The Assertion Library

0.2.1

Generated by Doxygen 1.7.6.1

Tue Apr 23 2013 22:31:29

Contents

1 C Basic Library: Assertion Library **1**

1.1 Introduction 1

1.2 How to Use The Library 2

1.3 Boilerplate Code 2

1.4 Future Directions 2

1.5 Contact Me 3

1.6 Copyright 3

2 File Index **5**

2.1 File List 5

3 File Documentation **7**

3.1 assert.c File Reference 7

3.1.1 Detailed Description 7

3.2 assert.h File Reference 7

3.2.1 Detailed Description 8

3.2.2 Define Documentation 8

3.2.2.1 assert 8

Chapter 1

C Basic Library: Assertion Library

Version

0.2.1

Author

Jun Woong (woong.jun at gmail.com)

Date

last modified on 2013-04-23

1.1 Introduction

This document specifies the Assertion Library which belings to the C Basic Library. - The basic structure is from David Hanson's book, "C Interfaces and Implementations." I modified the original implementation to make it more appropriate for my other projects as well as to increase its readability as done for the Exception Handling Library; conformance is not an issue here because trying to replace an exsisting standard library by "knocking it out" is already cosidered non-conforming.

Replacing <assert.h> with a version to support exceptions is useful based on some assumptions:

- Deactiaving assertions in product code does not bring much benefit to the program's performance (only profiling would be able to say how much assertions degrades its performance);
- Avoiding assertions showing cryptic dianostic messages leads users into a more catastrophic situation like dereferencing a null pointer;
- It is not that a diagnostic message like "Uncaught exception Assertion failed raised at file.c:12" without an expression that caused the assertion to fail conveys

less information than the message from the standard version, if the programmer can access to the problematic file; and

- Dealing with assertion failures by an exception has an advantage that every assertion can be handled in a uniform way (see an example below).

More detailed explanation on the motivation is given in Hanson's book, so I should stop here but I provides introduction to the library; how to use the facilities is deeply explained in files that define them.

The Assertion Library reserves identifiers `assert` and `ASSERT`, and imports the - Exception Handling Library.

1.2 How to Use The Library

Using the Assertion Library is not quite different from using the standard version. - Differntly from Hanson's original implementation, this implementation includes the standard version <assert.h> if either `NDEBUG` or `ASSERT_STDC_VER` is defined.

The following example shows how to handle assertion failures in one place:

```
#include "assert.h"
#include "except.h"

int main(void)
{
    EXCEPT_TRY
        real_main();
    EXCEPT_EXCEPT(assert_exceptfail)
        fprintf(stderr,
                "An internal error occurred with no way to recover.\n"
                "Please report this error to somebody@somewhere.\n\n");
        RERAISE;
    EXCEPT_END;

    return 0;
}
```

If you use an ELSE clause instead of the EXCEPT clause given above, any uncaught exception lets the program print the message before aborting. (See the documentation in the Exception Handling Library.)

1.3 Boilerplate Code

No boilerplate code is provided for this library.

1.4 Future Directions

No future changes on this library planned yet.

1.5 Contact Me

Visit http://code.woong.org to get the latest version of this library. Only a small portion of my homepage (http://www.woong.org) is maintained in English, thus one who is not good at Korean would have difficulty when navigating most of other pages served in Korean. If you think the information you are looking for is on pages written in Korean, do not hesitate to send me an email to ask for help.

Any comments about the library are welcomed. If you have a proposal or question on the library just email me, and I will reply as soon as possible.

1.6 Copyright

ERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Chapter 2

File Index

2.1 File List

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

assert.c
Source for Assertion Library (CBL) 7
assert.h
Header for Assertion Library (CBL) 7

Chapter 3

File Documentation

3.1 assert.c File Reference

Source for Assertion Library (CBL)

`#include "cbl/assert.h"` `#include "cbl/except.h"` Include dependency graph for assert.c:

Variables

- const except_t assert_exceptfail = { "Assertion failed" }

  exception for assertion failure.

3.1.1 Detailed Description

Source for Assertion Library (CBL)

3.2 assert.h File Reference

Header for Assertion Library (CBL)

`#include "cbl/except.h"` Include dependency graph for assert.h:

Defines

- #define assert(e) ((void)((e) || (EXCEPT_RAISE(assert_exceptfail), 0)))

  replaces the standard assert() with a version supporting an exception.

Variables

- const except_t assert_exceptfail

  exception for assertion failure.

3.2.1 Detailed Description

Header for Assertion Library (CBL) Documentation for Assertion Library (CBL)

3.2.2 Define Documentation

3.2.2.1 #define assert(*e*) ((void)((e) || (EXCEPT_RAISE(assert_exceptfail), 0)))

replaces the standard assert() with a version supporting an exception.

An activated assert() raises an exception named `assert_exceptfail` that is defined in assert.c. The differences between this exception version and the standard's version are

- The exception version does not print the given expression, `e`;
- The exception version does not abort; it merely raise an exception and let the exception handler decide what to do.

Possible exceptions: assert_exceptfail

Unchecked errors: none