CFLIB - Flexible Configuration Library

Patchlevel 20

Generated by Doxygen 1.5.8

Sat Jan 31 03:05:11 2009

Contents

1 CFLIB Documentation **1**

1.1 Introduction and General Issues . . . 1
1.2 Basic Usage . . . 1
1.3 Retrieval Functions . . . 2
1.4 Setting Parameters and Saving the Configuration . . . 2
1.5 General Utilities . . . 3
1.6 Advanced Usage . . . 3
1.6.1 Error Handling . . . 3
1.6.2 Report Generation . . . 3
1.6.3 Configuration Debugging . . . 3
1.6.4 Get Information About a Configuration Parameter: . . . 3
1.6.5 Other Functions and Features . . . 3
1.7 Compilation and Development Issues . . . 4
1.8 Help and Support . . . 4

2 CFLIB License **4**

3 Purpose of CFLIB **15**

4 General Notes **15**

4.1 The names "CFLIB", "libcf", "∗cf∗" . . . 15
4.2 Input and Output . . . 15
4.3 General (future) tasks . . . 15
4.4 Other general issues . . . 16

5 Simple Usage Example **16**

6 Configuration Initializer **16**

6.1 Parameter Name . . . 17
6.2 Parameter Default Value . . . 17
6.3 Commandline Option for Parameter . . . 17
6.4 Special Option Flags Mask for Parameter . . . 18

7 Configuration Parsing Levels and Source/Origin Options **18**

7.1 Origin Window . . . 18
7.2 Parsing Levels . . . 18
7.3 Parsing Levels in the Initialization Process . . . 19

7.3.1 Parsing Level Priorities . . . 19
7.4 Parsing of Configuration Files . . . 20
7.5 Residual Items . . . 20

8 Configuration Files **20**
8.1 Configuration File Format . . . 21
8.2 Reading Configuration Files . . . 21
8.3 Writing a (private) Configuration File . . . 21

9 Types of Parameters **22**
9.1 Parameter Data Types . . . 22
9.2 Special Processing Instructions . . . 22
9.3 Special Handling Instructions . . . 22
9.4 CFLIB Parameters . . . 23
9.5 FLAG Parameter Type . . . 23

10 Advanced Usage Example **23**

11 Compilation and Development **25**
11.1 Library Versions . . . 25
11.2 Platforms . . . 25
11.3 Building the Library . . . 26
11.4 Binaries and Executables . . . 26
11.5 Minimal CFLIB Replacement . . . 26
11.6 System and Compiler dependent Defines . . . 26

12 Bug List **27**

13 Todo List **28**

14 Module Index **28**
14.1 Modules . . . 28

15 Data Structure Index **28**
15.1 Data Structures . . . 28

16 File Index **28**
16.1 File List . . . 28

17 Module Documentation **29**
17.1 CFLIB Core Features . . . 29

17.1.1 Detailed Description . . . 31
17.1.2 Function Documentation . . . 31
17.2 Special Options Mask . . . 33
17.2.1 Detailed Description . . . 35
17.3 Error Handling . . . 35
17.3.1 Detailed Description . . . 38
17.3.2 Function Documentation . . . 39
17.4 Advanced Features . . . 40
17.4.1 Detailed Description . . . 40
17.4.2 Function Documentation . . . 40
17.5 Information Retrieval . . . 42
17.5.1 Detailed Description . . . 43
17.5.2 Define Documentation . . . 43
17.5.3 Function Documentation . . . 46
17.6 Setting and Saving the Configuration . . . 48
17.6.1 Detailed Description . . . 48
17.6.2 Function Documentation . . . 48
17.7 General Utilities . . . 50
17.7.1 Function Documentation . . . 51
17.8 Report Generation . . . 53
17.8.1 Function Documentation . . . 54

18 Data Structure Documentation **54**
18.1 CONFERR Struct Reference . . . 54
18.1.1 Detailed Description . . . 55
18.2 CONFIG Struct Reference . . . 55
18.2.1 Detailed Description . . . 55

19 File Documentation **55**
19.1 include/cf.h File Reference . . . 55
19.1.1 Detailed Description . . . 65
19.1.2 Define Documentation . . . 65

1 CFLIB Documentation

1.1 Introduction and General Issues

CFLIB is a small, simple, flexible and portable ANSI C Library to be used as configuration interface for user programs. CFLIB builds and maintains a compact database structure consisting of a list of parameters with their name, content and some additional information about each parameter.

CFLIB targets the basic needs of technical, scientific or other programmers who want to spend minimal time on coding input, output, variable parsing, report generation and the like but still have a simple to use, reliable, flexible and portable configuration interface for their programs.

Main Features:

- Commandline, environment and terminal input parsing
- Configuration files
- File search
- Template driven report generation
- Automatic time and date update

See also:

- Purpose of CFLIB
- CFLIB License
- The names "CFLIB", "libcf", "∗cf∗"
- General Notes

Project Homepage:

- http://cflib.berlios.de

1.2 Basic Usage

1. Include cf.h

2. Define the Configuration Initializer, an array of CONFIG structures

3. Call cfinit() with Configuration Initializer and Commandline. The most compact initialization is done by cfstart(), a wrapper function for cfinit() that includes error reporting, usage message and (optional) debugging output. On initialization the following data sources are inspected or parsed in the order presented:

- Commandline Arguments (or compatible structure given to cfinit()) according to the description in Commandline Option for Parameter
- Environment Variables
- Configuration Files

- Built-in User-defined Default from Configuration Initializer: Parameter Default Value
- Get parameter value from Standard Input `stdin`, if required

4. Use the Retrieval Functions **cfget**∗() to access configuration parameters

5. Compile your program, linking the appropriate CFLIB library file for your platform and setup, usually the file name is *libcf.a* which means the library is referred to as **"cf"**. You can change the names to fit into your setup: See The names "CFLIB", "libcf", "∗cf∗"

6. Run your program, test CFLIB functionality and adjust the Configuration Initializer according to your needs

See also:

Simple Usage Example
Configuration Parsing Levels and Source/Origin Options
CFLIB Core Features

1.3 Retrieval Functions

1. Get CFLIB Version and Copyright Information: cfgetvers(), cfgetsubvers(), cfgetcpr()

2. Get Usage Message for Output: cfgetusg()

3. Get Configuration Parameter Value:

- Get parameter value: cfget(), interpretation of content and return type depend on the type setting in the parameter's Special Options Mask
- Get string value: cfgetstr()
- Get integer value: cfgetnum()
- Get real/float value: cfgetreal()
- Inquire flag/switch status: cfgetflag()
- Get value of (next) residual command line argument: cfgetres()

Note:

All of these functions except cfgetres() require the parameter name as argument

1.4 Setting Parameters and Saving the Configuration

- cfput() : Update or Add Parameter (Utility Function Macro).
- cfputstr() : Update or Add Parameter *name* with string *content*.
- cfputtime() : Set all Time and/or Date entries in CFLIB DB to *now* or *today*.
- cfnosave() : Alter or query the CF_NOSAVE Flag of Parameter *name*.
- cfsave() : Write configuration data to a Configuration File or `stdout`.
- See Configuration Files for details

1.5 General Utilities

These functions and Macros are used in the library but do not depend on the configuration database or any cf∗() functions. They are (small) general tools that you can use in your program if you like.

- String Manipulation: EatWhiteSpace(), RemoveCR(), RemoveTrailSpace()
- File Utilities: FindFile(), BackupFile()
- Other Tools: IsATerminal(), DelFlag(), SetFlag()

1.6 Advanced Usage

1.6.1 Error Handling

CFLIB maintains a simple global Error Stack that is used by library functions like cfinit() when multiple errors can occur. Error Items consist of a numeric Error Code and (optionally) an Error Message string. Repeated calls to cfgeterr() will successively return error entries while deleting them from the stack until the list is empty. User programs may also use the CFLIB error stack by calling cfputerr() without the need to initialize a configuration.

For more details see: Error Handling - Error Codes, Functions and Structures.

1.6.2 Report Generation

- Generate Output from Template and current parameter values: cfform()

1.6.3 Configuration Debugging

These functions are thought to be used by the programmer working with CFLIB during development and testing of a program.

- Dump Configuration DB in human readable form: cfdump()
- Test and Dump Configuration Initializer: cfdinichk()

1.6.4 Get Information About a Configuration Parameter:

- Get source/origin of the parameter's value: cfgetsrc()
- Inquire Bit from parameter's Special Options Mask by Offset: cfflaginq()

1.6.5 Other Functions and Features

- Exit Configuration: cfexit()
- Expand User Home Directory in a File Path Parameter: cfhomexp()
- General (internal) retrieval function: cfgetent()

1.7 Compilation and Development Issues

- Platforms
- Binaries and Executables
- Building the Library
- System and Compiler dependent Defines
- Minimal CFLIB Replacement

1.8 Help and Support

- **Help** is this documentation
- **Support** is the (open) source code
- The project is maintained from time to time as needed ;-)
- Comments, Bug Reports or (better) Bug Fixes are welcome!
- See CFLIB License
- ... Have Fun!

2 CFLIB License

This file is part of **CFLIB** - Flexible Configuration Library.

Author:

Stefan Habermehl <stefan.habermehl@mcff.de>

Copyright:

(c) 1994,1995,1996,1997,1998,2006,2007,2008,2009 Stefan Habermehl

CFLIB is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

CFLIB is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with CFLIB (see LICENSE.txt). If not, see <http://www.gnu.org/licenses/>.

```
                    GNU GENERAL PUBLIC LICENSE
                       Version 3, 29 June 2007

 Copyright (C) 2007 Free Software Foundation, Inc. <http://fsf.org/>
 Everyone is permitted to copy and distribute verbatim copies
 of this license document, but changing it is not allowed.

                              Preamble
```

```
  The GNU General Public License is a free, copyleft license for
software and other kinds of works.

  The licenses for most software and other practical works are designed
to take away your freedom to share and change the works.  By contrast,
the GNU General Public License is intended to guarantee your freedom to
share and change all versions of a program--to make sure it remains free
software for all its users.  We, the Free Software Foundation, use the
GNU General Public License for most of our software; it applies also to
any other work released this way by its authors.  You can apply it to
your programs, too.

  When we speak of free software, we are referring to freedom, not
price.  Our General Public Licenses are designed to make sure that you
have the freedom to distribute copies of free software (and charge for
them if you wish), that you receive source code or can get it if you
want it, that you can change the software or use pieces of it in new
free programs, and that you know you can do these things.

  To protect your rights, we need to prevent others from denying you
these rights or asking you to surrender the rights.  Therefore, you have
certain responsibilities if you distribute copies of the software, or if
you modify it: responsibilities to respect the freedom of others.

  For example, if you distribute copies of such a program, whether
gratis or for a fee, you must pass on to the recipients the same
freedoms that you received.  You must make sure that they, too, receive
or can get the source code.  And you must show them these terms so they
know their rights.

  Developers that use the GNU GPL protect your rights with two steps:
(1) assert copyright on the software, and (2) offer you this License
giving you legal permission to copy, distribute and/or modify it.

  For the developers' and authors' protection, the GPL clearly explains
that there is no warranty for this free software.  For both users' and
authors' sake, the GPL requires that modified versions be marked as
changed, so that their problems will not be attributed erroneously to
authors of previous versions.

  Some devices are designed to deny users access to install or run
modified versions of the software inside them, although the manufacturer
can do so.  This is fundamentally incompatible with the aim of
protecting users' freedom to change the software.  The systematic
pattern of such abuse occurs in the area of products for individuals to
use, which is precisely where it is most unacceptable.  Therefore, we
have designed this version of the GPL to prohibit the practice for those
products.  If such problems arise substantially in other domains, we
stand ready to extend this provision to those domains in future versions
of the GPL, as needed to protect the freedom of users.

  Finally, every program is threatened constantly by software patents.
States should not allow patents to restrict development and use of
software on general-purpose computers, but in those that do, we wish to
avoid the special danger that patents applied to a free program could
make it effectively proprietary.  To prevent this, the GPL assures that
patents cannot be used to render the program non-free.

  The precise terms and conditions for copying, distribution and
modification follow.

                       TERMS AND CONDITIONS

  0. Definitions.

  "This License" refers to version 3 of the GNU General Public License.
```

"Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

"The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

A "covered work" means either the unmodified Program or a work based on the Program.

To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require,

such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to

produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

a) The work must carry prominent notices stating that you modified it, and giving a relevant date.

b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".

c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.

d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.

b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no

```
    further charge.  You need not require recipients to copy the
    Corresponding Source along with the object code.  If the place to
    copy the object code is a network server, the Corresponding Source
    may be on a different server (operated by you or a third party)
    that supports equivalent copying facilities, provided you maintain
    clear directions next to the object code saying where to find the
    Corresponding Source.  Regardless of what server hosts the
    Corresponding Source, you remain obligated to ensure that it is
    available for as long as needed to satisfy these requirements.

    e) Convey the object code using peer-to-peer transmission, provided
    you inform other peers where the object code and Corresponding
    Source of the work are being offered to the general public at no
    charge under subsection 6d.

  A separable portion of the object code, whose source code is excluded
from the Corresponding Source as a System Library, need not be
included in conveying the object code work.

  A "User Product" is either (1) a "consumer product", which means any
tangible personal property which is normally used for personal, family,
or household purposes, or (2) anything designed or sold for incorporation
into a dwelling.  In determining whether a product is a consumer product,
doubtful cases shall be resolved in favor of coverage.  For a particular
product received by a particular user, "normally used" refers to a
typical or common use of that class of product, regardless of the status
of the particular user or of the way in which the particular user
actually uses, or expects or is expected to use, the product.  A product
is a consumer product regardless of whether the product has substantial
commercial, industrial or non-consumer uses, unless such uses represent
the only significant mode of use of the product.

  "Installation Information" for a User Product means any methods,
procedures, authorization keys, or other information required to install
and execute modified versions of a covered work in that User Product from
a modified version of its Corresponding Source.  The information must
suffice to ensure that the continued functioning of the modified object
code is in no case prevented or interfered with solely because
modification has been made.

  If you convey an object code work under this section in, or with, or
specifically for use in, a User Product, and the conveying occurs as
part of a transaction in which the right of possession and use of the
User Product is transferred to the recipient in perpetuity or for a
fixed term (regardless of how the transaction is characterized), the
Corresponding Source conveyed under this section must be accompanied
by the Installation Information.  But this requirement does not apply
if neither you nor any third party retains the ability to install
modified object code on the User Product (for example, the work has
been installed in ROM).

  The requirement to provide Installation Information does not include a
requirement to continue to provide support service, warranty, or updates
for a work that has been modified or installed by the recipient, or for
the User Product in which it has been modified or installed.  Access to a
network may be denied when the modification itself materially and
adversely affects the operation of the network or violates the rules and
protocols for communication across the network.

  Corresponding Source conveyed, and Installation Information provided,
in accord with this section must be in a format that is publicly
documented (and with an implementation available to the public in
source code form), and must require no special password or key for
unpacking, reading or copying.

  7. Additional Terms.
```

"Additional permissions" are terms that supplement the terms of this
License by making exceptions from one or more of its conditions.
Additional permissions that are applicable to the entire Program shall
be treated as though they were included in this License, to the extent
that they are valid under applicable law. If additional permissions
apply only to part of the Program, that part may be used separately
under those permissions, but the entire Program remains governed by
this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option
remove any additional permissions from that copy, or from any part of
it. (Additional permissions may be written to require their own
removal in certain cases when you modify the work.) You may place
additional permissions on material, added by you to a covered work,
for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you
add to a covered work, you may (if authorized by the copyright holders of
that material) supplement the terms of this License with terms:

a) Disclaiming warranty or limiting liability differently from the
terms of sections 15 and 16 of this License; or

b) Requiring preservation of specified reasonable legal notices or
author attributions in that material or in the Appropriate Legal
Notices displayed by works containing it; or

c) Prohibiting misrepresentation of the origin of that material, or
requiring that modified versions of such material be marked in
reasonable ways as different from the original version; or

d) Limiting the use for publicity purposes of names of licensors or
authors of the material; or

e) Declining to grant rights under trademark law for use of some
trade names, trademarks, or service marks; or

f) Requiring indemnification of licensors and authors of that
material by anyone who conveys the material (or modified versions of
it) with contractual assumptions of liability to the recipient, for
any liability that these contractual assumptions directly impose on
those licensors and authors.

All other non-permissive additional terms are considered "further
restrictions" within the meaning of section 10. If the Program as you
received it, or any part of it, contains a notice stating that it is
governed by this License along with a term that is a further
restriction, you may remove that term. If a license document contains
a further restriction but permits relicensing or conveying under this
License, you may add to a covered work material governed by the terms
of that license document, provided that the further restriction does
not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you
must place, in the relevant source files, a statement of the
additional terms that apply to those files, or a notice indicating
where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the
form of a separately written license, or stated as exceptions;
the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly
provided under this License. Any attempt otherwise to propagate or
modify it is void, and will automatically terminate your rights under

this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or

hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this

License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

```
  If the disclaimer of warranty and limitation of liability provided
above cannot be given local legal effect according to their terms,
reviewing courts shall apply local law that most closely approximates
an absolute waiver of all civil liability in connection with the
Program, unless a warranty or assumption of liability accompanies a
copy of the Program in return for a fee.

                     END OF TERMS AND CONDITIONS

            How to Apply These Terms to Your New Programs

  If you develop a new program, and you want it to be of the greatest
possible use to the public, the best way to achieve this is to make it
free software which everyone can redistribute and change under these terms.

  To do so, attach the following notices to the program.  It is safest
to attach them to the start of each source file to most effectively
state the exclusion of warranty; and each file should have at least
the "copyright" line and a pointer to where the full notice is found.

    <one line to give the program's name and a brief idea of what it does.>
    Copyright (C) <year>  <name of author>

    This program is free software: you can redistribute it and/or modify
    it under the terms of the GNU General Public License as published by
    the Free Software Foundation, either version 3 of the License, or
    (at your option) any later version.

    This program is distributed in the hope that it will be useful,
    but WITHOUT ANY WARRANTY; without even the implied warranty of
    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
    GNU General Public License for more details.

    You should have received a copy of the GNU General Public License
    along with this program.  If not, see <http://www.gnu.org/licenses/>.

Also add information on how to contact you by electronic and paper mail.

  If the program does terminal interaction, make it output a short
notice like this when it starts in an interactive mode:

    <program>  Copyright (C) <year>  <name of author>
    This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
    This is free software, and you are welcome to redistribute it
    under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate
parts of the General Public License.  Of course, your program's commands
might be different; for a GUI interface, you would use an "about box".

  You should also get your employer (if you work as a programmer) or school,
if any, to sign a "copyright disclaimer" for the program, if necessary.
For more information on this, and how to apply and follow the GNU GPL, see
<http://www.gnu.org/licenses/>.

  The GNU General Public License does not permit incorporating your program
into proprietary programs.  If your program is a subroutine library, you
may consider it more useful to permit linking proprietary applications with
the library.  If this is what you want to do, use the GNU Lesser General
Public License instead of this License.  But first, please read
<http://www.gnu.org/philosophy/why-not-lgpl.html>.
```

3 Purpose of CFLIB

This library started as a Beginner Project in "C Library Building" following practical needs arising from scientific/technical modeling projects.

Most of the library code has been created in 1994/95 on an Atari ST4/16 MHz under TOS 2.05/MiNT 1.12 with gcc 2.5.8 and the Mintlibs Patchlevel 46. The C coding is probably not the best possible ;-) but the source as well as the executable and allocated memory structures are very compact, simple to modify and still fine for many applications that need a stable, portable and small configuration interface.

'Mission Statement' from 1994/95 README file:

> CFLIB is meant to be a flexible, sound and easy to use tool for C programmers. It provides a set of functions for a standard method for feeding a C program with all the (external) information it needs to perform as desired with a minimum of expense for both the programmer and the user: arguments, commands, program input, configurable and/or installation dependent features, system settings and a lot more can be passed to the program through different interfaces: command line, environment, configuration files or sections within them, interactive input and last but not least a built-in default. So it should be a fast and easy task for both the programmer and the user to build and change a configuration. The library will always take more space and perform slower than code that is written and optimized for a specific requirement, but this will only be a noticeable disadvantage in some cases. The library also provides some special features such as file search, time and date handling, generating simple, text template driven reports, etc. It is suitable for creating a comfortable and/or individual user interface for existing programs that don't have one. Some basic ideas for the library came from looking at the *termcap* library for easy and portable terminal I/O.

4 General Notes

4.1 The names "CFLIB", "libcf", "∗cf∗"

All "names" in this library source and documentation are nothing but technical acronyms just as "src", "inc", "C" and are **not** registered names, trademarks or anything like that ;-) You may change the names of the library, the include file and on compilation the function names and defines to whatever you like :-)

4.2 Input and Output

CFLIB is designed to be usable as a **Filter** program that reads from `stdin` and writes to `stdout` wherever it was desirable.

None of the library functions will produce any terminal output (to `stdout` or `stderr`) unless:

- The CF_QUERY flag is set in the Special Options Mask of an entry and the parameter value is still unassigned after parsing all other possible sources in cfinit().
- You call a function that reads from or writes to files with arguments that trigger the use of `stdin`, `stdout`, `stderr` etc. according to the function's documentation.

4.3 General (future) tasks

Todo

Make proper Man Pages with function references etc.

Make Doxygen Developer Documentation
More Modularization! Still very similar code in various functions

See also:

Bug List

4.4 Other general issues

- Library Versions

5 Simple Usage Example

```
/*
 *    Find a File in Path
 *
 *    Input:  Commandline or Environment
 *
 *    Output: File path, Help message or Error message
 *
 *    Compile: gcc fifi.c -o fifi -lcf -L../../lib<platform>
 *    Run:     ./fifi -h
 *    Test:    ./fifi gcc
 */

#include <stdio.h>

#include "../../include/cf.h"

int main( int argc, char **argv )
{
    int ret; char *tmp;

    CONFIG setting[] =
    {
        /* Name, Default Content, Commandline Option, Special Option Flags */
        { "PROGNAME", NULL,        '1', CF_FINDFILE|CF_SET_ARG|CF_NO_OPT_ARG, },
        { "FF_PATH",  NULL,        ' ', CF_PATH,                              },
        { "help",     CF_FLAG_OFF, 'h', CF_FLAG|CF_LAST,                      },
    };

    if ( !( ret = cfstart( setting, argc, argv, "help", CFS_USG ) ) ){

        if( ( tmp = cfgetstr("PROGNAME") ) != NULL )

            fprintf( stdout, "%s\n", tmp );
    }

    return ret;
}
```

6 Configuration Initializer

The **"Configuration Initializer"** is the major interface between the user program and CFLIB.

It is an Array of CONFIG items, just as the CFLIB Configuration Database itself.

Every parameter is characterized by:

- Parameter Name (CONFIG::name)
- Parameter Default Value (CONFIG::inhalt)
- Commandline Option for Parameter (CONFIG::option)
- Special Option Flags Mask for Parameter (CONFIG::flag)

An example initializer may look like this:

```
const CONFIG initializer[] = {
      // name         default         option  flag
    { "SPI_PROFILE", "~/.spi.cnf",   'c',    CF_SETFILE|CF_SET_ENV,      },
    { "SPI_SYSCONF", "/etc/spi.cnf",' ',    CF_SYS_SETFILE|CF_NOSAVE,   },
    { "SPI_DEFS",    "loop-0.A",     'f',    CF_SECTION,                 },
    { "SPI_ID",      "",             ' ',    CF_SET_PRIV,                },
    { "verbosity",   "1",            'v',    CF_INT|CF_CONCAT|CF_NOSAVE,},
    { "label_items", CF_FLAG_ON,     'l',    CF_FLAG,                    },
    { "label_date",  NULL,           ' ',    CF_DATE|CF_SET_PUT,         },
    { "label_logo",  NULL,           'L',    0,                          },
    { "output_file", NULL,           'o',    CF_CONCAT|CF_QUERY,         },
    { "TERM",        "0.2",          'T',    CF_STR|CF_IGN_ENV,          },
    { "SCALE",       "1",            'S',    CF_REAL|CF_IGN_ENV,         },
    { "LOOPMOD",     "random",       '1',    CF_NO_OPT_ARG,              },
    { "CYCLES",      "2009",         '2',    CF_INT|CF_NO_OPT_ARG,       },
    { "CF_DUMPVERB", CFD_COLHEAD,    ' ',    CF_INT|CF_NOSAVE|CF_LAST,   },
};
```

The Configuration Initializer controls the behavior and actions of all **cf***() functions, starting with the control of Parsing Levels in the Initialization Process.

6.1 Parameter Name

Every entry in the initializer must have a name, that is a non-empty string, which is searched for in the environment and in the configuration files and which is used as an argument to the inquiry functions. Parameters that have a name beginning with **"CF_"** may have special meaning, see CFLIB Parameters.

6.2 Parameter Default Value

As the default content of an entry you can specify a `NULL` pointer or a string, which may be empty. If CF_FLAG is set for that entry you should use the CF_FLAG_ON or CF_FLAG_OFF macros.

6.3 Commandline Option for Parameter

You can control whether and how an entry's content can be set from the Commandline by setting one of the following **"Option Specifier"** Characters (and Special Option Flags where indicated):

- '' (blank)
  - No Commandline Option for this Parameter
- '*c*' (*c* = any "normal" character)
  - Looks for '-c <*string*>' on the Commandline

  - Looks for '-c<*string*>' on the Commandline, if Flag CF_CONCAT is set in the Special Options Mask
  - Looks for '-c<*char*>' on the Commandline, if Flag CF_FLAG is set in the Special Options Mask

- '#' (# = a positive number = 1, 2, 3, ...)
  - Looks for the 1st, 2nd, 3rd, ... '<*string*>' that is not part of an option, if Flag CF_NO_OPT_ARG is set in the Special Options Mask

6.4 Special Option Flags Mask for Parameter

Most of the magic happens here! ;-) The **"Special Options Mask"** is a Bitmask of type CFFLAGTYP in Configuration Entry Structure Member CONFIG::flag containing Type, Instruction and Information Flags for a Parameter. See Special Options Mask for details.

See also:

Types of Parameters
Configuration Parsing Levels and Source/Origin Options

Attention:

The last entry in the initializer list must have the CF_LAST Flag set!

7 Configuration Parsing Levels and Source/Origin Options

CFLIB is designed to give you a maximum of possibilities for feeding parameters and parameter values into the configuration.

7.1 Origin Window

The library remembers the source/origin of all parameter values by setting one of the **"Source Flags"** in the **"Origin Window"** of the Special Options Mask ranging from Bit Offset 8 to 15 (see Parsing Levels). On initialization the Origin Window Flags **CF_SET_**$*$ of an entry in the Configuration Initializer are used to control Parsing Level Priorities.

7.2 Parsing Levels

"Parsing Levels" are equal to the Relative Bit Offset of the corresponding **CF_SET_**$*$ Bit within the "Origin Window":

Parsing Level : Description (Source Flag)

- 0 = CFP_PUT : Function Call or Automatic Initialization. (CF_SET_PUT)
- 1 = CFP_ARG : Commandline. (CF_SET_ARG)
- 2 = CFP_ENV : Environment. (CF_SET_ENV)
- 3 = CFP_PRIV : Private Configuration File. (CF_SET_PRIV)

- 4 = CFP_SYS : System Configuration File. (CF_SET_SYS)
- 5 : CFP_DEF : Built-in Default. (CF_SET_DEF)
- 6 : CFP_QRY : Standard Input Channel. (CF_SET_QRY)
- 7 : CFP_RESERVED : Reserved for CFLIB Subprojects

7.3 Parsing Levels in the Initialization Process

On initialization the Parsing Levels 1 through 6 are processed by cfinit() in that order and priority. If a Parameter is found that has been defined in the Configuration Initializer, its content is included as String Content of that parameter in the configuration database.

Whether and how the **Commandline** Arguments are used to get a parameter value depends on the CONFIG::option setting for that parameter and the Special Option Flags:

- CF_CONCAT : Argument is concatenated to option.
- CF_NO_OPT_ARG : Commandline argument not followig an option.

See details under Commandline Option for Parameter.

The **Environment** can be excluded as a possible source of a parameter value by setting the Special Option Flag

- CF_IGN_ENV : Do not look for Environment Variable

Configuration Files are only used when corresponding entries are defined in the Configuration Initializer, see Parsing of Configuration Files for details.

Interactive queries are not performed by default. If cfinit() shall ask the user for interactive input of items that are still unresolved after parsing all lower levels, you must set Special Option Flag:

- CF_QUERY : Ask the user for unresolved item after configuration parsing.

If an entry may not be empty (`NULL` or string "") after initialization, you can have cfinit() produce an appropriate error by setting the Special Option Flag:

- CF_MUST : Entry may not be empty (`NULL` or "").

7.3.1 Parsing Level Priorities

More precise control of parsing levels priorities and origin control for a parameter is available through setting one of the Flags in the "Origin Window" of the parameter's Special Options Mask in the Configuration Initializer :

- CF_SET_PUT : Must be set by a function call / automatic processing: Set this Flag on Time or Date Types of Parameters for automatic initialization with *now* or *today*
- CF_SET_ARG : Force initialization from commandline, ignore anything else
- CF_SET_ENV : Let environment variable override earlier setting

- CF_SET_PRIV : Let variable from private configuration file override earlier setting
- CF_SET_SYS : Let variable from system configuration file override earlier setting
- CF_SET_DEF : Let default from Configuration Initializer override earlier setting
- CF_SET_QRY : Force setting from interactive Query (`stdin`)

7.4 Parsing of Configuration Files

To indicate that an entry in the initializer list given to cfinit() represents the filename of one of the Configuration Files, set one of the following Flags for that parameter:

- CF_SETFILE : Entry is Private Configuration File.
- CF_SYS_SETFILE : Entry is System Configuration File.

Within the system configuration file you can choose a section, from which information is read as from a single file. To specify an entry in the list referring to the name of that section give this entry the flag

- CF_SECTION : Section in Configuration File.

As the filenames for the two configuration files are themselves entries in the database, these levels are revisited, if neccessary, in reverse order after parsing the built-in default.

7.5 Residual Items

Any commandline argument that did not match conditions for a parameter will be included in the configuration database as parameter without name marked with Flag CF_RESID in the Special Options Mask. These additional entries from the command line can be accessed by successive calls to cfgetres().

Residual items from one of the configuration files will also be accumulated in the database and can be accessed through inquiry by name with one of the Retrieval Functions. This is especially useful together with the report generation function cfform().

8 Configuration Files

A major task of the library is handling import and export of configuration parameters from/to files.

CFLIB knows two types of configuration files:

- **"Private Configuration File"** - User and/or program specific file in simple format to be read on initialization and optionally be updated by cfsave()
- **"System Configuration File"** - System and/or project specific file in extended format (supporting sections, see below) will only be used as a data source by cfinit() and will not be touched by cfsave() unless you explicitly specify the filename.

8.1 Configuration File Format

In a configuration file lines beginning with '#' are treated as comments and are ignored. Blank lines are ignored, too. A valid line in the file is of the form:

<name> = <entry>

Blank chars around the '=' are ignored. The name must match one of the entry's names in the initializer. In fact, any line not containing a '=' will be ignored, but it's better to indicate comments with '#' !

The optional sections in the system configuration file begin with a line like:

[*<sectionname>*]

and end with another line like this or with the file's last line. Anything after the closing bracket is ignored.

A simple example of a valid configuration file could look like this:

```
# This is my private configuration file for Project 1356 Branch C in spe

Search_Path  = /my/subproject/directory:/general/settings/directory
Section      = project_1356
Outfile      = my_subproject.cnf
ask_if_empty =
My_Flag =      ON
VERBOSITY = 1
```

The corresponding system configuration file could look like this:

```
# This is a system wide configuration file
[some_other_program]
blah = blubber
...
[project_1356]
# Settings for Project 1356 Branch B
VERBOSITY = 3
Outfile = project_1356b.cnf
X_EUR_USD=1.4562
[some_other_project]
...
```

8.2 Reading Configuration Files

A search for configuration files and import of data from these sources is only performed by the function cfinit() on initialization and only when appropriate Special Option Flags are set as described under Parsing of Configuration Files.

Configuration files are read once when the configuration database is initialized by a call to cfinit() or cfstart() using the internal function cfreadfile().

8.3 Writing a (private) Configuration File

Parameter export does not depend on any specific setting and can be performed whenever and as often as you like. For writing a configuration file call cfsave() with either the name of the file or NULL, in which case the current value of the "Private Configuration File" parameter will be used, if it exists. If an entry has the CF_NOSAVE flag set, it is excluded from saving. The "System Configuration File" may not be referred to directly.

Attention:

Writing **section** marks is **not** supported!

9 Types of Parameters

The internal format of a parameter's content CONFIG::inhalt is always a String (char Pointer).

The use and interpretation of parameters is controlled by a number of Flags in the Special Options Mask :

9.1 Parameter Data Types

- CF_STR : Entry is String. (**Default**).
- CF_INT : Entry is Integer.
- CF_REAL : Entry is Float.
- CF_FLAG : FLAG Parameter Type

9.2 Special Processing Instructions

Specific processing of a String Parameter's Value/Content in various CFLIB functions can be triggered by these Flags:

You can mark a parameter as Time **or** Date. These parameters will be initialized with *now* or *today* by cfinit() when Parsing Level Priorities are controlled by a CF_SET_PUT Flag in the Special Options Mask or on call of cfputtime():

- CF_TIME : Time string.
- CF_DATE : Date string.
- CF_FINDFILE : Entry is filename to be searched for in the path. FindFile() will be used on initialization to search the file in the **path** found in Environment Variable **PATH**. You can specify an alternative search path for this functionality in another database parameter with Flag CF_PATH set in the corresponding Configuration Initializer entry. Depending on the Operating System a list of possible of **extensions** of executables will be tried, if no extension is given with the filename:
  - Linux, Unix: "sh", "pl"
  - DOS,Windows, OS2: "exe", "com", "bat", "cmd"
  - TOS, MinT: "ttp", "tos", "prg", "app", "gtp"
- CF_EXPHOME : Expand Home Directory. See cfhomexp(). Configuration Files will have home directory expanded by default

9.3 Special Handling Instructions

A parameter can be marked as a **"volatile"** entry that is not saved by cfsave():

- CF_NOSAVE : Don't include in savefile / mark entry.

9.4 CFLIB Parameters

Some String Parameters in the configuration database are used by the library itself and are marked by one of the following Flags:

- CF_PRGNAME : Running Program's Name from commandline. This parameter **always** exists as the **first** entry in a configuration database successfully initialized by cfinit(). The default parameter name used if there was no corresponding entry in the Initializer is **"CF_PRGNAME"**.
- CF_SETFILE : Entry is Private Configuration File. See Configuration Files
- CF_SYS_SETFILE : Entry is System Configuration File. See Configuration Files
- CF_SECTION : Section in Configuration File. See Configuration Files
- CF_PATH : Search Path (for FindFile feature). See CF_FINDFILE and FindFile()
- CF_USAGE : Usage Message format string. See cfgetusg()

Optional Parameters for fine tuning CFLIB behavior without Special Option Flag, characterized by Parameter Name:

- **"CF_DUMPVERB"** - Non-Default Verbosity Mode for cfdump()

9.5 FLAG Parameter Type

The entry is treated like a boolean variable and the string content set from any of the Configuration Parsing Levels and Source/Origin Options except the default setting is interpreted as follows:

- CF_FLAG_OFF : String Content is:
  - beginning with a `'-'`
  - `"OFF"` or `"off"`
  - `"FALSE"` or `"false"`
- CF_FLAG_ON : any other case

Example:

- Initializer entry: { "extended_message", CF_FLAG_OFF, 'x', CF_FLAG, }
- Commandline: myprog -x-
- Configuration file: extended_message = OFF

10 Advanced Usage Example

Setup Configuration Initializer, generate Report and save Configuration Files

```
/*
 *    Generate Report from Template and Configuration
 *
 *    Input:  - Commandline, Environment, Section in System Configuration File,
 *              Private Configuration File, Standard Input for Variables.
 *            - Text Template (Form).
```

```
 *
 *    Output: - Generated Report (Filled Form) to File or stdout,
 *            - Configuration to Private Configuration File, Logfile or
 *              Standard Output
 */

#include <stdio.h>

#include "../../include/cf.h"

#define DEF_SETFILE "~/fill.cnf"
#define DEF_SYS_SETFILE "/etc/cfcommon.cnf"

int main( int argc, char **argv )
{
    int ecode;
    char *save, *infile, *outfile, line[CF_MAXERRSTR+1];
    char *savemode = "w";
    FILE *error_log = stderr;

    /* Hardcoded Configuration Setup and Defaults */
    CONFIG setting[] =
    {   { "FILL_INFILE",   NULL,             'i', 0x0,                        },
        { "FILL_OUTFILE",  NULL,             'o', 0x0,                        },
        { "FILL_VARDELIM", "$()",            'l', CF_CONCAT,                  },
        { "verb",          CF_FLAG_OFF,      'v', CF_IGN_ENV|CF_FLAG,         },
        { "help",          CF_FLAG_OFF,      'h', CF_IGN_ENV|CF_FLAG|CF_NOSAVE,},
        { "query",         CF_FLAG_OFF,      'q', CF_IGN_ENV|CF_FLAG,         },
        { "TIME",          NULL,             't', CF_TIME|CF_SET_PUT,         },
        { "DATE",          NULL,             'd', CF_DATE|CF_SET_PUT,         },
        { "FILL_CNF",      DEF_SETFILE,      'p', CF_SETFILE,                 },
        { "FILL_SAVE",     NULL,             's', CF_IGN_ENV|CF_CONCAT,       },
        { "SAVE_DATA",     CF_FLAG_OFF,      'x', CF_FLAG|CF_IGN_ENV,         },
        { "SFILE",         DEF_SYS_SETFILE,  ' ', CF_SYS_SETFILE,             },
        { "FILL_SECTION",  "fill_default",   'c', CF_SECTION|CF_LAST,         },
    };

    /* Initialize with Error Reporting */
    if( cfinit(setting,argc,argv) < 0 ){
        while( (ecode=cfgeterr(line,0)) == TRUE )
            fprintf( error_log, "error %d: %s", ecode, line );
        fflush(error_log);
        fputs( cfgetusg(), stderr );
        return (-1);
    }

    /* User Help and Configuration Overview */
    if( cfgetflag("help") ){
        fputs( cfgetusg(), stderr );
        if( cfgetflag("verbose") ){
            fprintf( stderr, "\nConfiguration Library PL %d - %s\n\nCurrent Configuration:\n\n",
                             cfgetvers(), cfgetcpr() );
            cfdump(stderr);
            fputs( "\nSources: 1=update function call, 2=commandline argument, 4=environment variable,\n",
            fputs(   "         8=private setfile, 10=system setfile, 20=built-in default, 40=interactive t
        }
        return (1);
    }

    switch(
        cfform( infile = cfgetstr("FILL_INFILE"),
                outfile = cfgetstr("FILL_OUTFILE"),
                cfgetstr("FILL_VARDELIM"),
                cfgetflag("query") )
    ){
        case CFE_FNF:
            fprintf(stderr,"read access error: %s\n",infile); break;
```

```
        case CFE_WAE:
            fprintf(stderr,"write access error: %s\n",outfile); break;
    }

    if( cfgetflag("verb") ){
        if( outfile == NULL ) outfile = "<stdout>";
        fprintf( stderr, "blank form: %s, filled form: %s\n", infile, outfile );
    }

    if( cfgetflag("SAVE_DATA") ){
        cfnosave(NULL, CF_FLAG_ON);
        cfnosave("TIME", CF_FLAG_OFF);
        cfnosave("DATE", CF_FLAG_OFF);
    }
    if( (save = cfgetstr("FILL_SAVE")) != NULL ){
        if( cfgetflag("SAVE_DATA") ) savemode = "a";
        if( *save == '\0' )
            cfsave( NULL, savemode );
        else if( *save == '+' )
            cfsave( "", savemode );
        else
            cfsave( save, savemode );
    }

    return (0);
}
```

11 Compilation and Development

11.1 Library Versions

There is no strict versioning of the library. A Library **"Patchlevel"** is defined as an Integer Number counting **"Major Versions"**. See cfgetvers().

The individual source files contain more detailed version information and they all support a file version identifier `$Id$ for` automatic update by SVN and other Source Management Tools. See cfgetsubvers().

11.2 Platforms

The library source code is fairly simple ANSI C code and should compile and link without problems on most platforms. Most of the development has been done with different versions of the **"GNU C Compiler"** `gcc` and related tools. All source modules compile free of errors and warnings with:

`"gcc -pedantic -pedantic-errors -Wall -Werror -ansi ..."`

So, if you have `gcc`, use it! Any other ANSI C compiler should also work, at least after minimal adaption to the library setup, see System and Compiler dependent Defines.

until now the library has been built and used under the following setups:

- gcc, Linux, Intel PC
- gcc, MinGW, MS Windows NT/XP, Intel PC
- gcc, MinT/TOS Atari ST
- cc, Unix, IBM AIX
- cc, Unix, SGI

- MSC, MS DOS/Windows Intel PC
- MSC, OS/2 Intel PC

11.3 Building the Library

Compiling the source modules and building the library should be a "straight forward" task:

- Make all objects from C sources, including *cf.h* and *cflib.h*
- Link the objects with `ar` or another tools to get a library executable

See also:

Makefiles under CFLIB project tree and System and Compiler dependent Defines

11.4 Binaries and Executables

- The library project should be seen as "pure" source/text code on distribution. Compile a library executable with a C compiler of your choice with appropriate setup for your platform.

- Binary and executable versions of the library found in the project tree should be seen as examples that worked under one specific setup but have not been intensively tested and may even not be up to date. If it works for you, feel free to use them. If you build a library executable which is not too ;-) dependent on a specific setup, you can include it as example in the project tree.

11.5 Minimal CFLIB Replacement

In project directory *src/examples* you may find a source file *cf_minimal.c*. You can adapt this example to your need and include it in your C source list on compilation instead of linking the library executable. This makes user programs that use CFLIB functions independent of the library at the price of very reduced functionality, which may be desirable for specific executables or if you have problems with building the library on the target platform.

11.6 System and Compiler dependent Defines

The following Defines are used in the library source to decide whether to include certain header files and use certain functions or defines:

- **_HAS_PWD** : If defined, include <**pwd.h**> and make use of function **getpwnam()** in cfhomexp() to find a user's home directory
- **_HAS_ISATTY** : If defined, use function **isatty()** to determine whether a stream is a terminal (for interactive query/input) in function IsATerminal()
- **_HAS_LIMITS** : If defined, include <**limits.h**> and use **PATH_MAX** defined therein in function FindFile()

The following Defines can be used to control, which features or functions shall be excluded from the library build:

- **_CF_NOFINDFILE** : Function FindFile(), component findfile.c
- **_CF_NOSAVING** : Functions cfsave() and BackupFile(), component cfwrite.c
- **_CF_NODEBUGGING** : Functions cfdinichk() and cfdump(), component cfdebug.c

The following Defines can be used to switch on certain Debug Features:

- **DEBUG_DINICHK** : Debug cfdinichk()
- **DEBUG_ERROR** : Debug cfputerr()
- **DEBUG_TIME** : Debug cfputtime()
- **DEBUG_NOSAVE** : Debug cfsave()
- **DEBUG_BACKUP** : Debug BackupFile()
- **DEBUG_FINDFILE** : Debug FindFile() usage in cfinit()
- **DEBUG_FORM** : Debug cfform()

The following Defines can be used to switch on certain other Features:

- **_PREFER_BACKSLASH** : Prefer Backslash as Directory Separator in function FindFile()
- **_PATHSEP_SEMICOLON** : Use Semicolon as Path Separator in function FindFile(), usually on systems where a ":" can appear in a directory path
- **_PATHSEP_COMMA** : Test Comma as Alternative Path Separator in function FindFile()
- **_CF_RESID_FREE** : Remove residual arguments after having read them all in cfgetent()?

The following Platform dependent Defines are used in the library code:

- **unx**
- **linux**
- **atarist**
- **__MINT__**

12 Bug List

Group errors There are still errors without entry in error stack

Global IsATerminal ANSI C doesn't have function isatty(), we always return TRUE

13 Todo List

Global cfdinichk Make cfdinichk() work as reliable, complete tool with much more testing and Special Options Mask validation!

Global cfform Make cfform() work with buffers instead of files

Page General Notes Make proper Man Pages with function references etc.

Make Doxygen Developer Documentation

More Modularization! Still very similar code in various functions

14 Module Index

14.1 Modules

Here is a list of all modules:

CFLIB Core Features 29

Special Options Mask 33

Error Handling 35

Advanced Features 40

Information Retrieval 42

Setting and Saving the Configuration 48

General Utilities 50

Report Generation 53

15 Data Structure Index

15.1 Data Structures

Here are the data structures with brief descriptions:

CONFERR (Library Internal: Error List Item) 54

CONFIG (CFLIB Configuration Database Entry) 55

16 File Index

16.1 File List

Here is a list of all documented files with brief descriptions:

include/cf.h (C Header File for CFLIB Flexible Configuration Library) 55

17 Module Documentation

17.1 CFLIB Core Features

Basic CFLIB Setup.

Data Structures

- struct CONFIG

  CFLIB Configuration Database Entry.

Defines

- #define Patchlevel "20"

  CFLIB Identification.

- #define MAXCONF 4096

  Maximum number of entries in configuration database.

- #define CF_MAXERRSTR 512

  Maximum string length for error message.

- #define CF_MAXLINE 20480

  Maximum string length for setfile and form parsing.

- #define CF_MAXQLINE 512

  Maximum string length for query.

- #define CF_MAXUSAGE 1024

  Maximum string length for usage string.

- #define CF_USG_DEFCOLS 80

  Default terminal width for usage string.

- #define CF_MAXTIMEBUF 256

  Buffer size for time and day.

- #define TRUE 1

  TRUE, if not defined.

- #define FALSE 0

  FALSE, if not defined.

- #define NULL (void *)(0L)

  NULL, if not defined.

- #define CF_FLAG_ON "\1"
  Flag is set.
- #define CF_FLAG_OFF ""
  Flag is not set.
- #define CF_NO_OPTION ' '
  Option is not set.
- #define TABLEN 8
  TAB length.
- #define CFP_PUT 0
  Function Call or Automatic Initialization.
- #define CFP_ARG 1
  Commandline.
- #define CFP_ENV 2
  Environment.
- #define CFP_PRIV 3
  Private Configuration File.
- #define CFP_SYS 4
  System Configuration File.
- #define CFP_DEF 5
  Built-in Default.
- #define CFP_QRY 6
  Standard Input Channel.
- #define CFP_RESERVED 7
  Reserved for Subprojects.
- #define CFS_NOT 0
  Start Mode: No action on error.
- #define CFS_ALL 1
  Start Mode: All error messages.
- #define CFS_NEG 2
  Start Mode: Only severe errors.
- #define CFS_USG 3
  Start Mode: Usage message if error was negative, error output like CFS_NEG.
- #define CFS_DEBUG 4
  Start Mode: Output like CFS_USG plus raw dump of configuration.

Functions

- int cfinit (CONFIG *set, int ac, char **av)

  Initialize CFLIB Configuration Database and parse possible sources for database entries according to the settings in set.

- void cfexit (void)

  Free allocated memory and reset the configuration database and error stack.

- int cfstart (CONFIG *setting, int ac, char **av, char *help, int mode)

  Setup Configuration Database.

Variables

- CONFIG * _conf [MAXCONF+1]

  Library Internal: CFLIB Configuration Database with at most MAXCONF parameters.

17.1.1 Detailed Description

Basic CFLIB Setup.

- General Defines
- Parsing Levels
- Configuration Entries (aka Parameters)
- Configuration Database: Array of Configuration Entries
- Initialization of the Configuration Database
- Resetting the Configuration Database

17.1.2 Function Documentation

17.1.2.1 int cfinit (CONFIG * *set*, int *ac*, char ** *av*)

Initialize CFLIB Configuration Database and parse possible sources for database entries according to the settings in *set*.

Parameters:

set Configuration Initializer : Pointer to Array of CONFIG items containing parameter name, default value, Commandline Option for Parameter and Special Options Mask

ac Argument Count from commandline or compatible

av Argument String array from commandline or compatible structure

Returns:

- 0 : configuration database initialized successfully

- >0 : Count of non fatal errors
- <0 : Fatal error(s) occurred:
  - CFE_MCF : Memory allocation for Configuration Failed.
  - CFE_MEF : Memory allocation in Error routine Failed.
  - any other absolute value is total error count

See also:

Configuration Parsing Levels and Source/Origin Options

17.1.2.2 void cfexit (void)

Free allocated memory and reset the configuration database and error stack.

This function should be used before a repeated call to cfinit() or cfstart(). Furthermore, it may be desired to call cfexit() when the configuration database occupies a lot of memory and is no longer needed.

17.1.2.3 int cfstart (CONFIG * *setting*, int *ac*, char ** *av*, char * *help*, int *mode*)

Setup Configuration Database.

Check errors and output usage message, if required

Parameters:

setting pointer to initializer

ac argument count from commandline or compatible

av argument string array from comandline or compatible

help Help/Usage String

- Name of a FLAG Parameter Type entry to cause a usage message, if flag is set
- Omit, if NULL

mode How to handle cfinit() return:

- CFS_NOT - Start Mode: No action on error.
- CFS_ALL - Start Mode: All error messages.
- CFS_NEG - Start Mode: Only severe errors.
- CFS_USG - Start Mode: Usage message if error was negative, error output like CFS_NEG.
- CFS_DEBUG - Start Mode: Output like CFS_USG plus raw dump of configuration.

Returns:

- 0 : configuration database has been initialized successfully, no help or usage message required, no fatal errors
- 1 : no fatal errors occurred, usage message displayed
- -1 : fatal error(s) occurred

17.2 Special Options Mask

The **"Special Options Mask"** is a Bitmask of type CFFLAGTYP in Configuration Entry Structure Member CONFIG::flag containing Type, Instruction and Information Flags for a Parameter.

Defines

- #define CF_LAST 0x0001

  Last entry in settings array.

- #define CF_PATH 0x0002

  Search Path (for FindFile feature).

- #define CF_SETFILE 0x0004

  Entry is Private Configuration File.

- #define CF_SYS_SETFILE 0x0008

  Entry is System Configuration File.

- #define CF_SECTION 0x0010

  Section in Configuration File.

- #define CF_PRGNAME 0x0020

  Running Program's Name from commandline.

- #define CF_TIME 0x0040

  Time string.

- #define CF_DATE 0x0080

  Date string.

- #define CF_SET_PUT 0x0100

  Source: Function Call.

- #define CF_SET_ARG 0x0200

  Source: Commandline Argument.

- #define CF_SET_ENV 0x0400

  Source: Environment Variable.

- #define CF_SET_PRIV 0x0800

  Source: Private Configuration File.

- #define CF_SET_SYS 0x1000

  Source: System Configuration File.

- #define CF_SET_DEF 0x2000

  Source: Built-in Default.

- #define CF_SET_QRY 0x4000

Source: Interactive Terminal Input.

- #define CF_NO_OPT_ARG 0x10000

  Commandline argument not followig an option.

- #define CF_CONCAT 0x20000

  Argument is concatenated to option.

- #define CF_IGN_ENV 0x40000

  Do not check environment for variable.

- #define CF_QUERY 0x80000

  Ask the user for unresolved item after configuration parsing.

- #define CF_STR 0x100000

  Entry is String.

- #define CF_INT 0x200000

  Entry is Integer.

- #define CF_FLAG 0x400000

  Entry is Flag.

- #define CF_REAL 0x800000

  Entry is Float.

- #define CF_FINDFILE 0x1000000

  Entry is filename to be searched for in the path.

- #define CF_MUST 0x2000000

  Entry may not be empty (`NULL` or "").

- #define CF_RESID 0x4000000

  Residual/additional entry from commandline/setfile/cfput.

- #define CF_USAGE 0x8000000

  Usage Message format string.

- #define CF_EXPHOME 0x10000000

  Expand Home Directory.

- #define CF_MALLOC 0x20000000

  Space for entry's content was mallocated.

- #define CF_FORCED 0x40000000

  Setting has been forced (already).

- #define CF_NOSAVE 0x80000000

  Don't include in savefile / mark entry.

- #define CF_SRC (CF_INT|CF_FLAG)

  Type for source/origin inquiry.

- #define CF_FLGINQ (CF_STR|CF_FLAG)

  Type for options mask inquiry.

- #define CF_TD (CF_DATE|CF_TIME)

  Date or Time entry.

Typedefs

- typedef unsigned long CFFLAGTYP

  Special Options Mask Type.

17.2.1 Detailed Description

The **"Special Options Mask"** is a Bitmask of type CFFLAGTYP in Configuration Entry Structure Member CONFIG::flag containing Type, Instruction and Information Flags for a Parameter.

- Special CFLIB properties: CFLIB Parameters
- Source/Origin: Parsing Levels
- Initialization options: Parsing Levels in the Initialization Process
- Type/Interpretation Flags: Parameter Data Types
- Special Processing Instructions
- Special Handling Instructions
- Information/Status markers:
  - CF_LAST must appear in the Configuration Initializer
  - CF_RESID marks one of the Residual Items
  - CF_MALLOC and CF_FORCED are for library internal use

17.3 Error Handling

Error Codes, Functions and Structures.

Data Structures

- struct CONFERR

  Library Internal: Error List Item.

Defines

- #define CFE_INIT 0

  INITialize error input.

- #define CFE_OK 0

  No error / everything OKay.

- #define CFE_NEP 1

  New Entry successfully Put into DB.

- #define CFE_EXIT 1

  Finish error input.

- #define CFE_ORA 20

  Option Requires an Argument.

- #define CFE_UKO 25

  UnKnown Option.

- #define CFE_FNF 30

  File Not Found, read access error.

- #define CFE_NSE 40

  No Section specifier End found, missing "]".

- #define CFE_NSC 50

  No private Setfile Configured.

- #define CFE_WAE 60

  Write Access Error.

- #define CFE_IFP 61

  Invalid Filename entry for Private setfile.

- #define CFE_EWN 70

  Entry Without Name.

- #define CFE_ICF 80

  Invalid Combination of Flags.

- #define CFE_EWC 90

  Entry Without Content.

- #define CFE_UOS 100

  Unlikely Option Specifier.

- #define CFE_IFC 110

  Invalid Flag Combination.

- #define CFE_NLE 120
  No Last Entry flag found.
- #define CFE_TIN 130
  Error reTurn from stdIN query.
- #define CFE_EFE 140
  Empty string in content for Filename Entry.
- #define CFE_USG 200
  Entries missing: USaGge advice.
- #define CFE_URI 210
  UnResolved Item (CF_MUST was set!).
- #define CFE_FBF -500
  File Backup Failed.
- #define CFE_BMF -510
  Backup: Memory allocation Failed.
- #define CFE_BOF 520
  Backup: Open source file Failed.
- #define CFE_BBF 530
  Backup: open target Backup file Failed.
- #define CFE_BRF 540
  Backup: Rename Failed.
- #define CFE_BWF 550
  Backup: Write Failed.
- #define CFE_NCA -10
  No Configuration database Available.
- #define CFE_NEA -20
  No Entry with that name Available.
- #define CFE_NSS -30
  No Source/origin is Set.
- #define CFE_ECP -40
  Entry's Content is a NULL Pointer.
- #define CFE_MEF -100
  Memory allocation in Error routine Failed.
- #define CFE_MCF -200
  Memory allocation for Configuration Failed.

- #define CFE_INF -9999

  Integer iNquiry Failed (?!).

- #define CFE_RNF -999.999

  Real/float iNquiry Failed (?!).

Functions

- int cfputerr (int ecode, char *string,...)

  Init, exit or append to Error List.

- int cfgeterr (char *string, size_t len)

  Error Code and Message Inquiry Function.

- void cfclearerr (void)

  Free all entries in error list.

- int cfreverr (void)

  Revert order of entries in error list from last->first to first->last.

Variables

- CONFERR * _cferr = NULL

  Library Internal: Error List Pointer.

- int _errcnt = 0

  Library Internal: Error List Counter.

17.3.1 Detailed Description

Error Codes, Functions and Structures.

- Error Codes and their mnemonic descriptions

Bug

There are still errors without entry in error stack

- Library Internal Error Variables

- Error Functions

17.3.2 Function Documentation

17.3.2.1 int cfputerr (int *ecode*, char * *string*, ...)

Init, exit or append to Error List.

Parameters:

ecode Error Code of the error that occurred

string Error message format string. Giving it a `NULL` pointer results in an empty error message string

... Arguments list according to format string

Returns:

- `<0` : A fatal error occurred (malloc failed)
- `>0` : Number of errors in error list

17.3.2.2 int cfgeterr (char * *string*, size_t *len*)

Error Code and Message Inquiry Function.

Parameters:

string Pointer to a string, to which the error message should be copied. Giving it a `NULL` pointer will omit message return

len Size of string, if `0` CF_MAXERRSTR will be used

Returns:

- `0` : if no error is available, everything is alright
- `!=0` : error code of the next error in list

17.3.2.3 void cfclearerr (void)

Free all entries in error list.

17.3.2.4 int cfreverr (void)

Revert order of entries in error list from last->first to first->last.

Returns:

- `>=0` : Number of errors in error list
- `<0` : Inconsistency with old error count, absolute value is new error counter

17.4 Advanced Features

Debugging and Utility Functions.

Defines

- #define CFD_CFDUMP 0
  Dump Mask Minimal.
- #define CFD_LIBHEAD 1
  Dump option CFLIB header.
- #define CFD_COLHEAD 2
  Dump option Column headers.
- #define CFD_SRCFLAGS 4
  Dump option Source flag description.
- #define CFD_DEFAULT CFD_COLHEAD|CFD_SRCFLAGS
  Dump Mask Default.

Functions

- int cfdump (FILE *fout)
  Debugging Function.
- int cfdinichk (CONFIG *set)
  *Debugging Function (***experimental***).*
- char * cfhomexp (char *name)
  Expand ~ or ~`user` in parameter content.

17.4.1 Detailed Description

Debugging and Utility Functions.

- Modes for cfdump()

17.4.2 Function Documentation

17.4.2.1 int cfdump (FILE * *fout*)

Debugging Function.

Dump CFLIB DB content to `fout`

Configuration options:

Set integer bitmask CFLIB variable **"CF_DUMPVERB"** to

- = CFD_CFDUMP : Minimal
- & CFD_LIBHEAD : CFLIB header
- & CFD_COLHEAD : Column headers
- & CFD_SRCFLAGS : Source flag description
- = CFD_DEFAULT : Default dump verbosity

Parameters:

fout Pointer to File opened for writing or `stdout/stderr/...`

Returns:

- `>0` : Number of entries in CFLIB DB
- `<0` : An error occurred:
  - CFE_NCA : No configuration database available

17.4.2.2 int cfdinichk (CONFIG * *set*)

Debugging Function (**experimental**).

Initialize CFLIB DB using the default given by `set`, checking validity and plausibility of entries

Parameters:

set Pointer to initializing CONFIG-Array

Returns:

- `0` : No error occurred
- `!=0` : An error occurred

Todo

Make cfdinichk() work as reliable, complete tool with much more testing and Special Options Mask validation!

17.4.2.3 char* cfhomexp (char * *name*)

Expand ~ or ~`user` in parameter content.

This function is used by default when reading Configuration Files and on initialization of parameters with the Special Option Flag CF_EXPHOME set.

- The environment is checked for the variables `LOGNAME` or `USER`, if no user name is given ("~/....")

- The `passwd` file is searched for the users home directory, if possible
- Otherwise the environment variable `HOME` is checked
- If all that fails, ~ will be omitted, `~user` will expand to "./user"

Parameters:

name Entry's name

Returns:

- `NULL` : an error occurred:
  - No Configuration database Available.
  - Entry's Content is a NULL Pointer.
  - Memory allocation for Configuration Failed.
- `!=NULL` : String pointer to original or expanded filename

17.5 Information Retrieval

These functions and macros read entries from an initialized CFLIB database.

Defines

- #define cfget(a) cfgetent(a,0)

  Get value (content) of named entry.

- #define cfgetstr(a) ((char *)cfgetent(a,CF_STR))

  Inquire CFLIB DB for String in content of named entry.

- #define cfgetnum(a) (*(int *)cfgetent(a,CF_INT))

  Inquire CFLIB DB for Integer value in content of named entry.

- #define cfgetreal(a) (*(float *)cfgetent(a,CF_REAL))

  Inquire CFLIB DB for Float (Real) value in content of named entry.

- #define cfgetflag(a) (*(int *)cfgetent(a,CF_FLAG))

  Inquire CFLIB DB for Flag value in content of named entry.

- #define cfflaginq(a, b) (*(int *)cfgetent(a,CF_FLGINQ|(31&b)))

  Inquire CFLIB DB for Bit set in entry's Special Options Fag CONFIG::flag.

- #define cfgetsrc(a) (*(int *)cfgetent(a,CF_SRC))

  Inquire CFLIB DB for Source of named entry's content.

- #define cfgetres() ((char *)cfgetent("",CF_RESID))

  Get next Residual Command Line Argument from CFLIB DB.

- #define cfgetcpr() "CFLIB (c) 1994-2009 Stefan Habermehl"

  Get Copyright Notice.

Functions

- void * cfgetent (char *name, CFFLAGTYP typ)
  Library internal function, use appropriate Macro functions!
- int cfgetvers (void)
  Get Library Version/Patchlevel.
- char * cfgetsubvers (void)
  Get Library Subversion Details.
- char * cfgetusg (void)
  Get Usage Message for (Terminal) Output.

17.5.1 Detailed Description

These functions and macros read entries from an initialized CFLIB database.

The exact name of the required parameter must be given as argument, where indicated.

17.5.2 Define Documentation

17.5.2.1 #define cfget(a) cfgetent(a,0)

Get value (content) of named entry.

Parameters:

a Entry's name

Returns:

Depending on configured type, see cfgetent() and Macro Definitions!

17.5.2.2 #define cfgetstr(a) ((char *)cfgetent(a,CF_STR))

Inquire CFLIB DB for String in content of named entry.

Parameters:

a Entry's name

Returns:

- `NULL` : An error occurred:
  - No configuration database available
  - No entry of this name available
  - Content is really `NULL`, Check that with cfgetflag() !!
- Any other : Pointer to string in content of entry name

17.5.2.3 #define cfgetnum(a) (∗(int ∗)cfgetent(a,CF_INT))

Inquire CFLIB DB for Integer value in content of named entry.

Parameters:

a Entry's name

Returns:

- CFE_INF : Integer inquiry failed because of
  - No configuration database available
  - No entry of this name available
  - Content doesn't begin with digit
  - Content is a `NULL` pointer
  - Content is really CFE_INF, Check that with cfgetstr() !!
- Any other : Integer value for named entry

17.5.2.4 #define cfgetreal(a) (∗(float ∗)cfgetent(a,CF_REAL))

Inquire CFLIB DB for Float (Real) value in content of named entry.

Parameters:

a Entry's name

Returns:

- CFE_RNF : Real/float value inquiry failed because of
  - No configuration database available
  - No entry of this name available
  - Content doesn't begin with digit or signum (+/-)
  - Content is a `NULL` pointer
  - Content is really CFE_RNF, Check that with cfgetstr() !!
- Any other : Float value for named entry

17.5.2.5 #define cfgetflag(a) (∗(int ∗)cfgetent(a,CF_FLAG))

Inquire CFLIB DB for Flag value in content of named entry.

Parameters:

a Entry's name

Returns:

- `TRUE` (1) : Flag is set

- FALSE (0) : Flag is not set
- <0 : An error occurred:
  - CFE_NCA : No configuration database available
  - CFE_NEA : No entry of this name available
  - CFE_ECP : Entry's content is a NULL pointer

17.5.2.6 #define cfflaginq(a, b) (*(int *)cfgetent(a,CF_FLGINQ|(31&b)))

Inquire CFLIB DB for Bit set in entry's Special Options Fag CONFIG::flag.

Debugging Function

Parameters:

a Entry's name

b Bit Offset, 0<=b<=31

Returns:

- TRUE (1) : Flag is set
- FALSE (0) : Flag is not set
- <0 : An error occurred:
  - CFE_NCA : No Configuration database Available.
  - CFE_NEA : No Entry with that name Available.

17.5.2.7 #define cfgetsrc(a) (*(int *)cfgetent(a,CF_SRC))

Inquire CFLIB DB for Source of named entry's content.

Parameters:

a Entry's name

Returns:

- 0-6 : Source of entry's content:
  - 0 : cfputstr() call
  - 1 : Command line / Arguments
  - 2 : Environment
  - 3 : Private Configuration File
  - 4 : System Configuration File
  - 5 : Default setting
  - 6 : Interactive input (query)
- <0 : An error occurred:
  - CFE_NCA : No configuration database available
  - CFE_NEA : No entry of this name available
  - CFE_NSS : No source set (should not happen!)

17.5.2.8 #define cfgetres() ((char *)cfgetent("",CF_RESID))

Get next Residual Command Line Argument from CFLIB DB.

Returns:

- `NULL` : An error occurred:
  - No Configuration database Available.
  - No more residual arguments available
- Any other : Pointer to string content

See also:

Residual Items

17.5.2.9 #define cfgetcpr() "CFLIB (c) 1994-2009 Stefan Habermehl"

Get Copyright Notice.

Returns:

Pointer to Copyright Message String

17.5.3 Function Documentation

17.5.3.1 void* cfgetent (char * *name*, CFFLAGTYP *typ*)

Library internal function, use appropriate Macro functions!

Inquire configuration database for content of entry name

Parameters:

name Entry's name

typ Expected/required type of content:

- `0` : Get type from entry's flag
- CF_INT : Integer
- CF_REAL : Real
- CF_FLAG : Flag
- CF_STR : String
- CF_SRC : Source
- CF_FLGINQ : Bit in entry flag (bit no. in lowest bytes)
- CF_RESID : Residual argument
- Anything else : String

Returns:

- for CF_STR, CF_RESID or default:

- NULL : An error occurred or nothing available:
  - No configuration database available
  - No entry of this name available
  - The entry's content is really NULL, Check that with cfgetflag() !!
  - No more residual argument (for CF_RESID)
- any other pointer to string in content of entry name
- for others: Pointer to return values of the corresponding macro function

17.5.3.2 int cfgetvers (void)

Get Library Version/Patchlevel.

Returns:

- >0 : Libary Patchlevel
- <=0 : Error

17.5.3.3 char* cfgetsubvers (void)

Get Library Subversion Details.

The Patchlevel returned by this function should match Patchlevel in the public include file cf.h

Returns:

Libary Patchlevel and Subversion (Source Revision marked by library internal header file *cflib.h*.

The return value is "burned" into the library executable and looks like:

```
CFLIB PL 20 $LastChangedRevision: 22 $
```

17.5.3.4 char* cfgetusg (void)

Get Usage Message for (Terminal) Output.

There are two flavours:

1. Let CFLIB do the job: Usage message is generated based on settings for commandline parsing and Special Options Mask found in the database

2. Deliver your own Usage Message: Just set the CF_USAGE Flag in the Special Options Mask of one parameter in the database to get a custom usage message (from hardcoded default, configuration file or environment). The delivered custom message string is taken as a format string for the printf() function: Use "%s" in the message string to have the program name inserted that CFLIB got from the default parameter **"CF_PRGNAME"** which by default is set to the name of the running program from the commandline at startup.

Returns:

- Pointer to usage string
- NULL : An error occurred:
  - No Configuration database Available.
  - malloc() for usage string failed

See also:

CF_MAXUSAGE : Maximum string length for usage string.
CF_USG_DEFCOLS : Default terminal width for usage string.

17.6 Setting and Saving the Configuration

Set/Update Parameter Values or Save a Configuration File.

Functions

- int cfputstr (char *name, char *content)
  Update or Add Parameter name *with string* content.
- int cfputtime (CFFLAGTYP td)
  Set all Time and/or Date entries in CFLIB DB to now *or* today.
- int cfnosave (char *name, const char *onoff)
  Alter or query the CF_NOSAVE Flag of Parameter name.
- int cfsave (char *file, const char *savemode)
  Write configuration data to a Configuration File or stdout.

17.6.1 Detailed Description

Set/Update Parameter Values or Save a Configuration File.

17.6.2 Function Documentation

17.6.2.1 int cfputstr (char * *name*, char * *content*)

Update or Add Parameter *name* with string *content*.

Parameters:

name Parameter Name

content New (String) Content

Returns:

- CFE_NEP : New Entry successfully Put into DB.

- 0 : Entry updated successfully
- <0 : An error occurred:
  - CFE_NCA : No Configuration database Available.
  - CFE_MCF : Memory allocation for Configuration Failed.

17.6.2.2 int cfputtime (CFFLAGTYP *td*)

Set all Time and/or Date entries in CFLIB DB to *now* or *today*.

Parameters:

td Target Selection Mask:
- CF_TIME : Set Time
- CF_DATE : Set Date
- CF_TD : Set Time and Date

Returns:

- >0 : Number of entries updated successfully
- <=0 : An error occurred:
  - CFE_NCA : No Configuration database Available.
  - CFE_MCF : Memory allocation for Configuration Failed.
  - CFE_NEA : No Entry with that name Available.

17.6.2.3 int cfnosave (char * *name*, const char * *onoff*)

Alter or query the CF_NOSAVE Flag of Parameter *name*.

When the configuration database is saved to a configuration file, the function cfsave() will exclude all items with the CF_NOSAVE flag set from the output.

The CF_NOSAVE flag can be set in the Special Option Mask CONFIG::flag for every entry in the Configuration Initializer given to cfinit() or cfstart() or later be set with this function for parameters in the current configuration database _conf

Residual Items will have the CF_NOSAVE flag set by default.

Parameters:

name Entry's name
- "" : All entries
- NULL : All hardcoded entries

onoff
- CF_FLAG_ON : Set Flag
- CF_FLAG_OFF : Delete Flag
- "i" : Inquire Flag

Returns:

- 0 : Entry updated successfully / Flag is OFF (for "i")

- `1` : Flag is ON (for `"i"`)
- `!=0` or `1` : An Error occurred:
  - CFE_NCA : No Configuration database Available.
  - CFE_NEA : No Entry with that name Available.

17.6.2.4 int cfsave (char * *file*, const char * *savemode*)

Write configuration data to a Configuration File or `stdout`.

Entries with the CF_NOSAVE flag will be excluded from the output. Use cfnosave() to inquire or alter that flag for an entry.

Parameters:

file • <*string*> : Name of regular file to be opened
- `""` : Write configuration to `stdout`
- `NULL` : Private configuration file will be overwritten or created, if an appropriate entry exists

savemode File open mode:
- `"w"` : Overwrite
- `"a"` : Append

Returns:

- `0` : Configuration has been saved to file successfully
- `!=0` : An error occurred:
  - CFE_NCA : No Configuration database Available.
  - CFE_IFP : Invalid Filename entry for Private setfile.
  - CFE_WAE : Write Access Error.

17.7 General Utilities

Defines

- #define CF_BACKBUFLEN 102400
  File copy buffer for BackupFile().
- #define DelFlag(a, b) a&=(~b)
  Delete bits of Mask `b` *from Mask* `a`.
- #define SetFlag(a, b) a|=b
  Set bits of Mask `b` *in Mask* `a`.

Functions

- void RemoveCR (char *ptr)
  String Utility Function.
- void RemoveTrailSpace (char *ptr)
  String Utility Function.
- char * EatWhiteSpace (char *ptr)
  String Utility Function.
- int IsATerminal (FILE *fp)
  Test whether stream is a terminal.
- int BackupFile (const char *file, char *modus)
  Copy or Rename File `"file"` *to Backup File* `"file~"` *or* `"file.bak"`.
- char * FindFile (const char *fname, const char *fpath, const char *const *fext)
  Find a File in Path trying Extensions.

17.7.1 Function Documentation

17.7.1.1 void RemoveCR (char * *ptr*)

String Utility Function.

Strip Carriage Return at end of string (after fgets) by introducing zero byte at CR position. Original source was "STELM" by Kees and Lemmens.

Author:

Kees and Lemmens

Parameters:

ptr pointer to beginning of string

17.7.1.2 void RemoveTrailSpace (char * *ptr*)

String Utility Function.

Strip whitespaces at end of string by introducing zero byte after last non-whitespace character

Parameters:

ptr Pointer to beginning of string

17.7.1.3 char* EatWhiteSpace (char * *ptr*)

String Utility Function.

Set pointer to next non-whitespace-character in string. Original source was "STELM" by Kees and Lemmens.

Author:

Kees and Lemmens

Parameters:

ptr Pointer to beginning of string

Returns:

Pointer to next non-whitespace-character in string

17.7.1.4 int IsATerminal (FILE * *fp*)

Test whether stream is a terminal.

Parameters:

fp File/Stream Pointer

Returns:

`TRUE` or `FALSE`

Bug

ANSI C doesn't have function isatty(), we always return TRUE

17.7.1.5 int BackupFile (const char * *file*, char * *modus*)

Copy or Rename File `"file"` to Backup File `"file~"` or `"file.bak"`.

Parameters:

file Name of regular file to be opened

modus Backup Mode:

- `"r"` Rename
- `"d"` Duplicate, Copy
- `"@e ?"` Default `"d"`
- `"@e ?d"` docs-style (.bak)
- `"@e ?c"`, `"??"` Un*x Style (~)

Returns:

- `0` : Configuration has been saved to file successfully
- `!=0` : An error occurred:
  - CFE_BOF : Invalid filename / open error
  - CFE_BMF : Backup: Memory allocation Failed.
  - CFE_BRF : Rename file failed
  - CFE_BBF : Source file open error
  - CFE_BWF : Target file write error

17.7.1.6 char* FindFile (const char * *fname*, const char * *fpath*, const char *const * *fext*)

Find a File in Path trying Extensions.

Author:

Eric R. Smith

License:

Public Domain

Parameters:

fname File Name

fpath Search Path: String of Directories separated by **PATHSEP1** (':' or ';') or **PATHSEP2** (nothing or ',')

fext Array of possible Extensions (optional, default is OS dependent)

Returns:

The name by which the file was found or `NULL`

17.8 Report Generation

Defines

- #define CF_DEF_VARDELIM "$()"

  Default variable delimiter for cfform().

- #define CF_MAXINC 8

  Maximum number of nested includes for cfform().

Functions

- int cfform (char *file, char *outfile, char *vd, int mode)

  Process a Template from file or `stdin` *and write generated Report to File or* `stdout`.

17.8.1 Function Documentation

17.8.1.1 int cfform (char * *file*, char * *outfile*, char * *vd*, int *mode*)

Process a Template from file or `stdin` and write generated Report to File or `stdout`.

The Report Template contains **Variables** like '$(*ident*)' or whatever you set in *vd*. *ident* may refer to a:

- Parameter Name like '$(*name*)' : Substitute variable by parameter's value (from DB)
- Include File Path like '$(FILE:*/my/path/to/incfile*)' : Substitute variable by content of include file
- File Path and Section Include like '$(FILE:*incfile*#sect*)'* : Include section from include file between [*sect*] and next [...]
- Variable File Include like '$(FILE::*varname*)' : Include file from location given in parameter *varname*

See also:

Advanced Usage Example

Todo

Make cfform() work with buffers instead of files

Parameters:

file Name of Template File, `NULL` for `stdin`

outfile Name of Output File, `NULL` for `stdout`

vd Variable Delimiters: string containing the three variable delimiters in the first three chars: to have variables like '$(*name*)', let the string be '$()'. This is also the default, if *vd* is `NULL` or string length < 3

mode Mode Mask:

- `0` : Normal
- `1` : Query for unresolved variables, (try to) include them in database
- `2` : Unset CF_NOSAVE flag for parsed vars
- `4` : Outfile Write *append*, else *write*

Returns:

- `0` : Configuration has been saved to file successfully
- `!=0` : An error occurred:
  - CFE_NCA : No Configuration database Available.
  - CFE_FNF : File Not Found, read access error.
  - CFE_WAE : Write Access Error.

18 Data Structure Documentation

18.1 CONFERR Struct Reference

Library Internal: Error List Item.

Data Fields

- struct _cfe * next
  Next Error Pointer.
- int errcode
  Numeric Error Code.
- char errstr [CF_MAXERRSTR]
  Error Message String of maximum length CF_MAXERRSTR.

18.1.1 Detailed Description

Library Internal: Error List Item.

18.2 CONFIG Struct Reference

CFLIB Configuration Database Entry.

Data Fields

- char * name
  Parameter Name
- char * inhalt
  Parameter Content, see Parameter Default Value.
- char option
  Commandline Option for Parameter
- CFFLAGTYP flag
  Special Options Mask

18.2.1 Detailed Description

CFLIB Configuration Database Entry.

19 File Documentation

19.1 include/cf.h File Reference

C Header File for CFLIB Flexible Configuration Library.

Data Structures

- struct CONFIG

  CFLIB Configuration Database Entry.

Defines

- #define Patchlevel "20"

  CFLIB Identification.

- #define MAXCONF 4096

  Maximum number of entries in configuration database.

- #define CF_MAXERRSTR 512

  Maximum string length for error message.

- #define CF_MAXLINE 20480

  Maximum string length for setfile and form parsing.

- #define CF_MAXQLINE 512

  Maximum string length for query.

- #define CF_MAXUSAGE 1024

  Maximum string length for usage string.

- #define CF_USG_DEFCOLS 80

  Default terminal width for usage string.

- #define CF_MAXTIMEBUF 256

  Buffer size for time and day.

- #define TRUE 1

  TRUE, if not defined.

- #define FALSE 0

  FALSE, if not defined.

- #define NULL (void *)(0L)

  NULL, if not defined.

- #define CF_FLAG_ON "\1"

  Flag is set.

- #define CF_FLAG_OFF ""

  Flag is not set.

- #define CF_NO_OPTION ' '

  Option is not set.

- #define TABLEN 8

  TAB length.

- #define CFP_PUT 0

  Function Call or Automatic Initialization.

- #define CFP_ARG 1

  Commandline.

- #define CFP_ENV 2

  Environment.

- #define CFP_PRIV 3

  Private Configuration File.

- #define CFP_SYS 4

  System Configuration File.

- #define CFP_DEF 5

  Built-in Default.

- #define CFP_QRY 6

  Standard Input Channel.

- #define CFP_RESERVED 7

  Reserved for Subprojects.

- #define CF_DEF_VARDELIM "$()"

  Default variable delimiter for cfform().

- #define CF_MAXINC 8

  Maximum number of nested includes for cfform().

- #define CF_BACKBUFLEN 102400

  File copy buffer for BackupFile().

- #define CF_LAST 0x0001

  Last entry in settings array.

- #define CF_PATH 0x0002

  Search Path (for FindFile feature).

- #define CF_SETFILE 0x0004

  Entry is Private Configuration File.

- #define CF_SYS_SETFILE 0x0008

  Entry is System Configuration File.

- #define CF_SECTION 0x0010

  Section in Configuration File.

- #define CF_PRGNAME 0x0020

  Running Program's Name from commandline.

- #define CF_TIME 0x0040

  Time string.

- #define CF_DATE 0x0080

  Date string.

- #define CF_SET_PUT 0x0100

  Source: Function Call.

- #define CF_SET_ARG 0x0200

  Source: Commandline Argument.

- #define CF_SET_ENV 0x0400

  Source: Environment Variable.

- #define CF_SET_PRIV 0x0800

  Source: Private Configuration File.

- #define CF_SET_SYS 0x1000

  Source: System Configuration File.

- #define CF_SET_DEF 0x2000

  Source: Built-in Default.

- #define CF_SET_QRY 0x4000

  Source: Interactive Terminal Input.

- #define CF_NO_OPT_ARG 0x10000

  Commandline argument not followig an option.

- #define CF_CONCAT 0x20000

  Argument is concatenated to option.

- #define CF_IGN_ENV 0x40000

  Do not check environment for variable.

- #define CF_QUERY 0x80000

  Ask the user for unresolved item after configuration parsing.

- #define CF_STR 0x100000

  Entry is String.

- #define CF_INT 0x200000

  Entry is Integer.

- #define CF_FLAG 0x400000

Entry is Flag.

- #define CF_REAL 0x800000

  Entry is Float.

- #define CF_FINDFILE 0x1000000

  Entry is filename to be searched for in the path.

- #define CF_MUST 0x2000000

  Entry may not be empty (`NULL` *or "").*

- #define CF_RESID 0x4000000

  Residual/additional entry from commandline/setfile/cfput.

- #define CF_USAGE 0x8000000

  Usage Message format string.

- #define CF_EXPHOME 0x10000000

  Expand Home Directory.

- #define CF_MALLOC 0x20000000

  Space for entry's content was mallocated.

- #define CF_FORCED 0x40000000

  Setting has been forced (already).

- #define CF_NOSAVE 0x80000000

  Don't include in savefile / mark entry.

- #define CF_SRC (CF_INT|CF_FLAG)

  Type for source/origin inquiry.

- #define CF_FLGINQ (CF_STR|CF_FLAG)

  Type for options mask inquiry.

- #define CF_TD (CF_DATE|CF_TIME)

  Date or Time entry.

- #define CFE_INIT 0

  INITialize error input.

- #define CFE_OK 0

  No error / everything OKay.

- #define CFE_NEP 1

  New Entry successfully Put into DB.

- #define CFE_EXIT 1

  Finish error input.

- #define CFE_ORA 20

  Option Requires an Argument.

- #define CFE_UKO 25

  UnKnown Option.

- #define CFE_FNF 30

  File Not Found, read access error.

- #define CFE_NSE 40

  No Section specifier End found, missing "]".

- #define CFE_NSC 50

  No private Setfile Configured.

- #define CFE_WAE 60

  Write Access Error.

- #define CFE_IFP 61

  Invalid Filename entry for Private setfile.

- #define CFE_EWN 70

  Entry Without Name.

- #define CFE_ICF 80

  Invalid Combination of Flags.

- #define CFE_EWC 90

  Entry Without Content.

- #define CFE_UOS 100

  Unlikely Option Specifier.

- #define CFE_IFC 110

  Invalid Flag Combination.

- #define CFE_NLE 120

  No Last Entry flag found.

- #define CFE_TIN 130

  Error reTurn from stdIN query.

- #define CFE_EFE 140

  Empty string in content for Filename Entry.

- #define CFE_USG 200

  Entries missing: USaGge advice.

- #define CFE_URI 210

  UnResolved Item (CF_MUST was set!).

- #define CFE_FBF -500

  File Backup Failed.

- #define CFE_BMF -510

  Backup: Memory allocation Failed.

- #define CFE_BOF 520

  Backup: Open source file Failed.

- #define CFE_BBF 530

  Backup: open target Backup file Failed.

- #define CFE_BRF 540

  Backup: Rename Failed.

- #define CFE_BWF 550

  Backup: Write Failed.

- #define CFE_NCA -10

  No Configuration database Available.

- #define CFE_NEA -20

  No Entry with that name Available.

- #define CFE_NSS -30

  No Source/origin is Set.

- #define CFE_ECP -40

  Entry's Content is a NULL Pointer.

- #define CFE_MEF -100

  Memory allocation in Error routine Failed.

- #define CFE_MCF -200

  Memory allocation for Configuration Failed.

- #define CFE_INF -9999

  Integer iNquiry Failed (?!).

- #define CFE_RNF -999.999

  Real/float iNquiry Failed (?!).

- #define CFS_NOT 0

  Start Mode: No action on error.

- #define CFS_ALL 1

  Start Mode: All error messages.

- #define CFS_NEG 2

Start Mode: Only severe errors.

- #define CFS_USG 3

  Start Mode: Usage message if error was negative, error output like CFS_NEG.

- #define CFS_DEBUG 4

  Start Mode: Output like CFS_USG plus raw dump of configuration.

- #define CFD_CFDUMP 0

  Dump Mask Minimal.

- #define CFD_LIBHEAD 1

  Dump option CFLIB header.

- #define CFD_COLHEAD 2

  Dump option Column headers.

- #define CFD_SRCFLAGS 4

  Dump option Source flag description.

- #define CFD_DEFAULT CFD_COLHEAD|CFD_SRCFLAGS

  Dump Mask Default.

- #define cfget(a) cfgetent(a,0)

  Get value (content) of named entry.

- #define cfgetstr(a) ((char *)cfgetent(a,CF_STR))

  Inquire CFLIB DB for String in content of named entry.

- #define cfgetnum(a) (*(int *)cfgetent(a,CF_INT))

  Inquire CFLIB DB for Integer value in content of named entry.

- #define cfgetreal(a) (*(float *)cfgetent(a,CF_REAL))

  Inquire CFLIB DB for Float (Real) value in content of named entry.

- #define cfgetflag(a) (*(int *)cfgetent(a,CF_FLAG))

  Inquire CFLIB DB for Flag value in content of named entry.

- #define cfflaginq(a, b) (*(int *)cfgetent(a,CF_FLGINQ|(31&b)))

  Inquire CFLIB DB for Bit set in entry's Special Options Fag CONFIG::flag.

- #define cfgetsrc(a) (*(int *)cfgetent(a,CF_SRC))

  Inquire CFLIB DB for Source of named entry's content.

- #define cfgetres() ((char *)cfgetent("",CF_RESID))

  Get next Residual Command Line Argument from CFLIB DB.

- #define cfgetcpr() "CFLIB (c) 1994-2009 Stefan Habermehl"

  Get Copyright Notice.

- #define cfput(a, b) cfputstr(a,(char *)b)

  Update or Add Parameter (Utility Function Macro).

- #define DelFlag(a, b) a&=(~b)

  Delete bits of Mask `b` *from Mask* `a`.

- #define SetFlag(a, b) a|=b

  Set bits of Mask `b` *in Mask* `a`.

- #define __CF_H__

  Marker: cf.h has been included.

Typedefs

- typedef unsigned long CFFLAGTYP

  Special Options Mask Type.

Functions

- int cfinit (CONFIG *set, int argc, char **argv)

  Initialize CFLIB Configuration Database and parse possible sources for database entries according to the settings in set.

- void cfexit (void)

  Free allocated memory and reset the configuration database and error stack.

- int cfform (char *infile, char *outfile, char *vardelim, int mode)

  Process a Template from file or `stdin` *and write generated Report to File or* `stdout`.

- void * cfgetent (char *name, CFFLAGTYP typ)

  Library internal function, use appropriate Macro functions!

- char * cfhomexp (char *name)

  Expand ~ or ~`user` *in parameter content.*

- int cfgetvers (void)

  Get Library Version/Patchlevel.

- char * cfgetsubvers (void)

  Get Library Subversion Details.

- int cfnosave (char *name, const char *onoff)

  Alter or query the CF_NOSAVE Flag of Parameter name.

- int cfputstr (char *name, char *content)

  Update or Add Parameter name *with string* content.

- int cfputtime (CFFLAGTYP td)
  Set all Time and/or Date entries in CFLIB DB to now *or* today.
- char * cfgetusg (void)
  Get Usage Message for (Terminal) Output.
- int cfgeterr (char *string, size_t len)
  Error Code and Message Inquiry Function.
- int cfputerr (int ecode, char *string,...)
  Init, exit or append to Error List.
- void cfclearerr (void)
  Free all entries in error list.
- int cfreverr (void)
  Revert order of entries in error list from last->first to first->last.
- int cfstart (CONFIG *set, int ac, char **av, char *help, int mode)
  Setup Configuration Database.
- int cfdinichk (CONFIG *set)
  *Debugging Function (***experimental***).*
- int cfdump (FILE *fout)
  Debugging Function.
- int cfsave (char *fname, const char *savemode)
  Write configuration data to a Configuration File or `stdout`.
- int BackupFile (const char *file, char *modus)
  Copy or Rename File `"file"` *to Backup File* `"file~"` *or* `"file.bak"`.
- void RemoveCR (char *string)
  String Utility Function.
- void RemoveTrailSpace (char *string)
  String Utility Function.
- char * EatWhiteSpace (char *string)
  String Utility Function.
- int IsATerminal (FILE *fp)
  Test whether stream is a terminal.
- char * FindFile (const char *fname, const char *fpath, const char *const *fext)
  Find a File in Path trying Extensions.

19.1.1 Detailed Description

C Header File for CFLIB Flexible Configuration Library.

Public Functions and Definitions

Note:

Include this file in the source code and link with the library executable, usually referring to *libcf.a* by calling "`(g)cc -lcf ...`"

Version:

SVN: $Id: cf.h 127 2009-01-28 16:50:14Z stefan $

Author:

Stefan Habermehl <`stefan.habermehl@mcff.de`>

Copyright:

(c) 1994,1995,1996,1997,1998,2006,2007,2008,2009 Stefan Habermehl

License:

`http://www.gnu.org/licenses` GNU General Public License v3 or later

19.1.2 Define Documentation

19.1.2.1 #define cfput(a, b) cfputstr(a,(char *)b)

Update or Add Parameter (Utility Function Macro).

Parameters:

a Parameter Name

b New Content (Type casted to expected Char Pointer)

Returns:

int cfputstr()

Index

Advanced Features, 40
advanced_features
    cfdinichk, 41
    cfdump, 40
    cfhomexp, 41

BackupFile
    utilities, 52

cf.h
    cfput, 65
cfclearerr
    errors, 39
cfdinichk
    advanced_features, 41
cfdump
    advanced_features, 40
cfexit
    cflib_core, 32
cfflaginq
    retrieval, 45
cfform
    report_generation, 54
cfget
    retrieval, 43
cfgetcpr
    retrieval, 46
cfgetent
    retrieval, 46
cfgeterr
    errors, 39
cfgetflag
    retrieval, 44
cfgetnum
    retrieval, 43
cfgetreal
    retrieval, 44
cfgetres
    retrieval, 45
cfgetsrc
    retrieval, 45
cfgetstr
    retrieval, 43
cfgetsubvers
    retrieval, 47
cfgetusg
    retrieval, 47
cfgetvers
    retrieval, 47
cfhomexp
    advanced_features, 41
cfinit
    cflib_core, 31
CFLIB Core Features, 29
cflib_core
    cfexit, 32
    cfinit, 31
    cfstart, 32
cfnosave
    setting_saving, 49
cfput
    cf.h, 65
cfputerr
    errors, 39
cfputstr
    setting_saving, 48
cfputtime
    setting_saving, 49
cfreverr
    errors, 39
cfsave
    setting_saving, 50
cfstart
    cflib_core, 32
CONFERR, 54
CONFIG, 55

EatWhiteSpace
    utilities, 51
Error Handling, 35
errors
    cfclearerr, 39
    cfgeterr, 39
    cfputerr, 39
    cfreverr, 39

FindFile
    utilities, 53

General Utilities, 50

include/cf.h, 55
Information Retrieval, 42
IsATerminal
    utilities, 52

RemoveCR
    utilities, 51
RemoveTrailSpace
    utilities, 51
Report Generation, 53
report_generation
    cfform, 54

retrieval
- cfflaginq, 45
- cfget, 43
- cfgetcpr, 46
- cfgetent, 46
- cfgetflag, 44
- cfgetnum, 43
- cfgetreal, 44
- cfgetres, 45
- cfgetsrc, 45
- cfgetstr, 43
- cfgetsubvers, 47
- cfgetusg, 47
- cfgetvers, 47

Setting and Saving the Configuration, 48
setting_saving
- cfnosave, 49
- cfputstr, 48
- cfputtime, 49
- cfsave, 50

Special Options Mask, 33

utilities
- BackupFile, 52
- EatWhiteSpace, 51
- FindFile, 53
- IsATerminal, 52
- RemoveCR, 51
- RemoveTrailSpace, 51