The NCBI C++ Toolkit Book

The NCBI C++ Toolkit Book

Release Notes (August, 2005)

Created: October 3, 2005.
Last Update: October 4, 2005.

The NCBI C++ Toolkit Book

The NCBI C++ Toolkit Book

Download Location

ftp://ftp.ncbi.nih.gov/toolbox/ncbi_tools++/2005/Aug_31_2005/

Source Archive Contents

Source Code Archives

- ncbi_cxx_unix--Aug_31_2005.tar.gz -- for UNIX'es (see the list of UNIX flavors below) and MacOSX/GCC
- ncbi_cxx_unix-- Aug_31_2005.gtar.gz -- for UNIX'es (see the list of UNIX flavors below) and MacOSX/GCC
- ncbi_cxx_win-- Aug_31_2005.exe -- for MS-Windows / MSVC++ 7.1 (self-extracting)
- ncbi_cxx_win-- Aug_31_2005.zip -- for MS-Windows / MSVC++ 7.1
- ncbi_cxx_mac_cw-- Aug_31_2005.tgz -- for MacOS 10.3.4 / CodeWarrior DevStudio for MacOS 9.2
- ncbi_cxx_mac_xcode-- Aug_31_2005.tgz -- for MacOS 10.3.4 / xCode 1.[1-5]

The sources correspond to the NCBI production tree sources from patch "CATCHUP_AUG_2005", which in turn roughly corresponds to the development tree sources from 8-12 of August, 2005.

New Development

CORELIB -- Portability and Application Framework

1. NStr class:
   - Revamp of StringToXxx() and XxxToString() methods. Added versions with a bit-wise flag parameter. Old enum methods made obsolete and will be removed in the next release. Please use versions with the flag parameter!
   - TokenizePattern() -- new method using a string (rather than a set of characters) for the delimiter.
2. CDll class - added TFlags type and new constructors. Added RTLD_LOCAL/ RTLD_GLOBAL support (UNIX only).

3 CDirEntry class:

- IsNewer() -- added time_t and CTime versions, and a flag to specify what to do if the directory entry does not exist or is not accessible.
- New method Stat() and struct SStat -- to get non-POSIX OS-dependent info.
- SetTime[T]() -- allow to set creation time.
- New method IsIdentical().
- Copy() can copy all supported types now.
- Rename() -- try to "copy/remove" if "rename" failed.
- New method GetEntriesPtr() -- faster for listing extra-large directories.
- Stat() has changed to return bool, instead of int.
- GetPath() has changed to return reference, instead of a copy.
- Rename() to copy on EACCES on Unix; introduce special bits a la chmod().

4 CFileDeleteList and CFileDeleteAtExit -- to help schedule deletion of files at the application exit.

5 CThread::fRunNice -- flag to run threads with low priority (MS-Windows only).

6 CException -- now can keep a severity attribute, and pass it to the Toolkit diagnostic stream.

7 Diagnostics -- DIAG_COMPILE_INFO now retrieves function/method name on all platforms except SUN Workshop.

8 CDiagFilter - changed semantic of the "!" (negation) operator from NOT to AND NOT for the expression evaluation purposes. Fixed some bugs along the way.

9 CDllResolver -- case-insensitive comparison of directory names on Windows in FindCandidates. Also, implemented a new algorithm for DLLs name resolving.

10 Added driver version traits class CDefaultDriverVersion to a Plugin Manager framework. Replaced NCBI_INTERFACE_VERSION(IFace) macro with the GetDefaultDrvVers() method call. Use CDefaultDriverVersion here and there (IClassFactory, CPluginManager, CSimpleClassFactoryImpl) instead of hard-coded template parameter TIfVer.

11 CVersionInfo -- fixed Match() and IsBetterVersion() methods. Added method IsUpCompatible(), improved version string parsing.

12 Changed signature of CNcbiApplicaion::DisableArgDescriptions() method. Now it takes variable of TDisableArgDesc type instead of bool.

13 Sleep*() API -- a better (more native) implementation for Linux.

14 ncbi_system::GetVirtualMemoryPageSize() -- to support platforms that define _SC_PAGE_SIZE.

15 CMetaRegistry -- support reloading .ini files that have changed since they were previously loaded.

16 CSysLog -- new diagnostic handler that uses the standard Unix system logging mechanism.

17 CNcbiApplication -- when the user explicitly requests help, send it to standard output rather than standard error.

18 P{Case,Nocase}_CStr -- new functors to reduce overhead when working with C strings (useful with CStaticArray{Map,Set}).

The NCBI C++ Toolkit Book

The NCBI C++ Toolkit Book

19 CObjectMemoryPool -- new class, to allow allocation of CObject derived objects from local memory pool. It can reduce time used on memory allocation due to thread locality. It also reduces used memory. CObject remembers source of memory for the object and deallocates it correspondingly.

20 pair_base_member<> -- new template, for empty base optimization.

21 CRef<> and CConstRef<> -- added second template parameter, for locking.

CONNECT -- Data streaming, Networking, and Dispatching

1 Default request method for HTTP has been changed to pseudo-name ANY, which is figured out to be either POST or GET depending on the content length of the request being created. Formerly, it was POST explicitly. The change should be fully backward compatible.

2 Additional flags introduced for FTP and SOCK connectors, mostly for fine control over data/command logging.

3 Some minor fixes in Conn_IOStream-derived classes; automatic CONNECT_Init() from constructor - to make logging easier and setup free.

4 SOCK_, CSocket API -- new call GetLoopbackAddress(). Also, allow '\r\0' as a line terminator in ReadLine().

UTIL -- Miscellaneous Low-Level APIs

1 CZipCompressionFile -- rewritten using compression streams. Use GZIP file format by default. CompressFile() now can write file name and mtime into GZIP file header.

2 CTar class:

- Update() -- new method to refresh archive content with new versions of the files that it already contains.
- Added support for (re)storing of file permissions, owner, and times.
- Made compatible with many TAR implementations in use these days.
- Underwent a significant redesign and troubleshooting.

3 RangeCollection -- fixed problems with unsigned ranges starting at zero and special values.

4 CThreadNonStop -- optimized thread shutdown procedure, using semaphores.

SERIAL -- Data Serialization (ASN.1, XML, Schema)

1 XML serialization::

- Implemented for containers of elements with mixed content.
- Corrected for elements with attributes -- to preserve leading white spaces in data.
- Fixed delayed reading of strings in XML data (strings were lost before).
- Corrected and optimized serialization of elements with content of type ANY.

2 Fix to not intercept user-defined exceptions thrown by user code from inside the object streams' hooks.

3 Allow for the allocation of serializable objects in CObjectMemoryPool.

4 CUnionBuffer<> -- new template for inlined objects in choices, to allow the placement of choice variants inside choice object even for non-primitive types.

5 Optimized parsing of binary ASN.1.

The NCBI C++ Toolkit Book

The NCBI C++ Toolkit Book

6 Use vector<Uint1> instead of inefficient vector<bool>.

7 Used enums to represent ASN.1 constants whenever possible.

DATATOOL -- Code Generator and Data Converter Tool

1 Allow white spaces in file paths.

2 Allow using multiple -m and -M command line arguments, to specify multiple input files with blanks in their names.

3 Fixed generation of the C++ code that uses multi-level namespaces.

4 Added possibility to tune-up generated C++ code for classes of types (using DEF file), that is, by data type rather than just element name.

5 Implemented generation of modular XML schema by ASN specification.

6 Implemented analysis of module dependency tree to generate correct modular DTD and XML schema.

7 Corrected generation of XML schema for elements of boolean type with default.

8 Improved diagnostics.

CGI -- CGI and Fast-CGI Application Framework

1 Added flags to control URL encoding/decoding of the cookies.

2 CCgiCookies -- allow handling invalid cookies: allow skipping or storing them. Added CCgiCookie::IsInvalid(). Throw an exception on an attempt to write a malformed cookie.

3 CgiContext -- allow creating a CGI context from an input stream.

4 CCgiRequest -- new Serialize() and Deserialize() methods, to pass the request through a stream.

5 CCgiCookie[s] -- new Write() method, to serialize cookie[s] content to a stream.

HTML -- HTML Generation Library

1 CHTMLNode::AttachPopupMenu() -- changed type of 3rd parameter from bool to TPopupMenuFlags. Also added parameter for canceling default event processing (default is true).

2 CHTML_input_button -- new class for <input type=button>.

3 CHTML_button -- resurrected, as most modern browsers support <button> tag, specified in the HTML 4.0.

4 CNCBINode -- added method ReInitialize().

5 CHTMLPage -- added support for #include command in HTML template libraries.

Berkeley DB API (bdb) -- Much Enriched C++ API Based On BerkeleyDB

1 Limit BLOB TTL prolongation on read. This fixes potential vulnerability (DDOS).

2 Added "overflow limit" cache parameter.

3 Optimized BDB cache shutdown procedure.

DBAPI -- Generic SQL Database Connectivity

1 Revamp of DBAPI exception class CDB_Exception. It is now inherited from the CException class, uses severity level from the CException class, and has method Clone().

The NCBI C++ Toolkit Book

The NCBI C++ Toolkit Book

2. Implemented a driver based on the new version of FreeTDS (v0.63).
3. IStatement - added SetAutoClearInParams() and IsAutoClearInParams() methods.
4. CDB_Object -- added GetTypeName() method.
5. Database unit test suite has been moved from CPPUNIT to Boost.Test.
6. Merged ftds/interfaces.hpp into dblib/interfaces.hpp
7. FreeFTDS8 driver -- fixed bind data types for variable-sized data types with Bulk operations.
8. DBLIB driver -- fixed column data buffer size to hold up to 8k in case of Bulk operations.

PYTHON database module based on DBAPI

1. CCursor - added get_proc_return_status method.
2. Moved Python DBAPI module test suite to Boost.Test.

ALGO/ALIGN -- Generic Alignment Algorithms

1. Cross-species flag added to Splign application. This flag only has effect in pair wise mode, i.e. when blast hits are computed internally.
2. In Splign's compartmentization algorithm, the default compartment penalty has been adjusted and additional parameter, minimal compartment identity has been introduced to filter out low-scoring compartments while still being able to identify them.
3. Assertion has been added to make sure that the scoring matrix is re-initialized with SetScoreMatrix(NULL) after every call to SetWm() and SetWms() (nucleotide case only).

BLAST

1. Added Seq-align-set representation of results to CRemoteBlast similar to that produced by local search classes (CBl2Seq).
2. PHI BLAST implementation reorganization and cleanup (CORE BLAST).
3. Nucleotide sequence filtering is delegated to ncbi::CSymDustMasker in the C++ BLAST APIs.
4. Reorganization of setup code to allow NCBI C++ object manager-free interfaces.
5. Added PSSM frequency ratios as input to PSSM engine.
6. Added engine version information (CORE/API BLAST).
7. Updated documentation.

BIO-OBJECTS -- Bio-Object Specific Utility Functions (Not Involving OM++)

1. CSeq_id, CTextseq_id -- refactored, introducing Set() methods that can be called on previously initialized IDs, and a static ParseFastaIDs() method.
2. New CSeqIdException class for uniform error reporting.
3. CSeq_id -- support several new prefixes in IdentifyAccession() method.
4. CSeq_id_Handle -- support new type of Seq-id: gpipe.

OM++ -- Object Manager -- For Retrieving and Processing Bio-Objects

1. Added CBioseq_Handle::IsSynonym().

2 CSeq_loc_Mapper -- extends partial ranges when mapping from a protein to a nucleotide.

3 CScope::GetBioseqHandles() -- to request multiple bioseqs.

4 Added option to change letter case in output of CSeqVector/CSeqVector_CI.

5 Added annotation iterator constructors from CBioseq_Handle, CRange, and strand.

6 Implemented 'Removed' handles. Use 'Removed' handles when transferring object from one place to another. Added invalidation of handles to removed objects.

7 Distinguish between shared and private manually added Seq-entries.

8 Redirect all open handles to new TSE when detaching from data loader.

OM++ LOADERS/READERS -- Data Retrieval Libraries for OM++

1 Added Phrap (ACE) format reader (supports both old and new format versions). Added ace2asn sample application to convert from ACE to ASN.1.

2 GenBank Data Loader -- added support for tRNA external annotations.

OM++ TEST programs

1 GenBank data loader test suite:

- Added option -no_external to exclude external annotations to applications test_objmgr_data and test_objmgr_data_mt.
- Test CSeqVector_CI post increment operator.
- Added test of non-location feature iterators.
- Added test of CAnnot_CI.

2 Object Manager test suite:

- Test insertion of removed handle.
- Added test for CFeat_CI over CBioseq_Handle without explicit Seq-id.
- Test the unlocking of handles.

OM++ DEMO program (objmgr_demo)

1 Added test of edit interface.

CTOOLS -- C-to-C++ NCBI Toolkit connectivity

1 Introduced routines to convert the severity values between C and C++ Toolkits.

BUILD FRAMEWORK (UNIX)

1 Support building with GCC 4.0.x, WorkShop 5.5, and modern versions of ICC (though we do not yet support ICC 9.0 on 32-bit platforms).

2 Added checks for some more third-party libraries: Boost.Test and IBM's International Components for Unicode (ICU). Improved checks for some other third-party libraries.

3 CONFIGURE:

- Use consistent (more debugging-friendly) settings for "--with-debug --with-optimization" builds.
- Run the build project lists (used by "--with-projects" and by update_projects.sh") through the C preprocessor to support #include directives.

The NCBI C++ Toolkit Book

The NCBI C++ Toolkit Book

BUILD FRAMEWORK (MSVC++.NET)

PTB -- Project Tree Builder for MSVC++ .NET

1. CONFIGURE-DIALOG -- new, GUI project for user to interactively confirm or modify configuration parameters.
2. Added filtering projects by optional project tag, which may be present in the project makefile.
3. Allow inclusion of list files into project list files via #include.
4. Allow white spaces in file paths.
5. Made it possible in generated projects to reference missing libraries without creating project dependencies.
6. Reuse the same PTB executable in all build configurations.
7. Prohibit single-threaded build configurations in projects that require multi-threading.
8. Added possibility to generate a custom build step with macro substitution.
9. Recognize and remove circular dependencies between ASN libraries.
10. Implemented conditional macro based on presence of a file.

3RD-PARTY PACKAGES

1. Updated ZLIB to version 1.2.3.

GRID (DISTRIBUTED COMPUTING) FRAMEWORK

1. Added Grid Worker Node Framework. This framework utilizes NetCache and NetScheduler components and simplifies development of distributed applications.
2. Added Remote CGI Framework. This framework uses Grid Worker Node Framework and offers an ease way for converting an existing CGI into a distributed application. Among other things, this helps solve the problem of the hogging of the Web servers' CGI slots by long-running CGIs, and server-side timeouts.
3. NetCache server improvements:
    - Improved diagnostics and error messaging
    - Added new more fault protected network protocol for storing BLOBs
    - netcache_control program -- new option -t to get statistics
4. NetCache client API changes:
    - Added backup service (works when no other instances are available)
    - Reimplemented blob storage using fault tolerant network protocol
5. NetSchedule server improvements:
    - Implemented config realoding without server restart
    - Added admin utility with various options of remote queue monitoring
    - Added progress messages for long running jobs
    - Implemented client version control for grid management
    - Added support of load balancing
6. NetSchedule client API changes:
    - Added GetQueueList() method (retieve list of queues)
    - DumpQueue() can dump individual jobs

The NCBI C++ Toolkit Book

The NCBI C++ Toolkit Book

- Added methods to submit and receive progress messages
- Added support for client version control
- Implemented remote queue monitoring

Documentation

Document Location

The documentation is available online at http://www.ncbi.nlm.nih.gov/books/bv.fcgi?rid=toolkit.TOC&depth=2 as a book titled "The NCBI C++ Toolkit". This is an online searchable book.

The C++ Toolkit book also provides PDF version of the chapters; although these are not up to date in this release. The PDF version can be accessed by a link that appears on each page.

The older HTML documentation has been deprecated and is no longer being updated, and "The NCBI C++ Toolkit" online book at the previously listed URLs is the official documentation.

Document Content

Documentation has been grouped into chapters and sections that provide a more logical coherence and flow. New sections and paragraphs continue to be added to update and clarify the older documentation or provide new documentation. The chapter titled "Introduction to the C++ Toolkit" gives an overview of the C++ Toolkit. This chapter contains links to other chapters containing more details on a specific topic and is a good starting point for the new comer.

The DOXYGEN source browser is used to complement the traditional docs with structured DOXYGEN-generated "Library Reference" ones based on the in-source comments. You can access the DOXYGEN source browser for the NCBI code from:

http://www.ncbi.nlm.nih.gov/IEB/ToolBox/CPP_DOC/doxyhtml/

The above link is also available under the "Browsers" that appears on each page.

You can also access the CVS code repository via a Web interface. These links also appear in the sidebar box on each page.

A C/C++ Symbol Search query appears on each page of the online Toolkit documentation. You can use this to perform a symbol search on the public or in-house versions of LXR, Doxygen and Library. The Library search finds the library(es) where the symbol (such as function) is defined in.

Building on MacOS

We now build all the libraries and most of the applications including the Genome Workbench (gbench), some test and a few demo applications. We also build the FLTK library's GUI editor, fluid.

All apps are built as application bundles except gbench_plugin_scan and datatool which are built as command-line apps. Any of the applications can be built as command line apps by tweaking the build scripts or the CodeWarrior projects.

GCC

Uses a regular Unix build pattern: run configure, then make.

The NCBI C++ Toolkit Book

The NCBI C++ Toolkit Book

Xcode

When building the toolkit with Xcode, the latest version of Xcode (at least 1.5) is required for the trouble-free build. Build procedure is as follows: open, build and run a project file in compilers/xCode. This is a GUI tool to generate a new NCBI C++ Toolkit Xcode project. You'll have an option to specify third-party installation directories and choose which packages (libs, applications and tests) to include into the final project. The option to automatically download and install all the third-party libraries is also available.

Xcode build fully supports all the latest Apple innovations: distributed builds, optional CPU specific optimization, pre-compiled headers, fix & continue, code cleanup and Zero Link (with few exceptions).

Xcode build has a Shell Script build phase for each Target dependent on generated ASN files. These shell scripts use datatool to regenerate source files each time the ASN specification files do change.

Xcode builds all libraries as a Mach-O dynamically linked shared ones (.dylib) and all Genome Workbench plugins as Mach-O bundles (also .dylib extension). Note, that Xcode will place Genome Workbench plugins inside Genome Workbench application bundle (Genome Workbench.app/Contents/MacOS/plugins).

CodeWarrior

To build the toolkit with CodeWarrior use an AppleScript editor to open and run the script files makeLibs.met and makeApps.met. You must use a script editor capable of opening a script file larger than 32K, such as Apple's Script Editor v2.0, or Smile. Script Editor v1.9 will not work since makeLibs.met just got too big. The command-line tool osascript also works.

On running the scripts you will be prompted as to which targets you want to build. Or if you always build the same targets they can be specified by including an empty file or folder in the compilers:mac_prj folder with the name 'Build' followed by the keywords of the targets you want built. The keywords are: Debug and Final. For example, to build only the Debug targets use: "Build Debug", to build both debug and release (final) versions: "Build".

If you install the C++ Toolkit under a different name than "ncbi_cxx" or in a different location than your home directory, you can edit the script's properties, pRootFolderName and pRootFolderPath, to override these defaults. Note: these paths, and those mentioned below, must be entered in a Mac format (e.g. disk:Users:username:) not in Unix format (e.g. /Users/username/). The disk name (and its following colon) may be omitted.

Certain third party libraries (see Table 1) are required to build some parts of the C++ toolkit. The scripts will try and find them if they are in your home directory, or you can specify where they were installed using properties at the beginning of the script.

Table 1. Third Party Libraries

Library	Property	Example
FLTK 1.1.6 (w/ NCBI patches)	pFLTKRootFolder	" home:mhome:fltk-1.1.6-ncbi3"
BerkeleyDB 4.2 (or 4.3)	pBdbRootFolder	"Users:myhome:mylibs:db-4.2.52" (or "... db-4.3.21")
SQLite 2.8.13	gSqliteFolder	

The NCBI C++ Toolkit Book

The NCBI C++ Toolkit Book

You do not have to build the FLTK or BDB libraries separately. This is done by the scripts and CodeWarrior along with the Toolkit libraries. Just unpack the source bundles in your home directory or where ever you have specified in the appropriate properties. The root folders for FLTK and BDB do not have to have any particular names. If there is more than one version the scripts will grab whichever is last alphabetically (e.g. fltk-1.1.4r2 will get used instead of fltk-1.1.3).

The scripts normally halt on any CodeWarrior compilation errors. If you want them to continue and save errors, set the next script property, pSaveContinueOnErrors, to true. Compilation errors for a project will be saved in a file in the same folder as the project being built, with a name in the following format: projectName-targetNumber.errs (e.g. xncbi-2.errs).

The Genome Workbench's configuration file(s) is stored in the user's Library:Application Support:gbench folder.

CFM builds are not supported. OS 8 or 9 are not supported. We know 10.3 works. We think 10.1 still works, and 10.2 might work.

Platforms (OS's, compilers used inside NCBI)

This release was successfully tested on at least the following platforms -- but may also work on other platforms. Since the previous release, some platforms were dropped from this list, just because we do not use them here anymore, and some were added (these new platforms are highlighted using bold font). Also, it can happen that some projects would not work (or even compile) in the absence of 3rd-party packages, or with older or newer versions of such packages -- in these cases, just skipping such projects (e.g. using flag "-k" for make on UNIX), can get you through.

Unix

Table 2. Unix OS's and Supported Compilers

Operating System	Architecture	Compilers
Linux-2.4.23 (LIBC 2.3.2)	INTEL	GCC 3.4.0, 3.0.4, 2.95.3
Linux-2.4.23 (LIBC 2. 3.2)	INTEL	ICC 8.0
Linux-2.6. 11 (LIBC 2.3.3)	INTEL/64	GCC 3.4.3, 4.0.1
Linux-2.6. 11 (LIBC 2.3.3)	INTEL/64	ICC 9.0
Solaris-8	SPARC	Sun C++ 5.3 (WorkShop 6 update 2) Patch 111685-21 (64-bit, 32-bit)
Solaris-8	SPARC	GCC 3.4.3
Solaris- 10	SPARC	Sun C++ 5.5 Patch 113817-13 (64-bit, 32-bit)
Solaris-9	INTEL	Sun C++ 5.3 (WorkShop 6 update 2) Patch 111685-13
Solaris-9	INTEL	GCC 3.4.3
IRIX64-6.5	SGI-Mips	MIPSpro 7.3.1.3m (64-bit, 32-bit)
FreeBSD-4.10	INTEL	GCC 3.4.2
Tru64 (OSF1) V5.1	ALPHA	GCC 3.3.2

The NCBI C++ Toolkit Book

The NCBI C++ Toolkit Book

MS Windows

Table 3. MS Windows and Supported Compilers

Operating System	Compilers
MS Windows	MSVC++ .NET (7.1). See documentation for building the Toolkit with MS Visual C++ .NET. NOTE: We also have 3rd-party packages archive for this platform, easily built with MSVC++ .NET (7.1).

Mac OS X

Table 4. Mac OS, and Supported Compilers

Operating System	Compilers
MacOS 10.3	GCC 3.3
MacOS 10.3	CodeWarrior 9.2
MacOS 10.3 (PowerPC, Intel)	Xcode 1.5 - 2.2

Caveats and Hints

MacOS 10.X / CodeWarrior 9.2

1. Not all of the test or demo applications are built.
2. The source code for the latest release of FLTK (1.1.x), BerkeleyDB (4.x) and SQLite (2.x) should be present. See the installation instructions for details.

MacOS 10.2/GCC 3.3

At least the GCC 3.3 update for Dec. 2002 Developers Tools required from Apple.

GCC 2.95

1. Poor MT-safety record.
2. Relatively incomplete/incorrect (comparing to modern compilers) STL implementation.
3. It is going to be deprecated in NCBI rather soon -- as soon as we have any significant trouble with its maintenance.

GCC 3.0.4

1. Destructor of constructed class member is not called when exception is thrown from a method called from class constructor body (fixed in GCC 3.3).
2. STL stream uses locale in thread unsafe way which may result to segmentation fault when run in multithread mode (fixed in GCC 3.3).
3. Long-file support for C++ streams is disabled/broken (first broken in 3.0, fixed in 3.4).

GCC 3.3

Other than the feature described below, GCC 3.3.2 had been very good to us; it had a lot of very ugly bugs finally fixed.

1. Painfully slow linking in debug mode on Linux with GCC-3.3 compiler. -- Starting with binutils 2.12 linker tries to merge constant/debug strings marked for merging in object files. But it seems it does this job very inefficiently - I've seen messages about it in internet. GCC starting with version 3.2 marks section of string constants ready for merging, and also has an option to disable this flag in object files (-fno-merge-

The NCBI C++ Toolkit Book

The NCBI C++ Toolkit Book

constants). Adding this flag to compilation stage allows to avoid slow linking. GCC 3.3 also sets merge flag for debug sections and unfortunately there is no option to disable this flag. As a result, linking of debug executables significatly slower than with gcc 3.0.4. The slowdown rate depends on size of debug strings section and it's non-linear, so bigger projects will suffer more of this bug (N^2). Binutils 2.15 fixes this. The link time still 2 times slower than without symbol merge, but the resultant executable is about two times smaller in size, and no compiler patching is necessary. We are still testing it in-house. We had to patch GCC 3.3 in-house with the fix described at http://lists.boost.org/MailArchives/boost/msg53004.php.

2. Long-file support still broken.

GCC 3.4.x, 4.0.x

1. The "Painfully slow linking..." (see GCC3.3,! [1] above) was an issue, and we had to patch it in-house to speed up, a la GCC 3.3 -- until we finally upgraded to binutils 2.15.
2. At least on Linux, ifstream::readsome() does not always work for large files, as it calls an ioctl that doesn't work properly for large files (we didn't test whether 4.0.x fixed this).
3. At least on Linux, GCC 3.4.[0,1] optimizer (very rarely) generates incorrect code when comparing enumerated values in else-ifs. (Fixed in 3.4.2)
4. GCC 3.4.3, 3.4.4 and 4.0 have a bug in the C++ stream library that affects some parts of our code, notably CGI framework. (Fixed in 4.0.1)

Last Updated

This section last updated on October 04, 2005.