6

Prolog IV's graphical environment

THIS CHAPTER briefly describes Prolog IV's graphical environment. It includes the description of the various menus, windows and dialogs available, as well as a list of the graphical primitives which may be accessed from Prolog IV.

6.1 Starting Prolog IV

Prolog IV's graphical environment is implemented with TCL/TK, a public-domain tool. TCL is a shell-oriented programming language, and TK its associated graphical library, originally designed for X11, and now available under the Windows-xx environments, as well as under MacOS. From a console-based environment, you can start Prolog IV with its graphical environment by typing the command `prolog4e` in a « shell » window. From a « point-and-click » environment, you can start Prolog IV with its graphical environment by clicking on the appropriate icon.

The environment starts with a panel with a few buttons, as well as a console window. This console should contain some text very close to the following:

```
[Connection OK]
Prolog IV v1.0 (222), Juillet 1996 (C)PrologIA 1995,1996
[ Fri Jun 14 10:45:43 MET DST 1996 ]

>>
```

If no text is displayed in this window, then you should check the following:

- the end of the installation procedure has not been performed;
- There is not enough memory to run Prolog IV on your machine:
  - maybe that you have too many processes running on the machine,
  - or Prolog IV is configured to use a very large workspace. You may reduce them in the « Preferences » dialog, under the « Prolog » theme.

6.2 Initial configuration

This is to be done at least once, usually during the installation. The Prolog IV parameters need to be configured. This is done in the perferences dialog, under the « Prolog » theme. The minimum setup consists in putting in the « Exec Name: » the path[1] to Prolog IV's main executable, and to select the buttons « Uses Tcl » and « Auto Launch ».

In order to save this configuration for future sessions, you then need to activate the « Save Preferences » item from the « Info » menu in the main panel.

You can then start the Prolog IV program by activating the « Run » item in the « Process » menu in the window named « Console Prolog IV ». You can also quit the graphical environment and start it again, which will allow you to check that the setup information has been saved correctly.

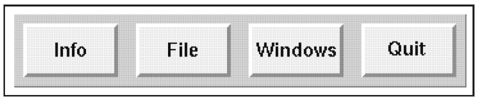
6.3 The main panel

The graphical environment includes a main pnel, some dialogs, and some editing windows called « editors. »

The main panel contains the access to all parts of the graphical environment, *i.e.* to the editors, to various informations, and to Prolog IV, which runs in a window called « Console Prolog IV » which we will call console for short.

The environment is not directly linked to Prolog IV, and it is theoretically possible to start any program in it which uses the normal input/output procedures. However, most of the environment's functionalities are to be used with Prolog.

Here is the description of the various elements of the main panel:



6.3.1 The « Info » menu

The « Info » menu contains various informations, such as the access to the « Read-me » file, to the preferences, to a bug report dialog, . . .

About Prolog IV ... : displays Prolog IV's current version.

Preferences ... : gives access to the Preferences dialog.

Save Preferences : saves the current preferences in the HOME directory of the present user.

Credits ... : some credits about parts of the software.

Bug Report ... : opens a « normalised » dialog to send descriptions of problems or suggestions about Prolog IV.

Read Me ... : displays a window which contains informations about the graphical environment which could not be included in the manual.

[1] You can use the $\sim$ substitutions in the beginning of the path.

6.3.2 The « File » menu

The « File » menu gives access to some general functionalities concerning text editors.

New : creates a new (empty) editor window.

Open ... : opens a file selection dialog box, and then opens the selected file in a new editor window.

Save All : saves all the editors which have been modified. Only asks the programmer when a window has never been saved (in order to get a file name).

Close Unedited : closes all the unedited editor windows (destroys them).

6.3.3 The « Windows » menu

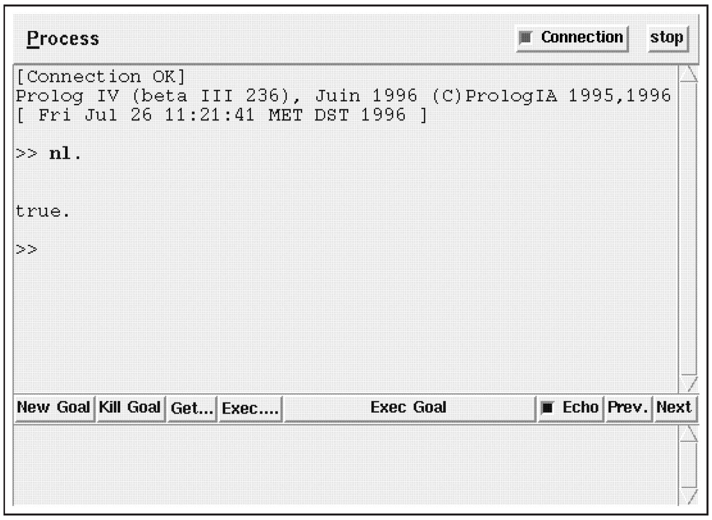
This menu shows a list of all the open editor windows (and only these windows; the console is not listed). When an item is selected, the corresponding window is displayed on front of the others (deiconifying it if necessary).

6.3.4 The « Quit » button

This button exits Prolog IV's development environment. If needed, a dialog proposes to save all the modified editor windows.

6.4 The Prolog IV console

The console is the window which allows the programmer to interact directly with Prolog IV. It contains a menu bar, an editable text panel in which appear all Prolog IV's inputs and outputs, as well as another text panel, which we will call « Queries » panel, in which the user may put frequently used queries.



The menu bar contains the following menus and buttons:

6.4.1 The « Process » menu

Run : Starts the executable specified in the « Exec. Name: » field from the « Prolog » preferences dialog, by concatenating to it the options from the

« Options: » field, then a double-dash « -- » and finally the options of the « User-Options » field. If a Prolog IV process was already running in the console, then it is killed abruptly (without confirmation).

Kill : Abruptly terminates the execution of the process running in the console. Under UNIX, this is done by a SIGKILL signal.

Preferences ... : displays the preferences dialog.

6.4.2 The « Stop » button

This button send a user interrupt to the process currently running in the console. In the case of Prolog IV, this results in the interruption of the currently running program, unless user interrupts are handled by the program. Under UNIX, this interruption is performed using the SIGINT signal.

6.4.3 The « Connection » button

When this button is checked, then Prolog IV is actively running in the console, and its standard input/output is taken from the console. It may be unchecked when no process is running, or when the process' input/output goes to the debugger window (in which a similar button is present).

Note: It is not possible to get the `stderr` output stream in the console.

6.4.4 The « queries » panel

The « Queries » panel includes a text part (with scrollbar), as well as a few buttons and menus. This panel allows the user to edit queries, evaluate them, and browse through them. The goals in the panel are organised in a stack, in which the user may freely insert and suppress goals.

New Goal : creates an empty new goal, which is inserted in the stack just after the one currently displayed.

Kill Goal : destroys the currently displayed query. If present, the next goal is shown; otherwise the preceding one is shown.

Get ... : loads in the text panel the selected goal.

Exec ... : evaluates the selected query, without loading it in the text panel.

Exec Goal : evaluates the query presently in the text panel.

Echo : When this button is checked, the evaluated queries are displayed in the console.

Prev. : loads in the text panel the previous query from the list. If there is no previous query, beeps instead.

Next : loads in the text panel the next query from the list. If there is no next query, beeps instead.

Shortcuts are defined for some buttons (pay attention to the case of the letter, since they are significant):

- `Enter` (from the numerical pad) for the « Exec Goal » button, *i.e.* to evaluate the current query,

- `CTRL-p` for the « Prev. » button,
- `CTRL-n` for the « Next » button,
- `CTRL-N` for the « New Goal » button,
- `CTRL-K` for the « Kill Goal » button.

In addition, `CTRL-D` may be used in order to duplicate the query presently in the text panel. This allows the user to « store » the current query for future use, while editing into another query.

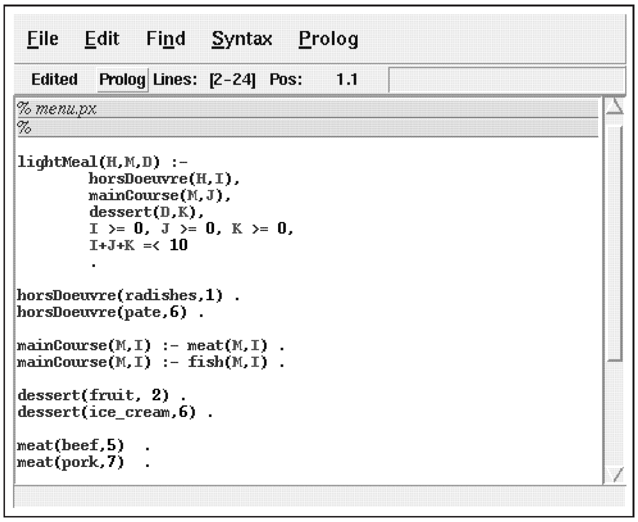
6.5 The TCL/TK console

This console may be used to run Tcl commands. After typing a command, one simply needs to select it and press the « Eval Selection » button. One may also, if the command fits on a single line, press on the `Enter` key from the numerical pad.

This console should only be used by persons who know TCL. Currently, the versions used are Tcl 7.3 and Tk 3.6, as well as TclX 7.3 .

6.6 The editors

Each editor window has its own menu-bar, as well as a status line, which is displayed immediately under the menu bar. The title of the window is the name of the file (Untitled-n if it has not yet been given a name), as well as its path.



The status line contains the following elements, from left to right:

- The file current status, which is either empty or « Edited » if the file has been modified since it was last saved.
- A menu showing which language is used in the editor (to be positioned by the user). This is only important for dynamic syntax coloring.
- The interval of the lines which are currently visible in the window, in the form *row.column*. Lines are numbered from 1, columns from 0.

- a status string, showing a possible currently executing action.

The menu bar is made of the « File », « Edit », « Find », « Syntax », « Prolog » menus, which are described below.

6.6.1 The « File » menu

New Window... : creates a new (empty) editor window.

Clear All : empties the window, after asking for confirmation if needed.

Load File ... : displays a file selection dialog box, and loads the selected file in the current editor window, replacing the current file. Asks for confirmation if needed.

Save : saves the contents of the window in a file. If the content is not yet associated with a file name, asks for one.

Save As ... : saves the contents of the window in a file, whose name is specified in a file selection box, and renames the window accordingly.

Save To ... : saves the contents of the window in a file, whose name is specified in a file selection box, without changing the name of the window.

Close : closes the window, after asking for confirmation if needed.

6.6.2 Le menu « Edit »

The « Edit » menu resembles very much those usually found on Macintoshes and NeXTs.

? Undo : Not yet implemented.

Cut : copies the current selection (if it exists) into the copy-buffer and destroys this selection.

Copy : copies the current selection (if it exists) into the copy-buffer.

Paste : pastes the content of the copy-buffer at the insertion point (without changing the content of the copy-buffer).

Paste Selection : pastes the text of the current X11 selection at the insertion point (without changing the content of the copy-buffer).

Clear : destroys the content of the copy-buffer.

Shift Left : shifts to the left (one tab) the lines of the selection.

Shift Right : shifts to the right (one tab) the lines of the selection.

Misc. : display a sub-menu in which the user may decide of the auto-indent option used or of the dynamic coloring scheme used.

Select All : selects the whole content of the window.

Font list : displays a sub-menu in which the user may select the font to be used in the window.

6.6.3 The « Find » menu

The « Find » allows the user to perform searches and replacements, while taking into consideration or ignoring the case of the letters, or by searching for regular expressions. The « Find/Replace » may be activated from this menu; it is described further in the present chapter.

Find ... : activates the find/replace dialog.

Find Next : finds and selects the next occurrence of the currently searched string.

Find <Selection> : finds and selects the next occurrence of the currently selected string.

Replace : replaces the current selection by the replace string.

Replace & Find : replaces the current selection by the replace string, and finds the next ocurrence of the currently searched string.

Replace All : replaces the currently searched string by the replace string in the whole file.

Find All : colors all occurrences of the currently searched string. This coloring is not a selection, but simply a coloring, without incidence on the edition.

Next Found : jumps to the next colored area in the window.

Unhighlight All Found : eliminates all the text coloring introduced by previous searches.

Selection – > Find-field : places the currently selected text into the currently searched string.

Selection – > Replace-field : places the currently selected text into the replace string.

6.6.4 The « Syntax » menu

Check Syntax : checks the syntax of the contents of the window. presents the erroneous lines (which contain an error or a warning) in a specific style (color). After doing so, the « Goto Next Error » option may be used to browse through them. One can note that when the mouse cursor is in front of a colored line, then the number and message corresponding to the error or warning are displayed.

Check Syntax (Selection) : same as above, buts only checks the syntax in the current selection.

Goto Next Error : jumps to the next line colored by « Check Syntax ». If wrapping over the end of file, beeps, and selects the first error in the file.

Clear All Error Marks : removes from the file all the *error* and *warning* specific styles.

Clear Warning Marks : removes from the file all the *warning* specific styles, allowing the user to concentrate on the errors.

Syntax Coloring : colors the file, using Prolog syntax. This feature is experimental, and it cannot be undone.

6.6.5 The « Prolog » menu

This menu gives access to the main rule compilation and consultation commands. This can only be used if a Prolog IV program is currently running, since its compiler is used. By default, all operations first perform a syntax checking before to perform a compilation operation.

Check Syntax before : If this item is checked (the default), then the activation of the following items will first trigger the syntax checking of the zone to be compiled. If errors are encountered during the checking, a message is displayed and the pending action is cancelled.

Debug : If this item is checked, the compilations are performed in debug mode.

Compile Window : Compiles the content of the window (not of the file).

Compile Selection : Compiles the content of the current selection.

Recompile Window : Recompiles the content of the window (not of the file).

Recompile Selection : Recompiles the content of the current selection.

Consulting : Through a sub-menu, gives access to the consultation commands (seldom used).

6.7 The debugger

This window is an interface with the Prolog IV debugger, which is described in the « Environment. » It offers the visualisation of the source code which has been compiled using the compiler's `debug` option.

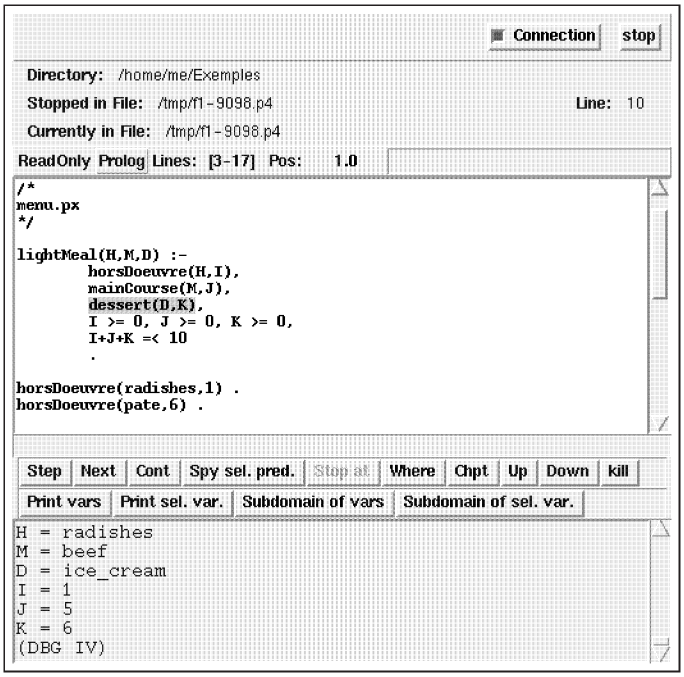
6.7.1 Description

This window is divided into three main parts, which are:

- a text panel in which a Prolog source may be displayed by the debugger.
- a panel a buttons, which give access to the debugger commands.
- a text panel which contains the input/output from the debugger. This is where the user may type debugger commands.

A check button, called Connection indicates whether or not the Prolog IV output is displayed in this window (rather than in the Prolog IV console).

The buttons from the central panel simplify the usage from the most coommonly used debugger commands, by replacing their typing by a simple click. Since clicking on one of these buttons is normally equivalent to typing the associated command, the debugger must be waiting for a command when using them.



6.7.2 Usage

In order to use the source-oriented debugger, the contents of the window must have been compiled in `debug` mode, by checking the « Debug » item in the editor's « Prolog » menu, and by then compiling the content of the window.

In the Prolog IV console, the `debug` primitive needs to be used before to type a query. During execution of the query, the debugger window comes in front of the other ones as soon as a portion of code compiled in `debug` mode is active.

It is sometimes necessary to perform the `step` action several times before to get a source file loaded, since some of the constraints cannot be shown at the source level.

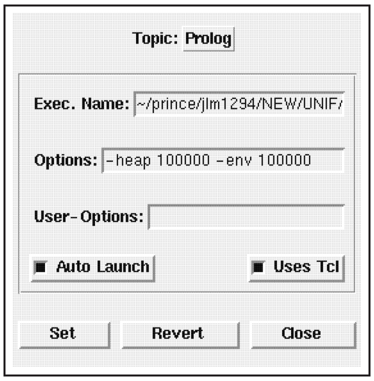
6.8 Dialogs

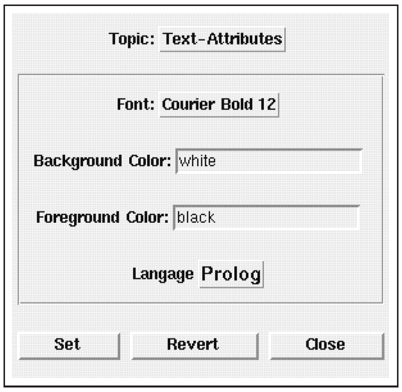
In this section, we present the main dialogs which are used in the Prolog IV graphical interface.

6.8.1 Preferences

This dialog allows the user to modify some of Prolog IV's settings, in two different themes:

- one about the parameters of the Prolog IV executable which runs in the console window;
- and one to set some global parameters about editor windows.





In both cases, the buttons Set, Revert and Close are present on the window. Their meanings are as follows:

$\boxed{\text{Set}}$: validates the values which have been entered.

$\boxed{\text{Revert}}$: resets the values to their values at the last Set.

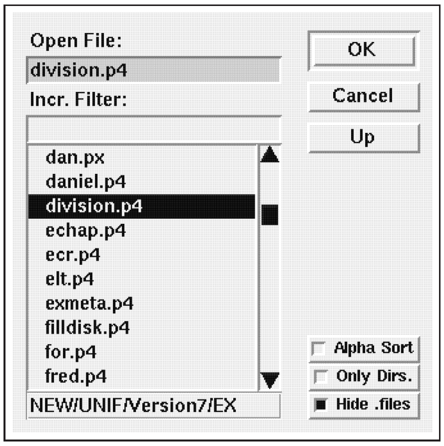
$\boxed{\text{Close}}$: closes the window: all changes which have not been validated by Set are lost.

Changing the theme also loses all the non-validated changes.

In order to save these various parameters for future sessions, you need to use the « Save Preferences » option from the « Info » menu in the main panel.

6.8.2 File selection dialog

This dialog is used to select a file, either to load it in a file or to give a file name before saving the content of a window. Here are the various parts of this dialog.



The editable fields are:

$\boxed{\text{Open File:}}$: or

$\boxed{\text{Load File:}}$: or

$\boxed{\text{Save File:}}$: , according to the nature of the operation. this field contains the name of the file to load or save.

Incr. Filter: : is used to incrementally type a file name. The content of the file list is updated as the typing goes.

A scrolled list of the current directory's files is displayed. It is possible to double-click on a file to directly select it (the dialog is then closed). When double-clicking on a directory, this directory becomes the current directory and the file list is updated.

Under the list is a scrollable label which contains the current directory name. This label cannot be edited, but it may be scrolled with the mouse.

The buttons :

OK : validates the current choice. If it is a file, it is selected; if it is a directory, then this directory becomes the current directory.

Cancel : cancels the on-going operation and closes the file selection dialog.

Up : makes the ancestor directory the current directory.

There also are come check buttons which, when selected, alter the behavior of the file selection dialog in the following way:

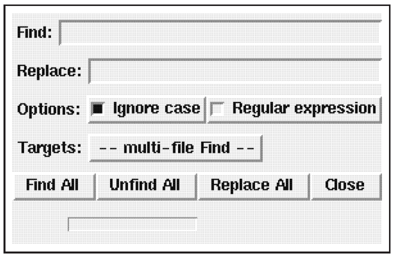
Alpha Sort. : sorts the names alphabetically, without taking the case into account.

Only Dirs. : only shows directories (files are hidden).

Hide .files : hides the files whose names start by a dot.

6.8.3 Search/Replace

This dialog is used to find strings in the editors and to replace these strings by other strings.



The editable fields are :

Find: : contains the string to find.

Replace: : contains the replacement string.

Checkbuttons are used to indicate the type of search to be performed:

Ignore Case: : ignores the case in the search.

Regular Expression: : the content of the Find: field is interpreted as a regular expression.

The Targets: menu may be used to select among the active editors which ones will be searched. One or several names can be selected from the menu.

The action buttons are:

- **Find All** : looks for the string in the Find: field in all the specified files, and colors the occurrences.
- **Unfind All** : removes all the coloring for the string currently in the Find: field.
- **Replace All** : replaces all occurrences of the string in the Find: field by the string in the Replace: field, in all the selected files.
- **Close** : closes the window.

A gauge and two occurrence counters (one for the current target file, and the other for all files) are displayed at the bottom of the dialog.

6.9 Keyboard and mouse

6.9.1 The mouse

TCL uses the two leftmost buttons of the mouse (*i.e.* both buttons on a two-button mouse, and the left and middle buttons on a 3-button mouse).

- The leftmost button is used for the main actions.
- The rightmost button's use depends on the context.

For instance, a click with the right button on a menu detaches it.

In all text panels and editable fields, maintaining the right button down while moving the mouse can be used to move rapidly.

Text selection is performed with the left button, with Shift-clic to extend a selection up to the point clicked.

6.9.2 Keys

Here are some informations about the use of the keyboard under some precise conditions.

Menu shortcuts

These shortcuts can be used to browse through the menus without using the mouse. They are highlighted in the menus by an underlined character in the menu name.

In order to get access to a menu, type `MOD`−*letter*, where *letter* is underlined in the current menu bar. When the menu is visible, the arrows can be used to browse through the items, and the `Return` key to select an item. A letter may also be used to select directly an item.

The `MOD` key is a modifier which is platform-dependant; on many platforms, the `Alt` key is used. On Sun stations, it is called `Alt Graph`, and on NeXT, it is called `Command`.

Keyboard shortcuts

Keyboard shortcuts can be used to perform an action, without going through a menu to activate it. They are usually shown on the right of the corresponding menu item.

Erase the selection

`Ctrl-d` erases the selection in a text editor or editable field.

Menu selection

The `F10` key opens the first menu of the window.

6.10 Miscellaneous

- The table of contents of the Read-me window is hypertext.
- The syntactic analyser used in the Check Syntax operation takes into account the `op/3` directives which appear in the file, but not the current state of Prolog IV.
- When the mouse cursor passes over a line which is colored by an error or a warning, the corresponding message is displayed in a box.
- There is an automatic backup procedure for editors. For every edited editor window, a file is created in the `/tmp` directory. Its name is `b.ed`N`.frEdit-`PID where N is an integer and PID is the process number of the graphical environment to which the editor belongs. The files are saved automatically every 5 minutes.
- Pasting something when the cursor is in the current selection works like if the something was pasted immediately after the selection.
- All temporary files (created in `/tmp`) are deleted at the end of a session.

6.11 Graphical primitives

6.11.1 Preliminaries

The graphical primitives may be used to create drawings, in the form of lines, rectangles and text. These primitives create « vectorial » objects, in the sense that the can be selected and destroyed selectively. It is also possible to enhance these objects by using TCL directly (which requires some expertise in TCL).

Coordinates

The coordinates are denoted by Prolog IV numbers, which can be integers or floating-point numbers. Rationals may also be used, but they need to have « small » values (*i.e.* it must be possible to handle them in C).

The coordinate system is defined as follows in each graphical panel:

- The x axis is horizontal and goes from left to right.
- The y axis is vertical and goes from top to bottom.
- The origin (coordinates 0,0) is in the panel's top right corner.

6.11.2 Primitives

gr_init initialises the graphical system (including the graphic window). This primitive must be used once before to use any other graphical primitive.

gr_moveto(X,Y) moves the pen to the point (X,Y), which becomes the current point.

gr_lineto(X,Y) draws a line from the current point to point (X,Y), which becomes the current point.

gr_line(DX,DY) draws a line from the current point (X_0,Y_0) to the point ($X_0 + DX, Y_0 + DY$), which becomes the current point.

gr_rect(V, [X,Y,X2,Y2]), gr_rect(V, X,Y,X2,Y2) draw a rectangle :

- V is the action (`frame` or `paint`).
- X,Y are the coordinates of the upper left corner.
- $X2$,$Y2$ are the coordinates of the lower right corner.

gr_drawstring_at(S, X,Y) draws a string object containing string S at point (X,Y).

gr_drawstring_at(S, X,Y, X2,Y2) draws a string object containing string S in the rectangle defined by its corners (X,Y) and ($X2$,$Y2$).

gr_arrow(X,Y, X2,Y2) draws an arrow from point (X,Y) to point ($X2$,$Y2$).

gr_setxfont(FX11) sets the current font to font $FX11$ (using the X11 font names).

gr_setfont(N), gr_setfont(N, T), gr_setfont(N, T, G) set the current font using some parameters:

- N is the name of the font (helvetica, times, courier, etc)
- T is the size (12, 14, 18, 24, etc)
- G is the typestyle (normal, bold, medium, etc)

The number of available fonts depends on the paltform.

gr_fgcolor3(R, G, B) positions the current color of objects to the value #RGB in which each component (red,green,blue) is defined by a number between 0 and 255.

gr_fgcolor(Col) positions the current color of objects to the value Col which must be a X11 color name (*e.g.* aquamarine, slategray, darkorchid, ...).

gr_settag(Tag) assigns the tag Tag to the next objects cerated. The special value `{}` states that no tag is to be used.

gr_erase destroys all the objects from the current graphical window.

gr_erase(Tag) destroys all the objects labeled by Tag from the current window.

gr_drawtree(T, R) draws the tree T in the rectangle R, which is a list of coordinates.

6.12 Problems and limits

The version of TCL/TK currently used is not the latest version, so some problems of this older version are still present. The next version of Prolog IV will use the latest version of TCL/TK, and most of the problems will be fixed.

- Under some specific (and rare) conditions, putting a window in the foreground may take a few seconds.
- In order to have access to the accelerators, the focus must be transferred explicitly to the window (*i.e.* by clicking on it).
- Syntax coloring can take some time on large files.
- It is not possible to find and replace backwards.
- It is not possible to find a string which contains a carriage return.
- Syntax coloring is not yet implemented for the C/C++ and TCL modes.
- There is no « Undo » in the editor.
- Some menu items are not yet implemented, and stay grey permanently.
- In some instances, the source code shown in the debugger window is not the right one.