NAME

ccfe - the Curses Command Front-end.

SYNOPSIS

ccfe [*OPTION*]... [*SHORTCUT*]

DESCRIPTION

ccfe is a simple tool to quickly supply an interactive screen-oriented interface to command line scripts and commands: it prompts users for the informations needed to run them, and can be programmed with your preferred shell to provide predefined selections and/or run-time defaults. It also provides a menu system to hierarchically organize them and a viewer to browse the standard output and standard error of invoked script/command.

Options

-c Print some configuration parameters and exit. May be useful in pre-installation scripts for packaged **ccfe plugins**.

-d Enable verbose logging for debugging. It depends from the value of the **permit_debug** parameter in the **ccfe.conf**(5) configuration file.

-h Display help message and exit.

-l *PATH*

Override the configured library directories (global and user) to search for menus and forms to load: for example, you can create in the current directory the form `myapp.form` as frontend to run the script `myapp.sh` and then run it by

```
ccfe -l . myapp
```

Only one path can be specified. This can be useful to distribute software with **ccfe** as front-end, or if you want to use forms and menus in a directory alternative to default.
Note that *PATH* is absolute for every **ccfe** instance.
Please see also the explanation of the **CCFE_LIB_DIR** environment variable.

-s Print ordered list of available shortcuts to forms or menus and exit. A tilde character ("~") after shortcut name means that menu or form is user private and located in its home directory.

-v Print version and exit.

SHORTCUT

Initial form or menu name to load (without extension).

SCREEN DESCRIPTION

ccfe has the following type of screens:

Menu screens

Show a list of items and when a user select one of them, another menu or form is presented, or a shell command is executed.

Form screens

Show a list of fields, then a shell command is executed using the fields value for arguments and options. Each field can have some flags before and/or after them:

***** Indicates that a value in the field is required to proceed.

+ Indicates that is available the pop-up list of valid values for the selection by pressing the key **<list key>** (please see the following section **KEY BINDINGS**).

Indicates that the field accept only numeric values (please see the **type** attribute description of the **ccfe_form**(5) manpage).

Moreover, depending of the configuration in **ccfe.conf**(5) file, the following symbols also have a specific meaning:

[ ] Field value size delimiters. Please see the attribute **value_delimiters** in the **ccfe.conf**(5) manpage.

< > In the fields with the attribute **hscroll** enabled, they indicate off-screen data ahead or behind. Please see the attributes **len** and **hscroll** in the **ccfe_form**(5) manpage.

When the value of a field change regarding the one at the form submission, it will be highlighted according the attributes **changed_value_fg changed_value_bg** of the **field_attr{}** and **active_field_attr{}** stanzas in the **ccfe.conf**(5) configuration file.

Output Browser screens

The stdout and stderr of a shell command executed by a menu or a form can be browsed in a user-friendly mode, like when they are redirected and piped to **more**(1) or **less**(1) or **pg**(1) standard commands.

Each of the above **ccfe** screens is divided in three areas:

Header

The top area which contains the following informations:

username@hostname (if enabled in **ccfe.conf**(5))

Screen title

Screen name: Filename of the form/menu in the *dialog area* (if enabled in **ccfe.conf**(5))

Ins/Ovl flag: Report the insert/overlay character mode in fields of forms (only in form screens)

Op/Pg indicator: Report the position of the highlighted menu option (in menu screens) or the page displayed of a multi-page form (in form screens and in the output browser screen).

Dialog area

The middle area which contains a menu, a form or the output of a command. *standard output* and *standard error* are mixed in the output of a command, but they are displayed with different screen attributes (please see the parameters **stdout_attr** and **stderr_attr** in the **ccfe.conf**(5) manpage).

Footer The bottom area which contains the list of the enabled function keys.

It is possible to add and remove some aesthetic attributes (lines, alignment, video attributes, etc.) to change the aspect of the screens: please see the **ccfe.conf**(5) manpage.

Form pop-up list

When you press **<list key>** key in a field with flag **+**, you open a pop-up window and you can select one or more (it depends from the **list_cmd** options) items from its list. The bottom of the window shows the active keys and in the upper right corner there are the following informations:

< Indicates that there is off-screen data in the left of the window. They can be viewed by scrolling the data pressing the **<Left arrow>** and **<Right arrow>** keys.

x/y:z Indicates the position of the cursor in the list of items: **x** is the item highlighted, **y** is the total number of items in the list, **z** is the number of the selected items. **:z** is displayed only if *list-type* is **multi-val** in the **list_cmd** attribute, as described in **ccfe_form**(5) manpage.

> Indicates that there is off-screen data in the right of the window. They can be viewed by scrolling the data pressing the **<Left arrow>** and **<Right arrow>** keys.

KEY BINDINGS

Here is described the function of the keys enabled in the **ccfe** screens. Note that the meaning of the function keys **<F1>**,...,**<F10>** can be remapped in the `ccfe.conf` configuration file. They are referred by their function definition: for example, if the key **<F5>** is defined as **key_f5** = **back** in `ccfe.conf`, below it will be referred as **<back key>** instead **<F5>**. Please see the **key_f1**,...,**key_f10** parameters in the **ccfe.conf**(5)

man page.

Common Commands

The following keys are available in every kind of screen, but enabled in agreement with `ccfe.conf` configuration:

<help key>
Help/Contextual help (not yet implemented).

<redraw key>
Refresh screen.

<back key> or **<Esc>**
Return to previous screen (Cancel current action).

<shell_escape key>
Escape temporary to a shell (please see **user_shell** parameter in **ccfe.conf**(5) man page).

<exit key>
Exit from **ccfe**.

Menu Commands

In menu screens, the following keys are enabled:

<Up arrow>
Move the cursor to the previous option.

<Down arrow>
Move the cursor to the next option.

<Home>
Move the cursor to the first option of the menu.

<End> Move the cursor to the last option of the menu.

<Enter>
Select the higlighted option.

Form Commands

In form screens, the following keys are enabled. Note that if there are off-screen data behind in the current field, a flag ">" is displayed at the end of the field, and if there are off-screen data ahead in the current field, a flag "<" is displayed at the beginning of the field.

<Up arrow>
Move the cursor to the previous field in the displayed page of the form. If pressed when in the first field of the page, the cursor will be moved to the last field of the page.

<Down arrow>
Move the cursor to the next field in the displayed page of the form. If pressed when in the last field of the page, the cursor will be moved to the first field of the page.

<Left arrow>
Move the cursor to the previuos character in the field value.

<Right arrow>
Move the cursor to the next character in the field value.

<Home>
Move the cursor to the first character in the field value.

<End> Move the cursor to the last character in the field value.

<PgUp>
Show the previous page of fields (if any) and move the cursor to the last field of the page.

<PgDn>
Show the next page of fields (if any) and move the cursor to the first field of the page.

<Enter>
Run the action of the form.

<Tab>, **<Shift>+<Tab>**
Browse (forward and backward) the list of accepted values in a field of type **BOOLEAN**, **NULLBOOLEAN** or of any other type where attribute **list_cmd** has *source-type* **const** and *list-type* **single-val**, for example:

```
list_cmd = const:single-val:"K : Kilobytes",
                            "M : Megabytes",
                            "G : Gigabytes"
```

Please see also the **list_cmd** attribute description of the **ccfe_form**(5) manpage.

<list key>
Show in a pop-up window a list of admitted values for the field. 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>**
Only available in multiple values lists, select or unselect the higlighted item.

<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 wide than the pop-up window. CHECK

</> 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>
Put the selected value(s) in the field and close the pop-up window.

<reset_field key>
Reset the value of the actual field to the one at the form submission.

<show_action key>
Show the action that will be executed pressing **<Enter>**, with the values at the moment in the fields.

<save key>
Save the current value of the fields in the logfile.

Output browser Commands

<Ctrl>+C
This combination of keys is available only during the execution of an action: sends a SIGINT signal to the running process.

</> Find a string in the output shown in the browser, starting from the current position. The search is case sensitive.

<n> or **<N>**
Find the next string occurrence of the one entered in the / (find) command.

<save key>
Save in a text file the output displayed in the browser, or save the action in a shell script ready for execution outside **ccfe**.

INVOCATION

When **ccfe** is started with its name, it executes the following ordered operations:

1. Read the main configuration file `/etc/ccfe/ccfe.conf`
2. If exists, read the user configuration file `$HOME/.ccfe/ccfe.conf`
3. If exists, load the main menu `/usr/share/ccfe/ccfe.menu`, else load (if present) the user menu `$HOME/.ccfe/ccfe.menu`

When you specify a *shortcut* (for example **foo**), instead to perform step 3 **ccfe** searches the following objects in the order in which they appear and loads the first that matches:

- `/usr/share/ccfe/foo.menu`
- `/usr/share/ccfe/foo.form`
- `$HOME/.ccfe/foo.menu`
- `$HOME/.ccfe/foo.form`

If any configuration file is found, then **ccfe** starts with its predefined configuration; if any *ccfe main menu* or *shortcut* is found, **ccfe** exits. Private user configuration and forms/menus are read depending by **load_user_objects** parameter in `/etc/ccfe/ccfe.conf` file.

If **ccfe** is started whith a different name because are configured one or more application instances, its behavior is different: please see the **APPLICATION INSTANCES** paragraph below.

SHORTCUTS

It is possible to load directly a specific menu or form instead the default menu (`ccfe.menu`) when **ccfe** is invoked. You have to specify the desired form o menu file name (without the extension), and it will be searched through the following ordered list of directories:

- `$CCFE_LIB_DIR/$CALLNAME` (for example `/usr/local/lib/ccfe`)
- `$HOME/.ccfe/$CALLNAME` (for example `/home/foo/.ccfe/ccfe`)

If a menu and a form in the same directory have identical filename, then the menu will be loaded. *$CALLNAME* is the name used for **ccfe** invocation, please see **APPLICATION INSTANCES**.

MULTILINGUAL SUPPORT

Not yet supported. Actually, you can only translate messages in the *ccfe messages definition file* `/usr/share/ccfe/msg/C/ccfe`.

APPLICATION INSTANCES

Implemented, bud not yet documented!

LOGGING

ccfe logging can be used by users to record the their actions, but is generally used to records detailed trace information useful when developing new menus or forms. Every user have a separate and private log file in

the /var/log/ccfe directory, and the detail of the informations recorded can be specified in the **log_level** parameter of the **ccfe.conf**(5) configuration file. Normally is logged only the output of actions with the **log** option specified (please see **ccfe.conf**(5) manpage), but when **ccfe** is called with **-d** option, according with the value of attribute **permit_debug** in **ccfe.conf**(5), it switches in "debug mode" and the maximum level of information are logged.

PLUGINS

ccfe can be easily expanded by adding **plugins** (or **modules**). A plugin is the collection of files required to add new funcionalities. It can be very simple (only one file for a main menu entry) or very complex (a file for the new main menu entry, new menus, new forms and all shell scripts required by actions). Example of simple plugin: if your **ccfe** main menu is a *dynamic menu*, to add the new function *top* which call the **top**(1) command, you only have to put the file `/usr/share/ccfe/ccfe/ccfe.menu/top.item` which the following content:

```
item {
  id     = TOP
  descr  = Display O.S. tasks
  action = system:top
}
```

In a hypothetical pre-installation script of a plugin package, you can call **ccfe** to know the prefix directory where to install the plugin files, for example, the line:

```
eval $(ccfe -c | grep LIB)
```

sets up the variable LIB_DIR with value */usr/local/ccfe/lib* (for example), or the line

```
eval CCFE_$(ccfe -c  | grep LIB)
```

sets up the variable CCFE_LIB_DIR.

ccfe instances supports **plugins.** Please see also the **DYNAMIC MENUS** section in the **ccfe_menu**(5) manpage and the samples provided with **ccfe** package for a more complex examples of **plugin**.

ENVIRONMENT

Before the execution of an **action {}** block, **ccfe** sets up the following environment variables:

CCFE_LIB_DIR

The *ccfe library directory* (for example `/usr/local/ccfe/lib`). It can be used in scripts called by **ccfe** to refer to absolute pathnames. You can also define this variable before run **ccfe** to change the default library directory; it is an alternative to option "-l": for example, to load a form in current directory:

```
ccfe -l . myform
```

is equivalent to

```
CCFE_LIB_DIR=. ccfe myform
```

or

```
CCFE_LIB_DIR=. ; export CCFE_LIB_DIR
ccfe myform
```

CCFE_IWD
Ccfe invocation directory, the current working directory of the shell when invoked **ccfe**.

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/persistent/`
Persistent forms data.

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

`/usr/share/ccfe/ccfe/`
CCFE forms, menu and objects used by them. Forms, menus and objects used by them when CCFE is invoked as **ccfe**.

`/usr/share/ccfe/foo/`
Instance configuration: forms, menus and objects used by them when CCFE is invoked as **foo**.

`/usr/share/ccfe/bar/`
Instance configuration: forms, menus and objects used by them when CCFE is invoked as **bar**.

`/usr/share/ccfe/msg/LOCALE_ID/ccfe`
Global messages configuration. For example: `/usr/share/ccfe/msg/C/ccfe`

`/usr/share/ccfe/msg/LOCALE_ID/CALLNAME`
Instance messages configuration. For example: `/usr/share/ccfe/msg/C/foo`

`/var/log/ccfe/$USER.log`
User activity logs (if enabled), useful for debug developing new forms and menus, form field values backup before changes and audit.

SECURITY

ccfe is only a front-end: the security of the execution of a command is based on the O.S. permissions of the user who invoked **ccfe**. Execution of privileged commands is possible, for example, by using tools as **sudo**(1) in the **action** section of forms and menus.

It is possible to deny users to make their own **ccfe** interfaces and disable the temporary shell escape: please see the **load_user_objects** and the **user_shell** parameters in the **ccfe.conf**(5) manual page.

SEE ALSO

ccfe.conf(5), **ccfe_form**(5), **ccfe_menu**(5), **sudo**(1).

`/usr/share/doc/ccfe/`