HTML-format calendar) option. It generates a yearly-planner style of HTML calendar whereby a single column for each month is used, resulting in table that gives a quicker overview of several months. Since there is less space for text, only the first character of the weekday and the first 5 characters of text from each event for that day are printed. The day numerics for holidays are colored red but the text of the holiday event is not printed. The day numerics are grey for Saturdays and bold black for Sundays.

−z *time_zone*

Forces *pcal* to ignore the moon file and to use its internal algorithm for moon phase calculations, adjusting the phase by *time_zone* hours (where *time_zone* is expressed in hours west of UTC).

For example, New York residents (USA Eastern time zone) would use '-z 5' while on Eastern Standard Time (winter) and '-z 4' while on Eastern Daylight Time (summer). People in India would use '-z-5.5'. Notice that fractional values are allowed.

This option may also be set semi-permanently by altering the makefile ('Makefile' for most environments, 'Makefile.DOS' for MS-DOS).

−h Causes *pcal* to write version information, parameter usage message, and full explanation of options and file formats (to *stdout*) and terminate.

−u Causes *pcal* to write version information and parameter usage message (to *stdout*) and terminate.

−v Causes *pcal* to write version information only (to *stdout*) and terminate.

Pcal predefines the symbol **vX_Y_Z**, where **X_Y_Z** denotes the current version of Pcal (e.g. version 4.9.0 predefines the symbol **v4_9_0**).

−a *output_language*

Select the output language (for the names of months and days on the calendar).

Currently, the following languages are supported: **ca** (Catalan), **cs** (Czech), **da** (Danish), **de** (German), **el** (Greek), **en** (English), **eo** (Esperanto), **es** (Spanish), **et** (Estonian), **fi** (Finnish), **fr** (French), **ha** (Hawaiian), **hu** (Hungarian), **it** (Italian), **lt** (Lithuanian), **lv** (Latvian), **nl** (Dutch), **pl** (Polish), **pt** (Portuguese), **ro** (Romanian), **ru** (Russian), **sk** (Slovak), **sv** (Swedish), and **uk** (Ukrainian). The default is **en**.

Note that this option does **not** specify the 'input language', which is the language used to process events in the configuration file ('English', by default, unless changed with the *input-language XX* directive). See the section *The Date File* for more details on specifying the 'input language'.

Note: In order to display diacritical marks, languages other than English require that the characters be remapped. Normally, no action is required since *pcal* automatically selects a remapping which is appropriate to the selected language. However, if you want to override the default remapping for a given language, you would use the **−r** option (e.g. "-r Latin1").

Furthermore, an appropriate font should be selected as needed using the **−t** option (e.g. "-t some-latin1-font-name"). Any language using the "Latin1" remapping (e.g. French, German, Italian, Spanish, etc) requires an ISO 8859-1 ('Latin1') font. The Greek language requires an ISO 8859-7 (similar to ELOT-928) font (available from Angelo Haritsis <ah@doc.ic.ac.uk>; also see http://www.hellenic.net/fonts/). Similarly, Russian requires a KOI8-R font while Ukrainian requires a KOI8-U font.

Pcal predefines the symbol **lang_XX**, where **XX** is the two-character abbreviation for the selected output language.

−r [*mapping*]

Specifies an 8-bit character set remapping (encoding) for printing the diacritical marks common to European languages.

Note: This option is not usually needed since *pcal* will automatically select an appropriate default character encoding (mapping) for the language for which the calendar is being generated.

The value specified for *mapping* is case-insensitive and may be abbreviated to the point where it is still unique. The value used may be any of the following:

- "none" (use built-in character set)
- "Latin1" (ISO 8859-1)
- "Latin2" (ISO 8859-2)
- "Latin3" (ISO 8859-3)
- "Latin4" (ISO 8859-4)
- "Cyrillic" (ISO 8859-5)
- "Greek" (ISO 8859-7)
- "Latin5" (ISO 8859-9)
- "Latin6" (ISO 8859-10)
- "Thai" (ISO 8859-11)
- "Latin7" (ISO 8859-13)
- "Latin8" (ISO 8859-14)
- "Latin9" (ISO 8859-15)
- "KOI8-R" (Russian)
- "KOI8-U" (Ukrainian)
- "Roman8"

This option may also be set semi-permanently by altering the makefile ('Makefile' for most environments, 'Makefile.DOS' for MS-DOS).

−T [B|I|R]

Select the default typeface (Bold, Italic, or Roman) for printing date/note text. This flag may be specified multiple times within the date file (via "opt") to reset the font style on the fly — for example, to print all holidays in Bold.

−W [left|center|right]
Specify the horizontal alignment of the month/year heading (left, center, right) (for monthly-format calendars only).

Any option taking a negative value (e.g. **−Y −###**) should be specified with no space between the option and the (negative) value to avoid *pcal* interpreting the value as an illegal flag and aborting. For example, use ‘ *−Y−50*’ instead of ‘ *−Y −50*’ on your option specification.

Any option (except **−G** and **−O**, for backward-compatibility) which normally takes an argument may be specified without the argument in order to reset the value to the program default. Note that while the **−D** option alone clears all the defined symbols, the **−U** option alone has no effect. The **−** (or **− −** as per System V) argument may be used to disambiguate command lines such as:

pcal −t 9 90

This could be written instead as one of the following:

pcal −t − 9 90
pcal −t − − 9 90

If the environment variable **PCAL_OPTS** is defined, its contents are parsed as a command line. Flags set via **PCAL_OPTS** override the program defaults, but are overridden by options set via **opt** lines in the configuration file or explicitly on the command line.

Additional Options For Debugging Only

The **−Z** flag is used to print debugging information which is of interest primarily to *pcal* hackers. This flag is a "hidden" flag; it does not appear as part of the usage message. At present, the following options are supported:

- -ZD print dates and text as read from date file
- -ZF print date file search paths
- -ZM print moon phases and identify quarters
- -ZO print option flags and where set
- -ZP print "preprocessor" debug info
- -ZT print dates and text as written to output file
- -Z turn off all debugging info

The subflags may be combined: e.g. "-ZDF" is equivalent to "-ZD -ZF". All of the aforementioned debugging information is written to stderr.

CAVEATS

- The ‘‘workday’’ and ‘‘holiday’’ keywords are aware of only those holidays which have already been flagged at the point where they appear. For example, consider January 1990:

```
      January 1990
  S  M Tu  W Th  F  S
     1  2  3  4  5  6
  7  8  9 10 11 12 13
 14 15 16 17 18 19 20
 21 22 23 24 25 26 27
 28 29 30 31
```

If the configuration file looked like this:

```
workday on_or_before all 15   payday
```

```
3rd Mon in Jan*              MLK day
```

... then *pcal* would mark the 15th as ‘‘payday’’ since at that point in the configuration file it has no way of knowing that January 15th will later be flagged as a holiday. If the two lines were reversed, such that the holiday preceded the ‘‘workday’’ wildcard, then *pcal* would work as intended, marking instead the 12th as ‘‘payday’’.

Also, beware of year boundaries which affect the handling of all of the day wildcard keywords. In general, it is best to place monthly wildcards such as the example above at the end of each year to achieve the desired effect.

- Only the positive ordinals may be used in conjunction with prepositions (e.g. "fourth Sunday before 12/25"). (It could be argued that "last Sunday before 12/25" should be accepted as a synonym for "first Sunday before 12/25", but then what does "last Sunday after 12/25" mean?)
- When the **−w** and **−p** options are used together, *pcal* revises the y-scale factor in order to use the entire portrait page; therefore, the user should avoid using use the **−y** option when using both the **−w** and **−p** options. Use of the **−w** option in any case effectively disables the **−m**, **−M**, **−j**, and **−J** options.
- The output of the **−c** option may be used as input to subsequent runs of *pcal*. Note, however, that **opt** lines (except for an automatic **opt −[A|E]**), comments, ‘‘note’’ text, and **ifdef**'d-out source will be lost.
- The **−S** option interacts with **note{/<number>}**; if used, it should be specified either on the command line or prior to the first **note** line in the date file.

SEE ALSO

Website for *pcal* and *lcal* (a lunar calendar generation application):

http://pcal.sourceforge.net

cal(1), calendar(1).

The old, simple Unix/BSD 'calendar' program, which can be used with *pcal* and the '-c' option is part of the BSD Main Utilities ('bsdmainutils') package and is available at the Debian site:

http://packages.debian.org/stable/source/bsdmainutils

This old program does not seem to be included with most GNU/Linux distributions these days.

For more information on PostScript, consult the free, online Adobe book entitled *PostScript Language Reference Manual*, which can be found here (as of Dec 2007):

http://partners.adobe.com/public/developer/ps/index_specs.html

AUTHORS

The original PostScript code to generate the calendars was written by Patrick Wood (Copyright © 1987 by Patrick Wood of Pipeline Associates, Inc.), and authorized for modification and redistribution. The calendar file inclusion code was originally written in "bs(1)" by Bill Vogel of AT&T. Patrick's original PostScript was modified and enhanced several times by King Ables, Tim Tessin, Joe Wood, Jeff Mogul, Mark Hanson, and others whose names have regrettably been lost. This C version was originally created by Ken Keirnan of Pacific Bell; additional enhancements by Joseph P. Larson, Ed Hand, Andrew Rogers, Mark Kantrowitz, Joe Brownlee, Andy Fyfe, Steve Grandi, Geoff Kuenning, Stefan Fronzek (1-column HTML output), Bill Bogstad (event deletion capability), and Bill Marr (embedded EPS images, command-line paper size specification, new paper sizes [legal & tabloid], additional character mappings for new languages, and various cleanups). The moon routines were originally written by Jef Poskanzer and Craig Leres, and were incorporated into *pcal* by Richard Dyson. International language support was initially added by Angelo Haritsis. Additional languages were added by Andrew Rogers (Esperanto), Lars Wirzenius (Finnish), Pedro Zorzenon Neto (Portuguese), Joel Fredrikson (Swedish), Volodymyr M. Lisivka (Ukrainian), Neeme Praks (Estonian, Russian, Latvian, Lithuanian), Peter Cernoch (Czech), Ferenc Kruzslicz (Hungarian), Carles Sadurni Anguita (Catalan), Dominik 'Chiron' Derlatka (Polish), Ewald Beekman (Dutch), Claudiu Costin (Romanian), Kenneth Geisshirt (Danish), Zdenko Podobny (Slovak), and Eric Nichols (Hawaiian).