NAME

ccfe.conf − the Curses Command Front-end configuration file.

DESCRIPTION

ccfe.conf is the main configuration file for **ccfe**. It determines the aspect of the screens provided by the Curses Command Front-end and some other features.

SYNOPSIS

The *ccfe configuration file* contains entries in the form:

```
global {}
menu_global {}
form_global {}
field_attr {}
browser_global {}
```

The order in which they appear is not important. **Comments** begin with the "#" character, and continue until the end of the line. Empty lines are ignored. Keywords are not case sensitive. Every *configuration section* is a collection of *parameters* with the following syntax:

```
configuration_section {
  parameter = value
  parameter = value
   ...
  parameter = value
}
```

Valid parameters are explained later with the following format:

parameter_name

Parameter description.

Values: Notes on the syntax of the value of the field. If is present a list of values, the first element of the list is the default. The `""` value represents the null value.
Required: Yes or *no*, depending if the parameter is mandatory or not.

Program instances

Implemented, but not yet documented!

Configuration overrides

It is possible to run CCFE by calling it with alternative names, so you can have a CCFE with a particular configuration for specific tasks: for example, you can configure CCFE to appear as the IBM AIX's System Management Interface Tool (SMIT), and invoke it with the **smit** command, so your operations people can have a similar interface to administer both AIX and both Linux hosts. In this case you simply need to make symbolic link (or a hard link) called **smit** to **ccfe**, so CCFE will refer all its objects with the name used to call it: it read the global configuration file **ccfe.conf** and next the *instance configuration file* smit.conf, then it refers for menus and forms to the directory $PREFIX/lib/smit instead the default `$PREFIX/lib/ccfe`, then load the default menu `$PREFIX/lib/smit/ccfe.menu`

GLOBAL CONFIGURATION SECTION

The **global {}** section defines general configuration parameters.

screen_layout

Defines the layout of the screen: **normal** if you want reverse background of the pop-up windows, centered menu options, screen areas separated by horizontal lines and titles in bold font, but if you plan to use **ccfe** with low functions terminals and you don't want these features, you can set it as **simple**.

Values: **normal|simple**
Required: No

hide_cursor

Show or hide the cursor.

Values: **YES**|**NO**
Required: No

show_screen_name

Show the file name of menu or form in the right of the top line of the screen.

Values: **YES**|**NO**
Required: No

path

Values: **YES**|**NO**
Required: No

shell

The shell used for the execution of every command string specified in forms and menus. **WARNING: no check is actually made on the validity of the shell:** for example, it is not verified that it is listed in `/etc/shells` system file.

user_shell

The shell called when user presses **<shell_escape key>** key. If **user_shell** is not present in configuration file, then the user's shell in `/etc/passwd` system file will be used. Else, if it is present but it is not valid (that is, not listed in `/etc/shells` system file), then the message with ID **BAD_SHELL_MSG** (see the **MULTILINGUAL SUPPORT** section in **ccfe**(1) manpage) will be displayed in a pop-up window. So, if you want to disable the user escape shell, you simply have to set an invalid value to **user_shell** parameter. For example

```
user_shell = DISABLED
```

log_level

Define the verbosity of the log file. The default configuration only log actions with **log** option. Except the first level, which can be used to trace user activity, all other details are useful for debugging purposes during new menus or forms development. Invoking **ccfe** with **-d** option enables the maximum log detail, depending by **permit_debug** parameter explained below.

1 (LOG_NORMAL)
Write to logfile the commands executed by the actions with the **log** attribute specified.

2 (LOG_LIST_CMD)
Write to logfile informations related **list_cmd** of type **command** attributes: executed commands and their output.

4 (LOG_DEFAULT_CMD)
Write to logfile details of commands executed to setup the default value of forms fields.

8 (LOG_ACTION_CMD)
Write to logfile the commands invoked by action blocks.

16 (LOG_FIELD_VALS)
Write to logfile the value of fields before execute action blocks.

32 (LOG_MENU_CHOICE)
Write to logfile selections in menus and pop-up menus.

64 (LOG_ACTION_OUT)
Write to logfile the output of executed action blocks.

128 (LOG_SYSCALL_ENV)
Write to logfile the environment used to execute external commands.

256 (LOG_SCAN_PATHS)
Write to logfile the search path used to look for forms, menus, and configuration files.

512 (LOG_INITFORM_OUT)
Write to logfile the output of executed init blocks.

Examples:

```
log_level = 1    # Log only actions with log attribute
```

log_level = 577 # 577 = 1 + 64 + 512

permit_debug

If **YES**, users can enable debugging messages in their logfiles by invoking **ccfe** with **-d** option. Please see the **ccfe**(1) manpage.

Values: **YES**|**NO**
Required: No

load_user_objects

Define if $HOME/.ccfe directory has to be scanned when load **ccfe** configuration, forms and menus.

Values: **NO|YES**
Required: No

key_f1
key_f2
key_f3
key_f4
key_f5
key_f6
key_f7
key_f8
key_f9
key_f10
key_f11
key_f12

Assign a function to keyboard standard programmable keys. The functions available are the following:

back Exits from current form or menu or pop-up window and go to the previous (if any) without executing action.

exit Exit from **ccfe**.

help Show the contextual help. Please see the **ccfe_help**(5) manual page.

list Show a pop-up window with a list of admitted values for the entry field to select from. Depending on the list type, it is possible to select just one value or multiple values. Keys enabled for the selection are:

<sel_items key> or **<Spacebar>**
Please see the explanation of the function **sel_items** below.

<A> or **<a>**
Only available in multiple values lists, select all the items.

<U> or **<u>**
Only available in multiple values lists, unselect all the items.

<Left arrow> and **<Right arrow>**
Scroll horizontally the menu if it is more wider than the pop-up window.

</> Find an item in the list, starting from the current and using the **menu_pattern**(3X) menu pattern match management.

<n> or **<N>**
Find the next item, using the search pattern previously entered with the **/** (find) command.

<Enter>
Close the pop-up window and put the selected value(s) in the field.

redraw Refresh the screen if it has been accidentally overwritten, for example, by a **write**(1) command from another user.

reset_field
Resets the value of the actual field to the one at the form submission.

save In form screens: save the actual value of the fields in the logfile. In output browser screen: save in a text file the displayed output, or save the action in a shell script ready for execution outside **ccfe**.

sel_items
Only available in multiple values lists, select or unselect the higlighted item.

shell_escape
Escapes temporary to a shell (please see **user_shell** parameter in **ccfe.conf**(5) man page).

show_action
In a *form screen*, displays a pop-up window with the action that will be executed by pressing **<Enter>**, with the values actually in the fields. In a *output browser screen*, displays a pop-up window with the executed action.

Please note that these parameters does not change the label of the function key on the screens: you have to modify the corrisponding definition **KEY_xxxx_LABEL** in the *global messages configuration file* `/usr/share/ccfe/msg/LOCALE_ID/ccfe`.

Example:

`In /etc/ccfe/ccfe.conf:`
key_f10 = exit

`and in /usr/lib/ccfe/msg/C/ccfe:`
KEY_F10_LABEL = "Quit"

Required: No

FORM CONFIGURATION SECTION

The **form_global {}** block defines the parameters common to all **ccfe** instance forms. Some parameters can depend by terminal name in which **ccfe** is running: please see the **TERMINAL-DEPENDENT SETTINGS** section below.

field_pad

Defines the character used to fill normal fields. Be careful to conversion to blanks performed by **ncurses**(3X) library: for example, if **field_pad="_"** and the user inserts the value "root_vg usr_vg var_vg" in a field, it appears on screen as "root_vg_usr_vg_var_vg", but the string "root vg usr vg var vg" will be passed to **action {}** block!

Values: A charachter delimited by double quote, for example: **field_pad = " "**
Default: **" "**
Required: No

hidden_field_pad

Defines the character used to fill fields of **hidden** type.

Values: A charachter delimited by double quote, for example: **hidden_field_pad = "X"**
Default: **"*"**
Required: No

initial_ovl_mode

Set the overlay mode at program startup to input characters in the fields; if **NO** the insert mode is selected.

Values: **YES**|**NO**
Required: No

show_changed_fields

Display with a different video attribute the value of the fields changed by the user. Please see the **changed_value_fg** and **changed_value_bg** parameters in the **FORM FIELD ATTRIBUTES CONFIGURATION SECTION**.

Values: **YES**|**NO**
Required: No

show_field_flags

Show flags (for instance, the ***** character before fields with required value) for every field.

Values: **YES**|**NO**
Required: No

show_dots

Fill with some dots the gap between prompt and value of fields of forms.

Values: **YES**|**NO**
Required: No

value_delimiters

Specify the characters to use to show when starts and ends the value of the field. Useful when **field_pad** is a blank.

Values: A couple of charachters delimited by double quote and separated by a comma: for example: **value_delimiters = "[","]"**
Default: **" "," "**
Required: No

field_value_pos

Define the column position of the screen for the fields value of the forms. The special value **-1** (default) means they will be right aligned.

Values: *integer value*
Default: **-1**
Required: No

fnkeys_rows

Define the number of bottom rows of the screen used to display the function key labels defined in the *ccfe message file definition*. Warning: no check is made on this value.

Values: *integer value*
Default: **1**
Required: No

FORM FIELD ATTRIBUTES CONFIGURATION SECTION

The **field_attr {}** and **active_field_attr {}** blocks defines the video attributes common to all **ccfe** instance forms. The accepted values are the standard **ncurses**(3X) video attributes, as defined in **<ncurses.h>**: for example

```
field_attr {
  label_fg = A_NORMAL
}
```

Please see them in the **curs_attr**(3X) manpage. It is possible to define **field_attr {}** and **active_field_attr {}** blocks for several terminal types: **ccfe** use the one with the terminal type where it is running; please see the **TERMINAL-DEPENDENT SETTINGS** section below.

label_fg
label_bg

Fields label foreground and background video attributes.

value_fg
value_bg

Fields initial value foreground and background video attributes.

changed_value_fg
changed_value_bg

Foreground and background video attributes for fields value changed by user.

Please see the **EXAMPLES** section below.

MENU CONFIGURATION SECTION

The **menu_global {}** block defines the parameters common to all **ccfe** instance menus.

mark_noaction_items

Options without defined action will be displayed inside parentheses.

Values: **NO|YES**
Required: No

fnkeys_rows

Define the number of bottom rows of the screen used to display the function key labels defined in the *ccfe message file definition*. Warning: no check is made on this value.

Values: *integer value*
Default: 1
Required: No

OUTPUT BROWSER CONFIGURATION SECTION

The **browser_global {}** block defines the parameters for the screen used to browse the output of an executed menu or form action. Available parameters are:

max_rows

Size of the buffer used to display the standard output and error of the action executed. If the action produces a bigger output, then a pop-up message window will inform the user how to view it for entire.

info_attr

Video attribute used to display the **ccfe** generated messages (for example, when an action is interrupted with **<Ctrl>+C**).

Values: a standard **ncurses**(3X) video attribute, as defined in **<ncurses.h>**. Please see them in the **curs_attr**(3X) manpage.
Default: **A_REVERSE**
Required: No

end_marker

String to show at the end of the output produced by **action{}** block. **info_attr** is used as video attribute.
To disable it, please set it to a null value:

```
end_marker =
```

Values: *string value*
Default: ****** end of output ******
Required: No

stderr_attr

Video attribute used to display the standard error rows.

Values: a standard **ncurses**(3X) video attribute, as defined in **<ncurses.h>**. Please see them in the **curs_attr**(3X) manpage.
Default: **A_BOLD**
Required: No

stdout_attr

Video attribute used to display the standard output rows.

Values: a standard **ncurses**(3X) video attribute, as defined in **<ncurses.h>**. Please see them in the **curs_attr**(3X) manpage.
Default: **A_NORMAL**
Required: No

fnkeys_rows

Define the number of bottom rows of the screen used to display the function key labels defined in the *ccfe message file definition*. Warning: no check is made on this value.

Values: *integer value*
Default: **1**
Required: No

TERMINAL-DEPENDENT SETTINGS

Some settings can depend on terminal type used to run **ccfe**. For example

```
form_global { field_pad.linux = "_" }
```

sets the parameter **field_pad** only for terminals of type **linux**. The *terminal ID* to use is the same of the environment variable TERM, case sensitive. The effective parameter value depends by the order of its configuration: for example:

- Sets **field_pad=" "** for all terminal types except **linux** field_pad="_".

  ```
  form_global {
    field_pad        = " "
    field_pad.linux  = "_"
    hidden_field_pad = "*"
  }
  ```

- Sets **field_pad=" "** for every terminal type: the **field_pad** overrides the preceding **field_pad.xterm** and **field_pad.linux**:

  ```
  form_global {
    field_pad.linux  = "_"
    field_pad.xterm  = "."
    field_pad        = " "
    hidden_field_pad = "*"
  }
  ```

Terminal-dependent settings are:

- form_global { field_pad }
- field_attr {}
- active_field_attr {}

EXAMPLES

```
# The Curses Command Front-end default configuration file.

global {
  screen_layout       = Normal
  hide_cursor         = YES
  shell               = /bin/sh
  log_level           = 1
  #path                = /path/to/add:/another/path/to/add
  #user_shell          = /bin/ksh
}

browser_global {
  max_rows    = 5000
  info_attr   = A_REVERSE
  stderr_attr = A_BOLD
  stdout_attr = A_NORMAL
  end_marker  = **** end of output ****
}

menu_global {
  mark_noaction_items = YES
}

form_global {
```

```
    field_pad           = " "
    field_pad.linux     = "_"
    hidden_field_pad    = "*"
    show_changed_fields = YES
    show_field_flags    = YES
    initial_ovl_mode    = YES
  }

  field_attr {
    label_fg         = A_NORMAL
    label_bg         = A_NORMAL
    value_fg         = A_NORMAL
    value_bg         = A_UNDERLINE
    changed_value_fg = A_BOLD
    changed_value_bg = A_UNDERLINE
  }
  field_attr.linux {
    label_fg         = A_NORMAL
    label_bg         = A_NORMAL
    value_fg         = A_NORMAL
    value_bg         = A_NORMAL
    changed_value_fg = A_BOLD
    changed_value_bg = A_NORMAL
  }

  active_field_attr {
    label_fg         = A_BOLD
    label_bg         = A_NORMAL
    value_fg         = A_NORMAL
    value_bg         = A_REVERSE
    changed_value_fg = A_NORMAL
    changed_value_bg = A_REVERSE
  }
```

FILES

`/etc/ccfe/ccfe.conf`
Global configuration.

`/etc/ccfe/CALLNAME.conf`
Instance configuration. For example: `/etc/ccfe/foo.conf`, `/etc/ccfe/bar.conf`

`$HOME/.ccfe/ccfe.conf`
User global preferences.

`$HOME/.ccfe/CALLNAME.conf`
User instance preferences. For example: `$HOME/.ccfe/foo.conf`

`/usr/share/ccfe/msg/LOCALE_ID/CALLNAME`
Instance messages configuration.

SEE ALSO

ccfe(1), **ccfe_form**(5), **ccfe_menu**(5), **ncurses**(3X), **curs_attr**(3X), **form**(3X), **shells**(5)