What's New

Forte Developer 7

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054 U.S.A.
650-960-1300

Part No. 816-2452-10
May 2002, Revision A

Send comments about this document to: docfeedback@sun.com

Contents

Before You Begin

The *What's New* describes the new features of this Forte™ Developer 7 compilers and command-line tools release.

Typographic Conventions

Typeface	Meaning	Examples
`AaBbCc123`	The names of commands, files, and directories; on-screen computer output	Edit your `.login` file. Use `ls -a` to list all files. `% You have mail.`
`AaBbCc123`	What you type, when contrasted with on-screen computer output	`% su` `Password:`
AaBbCc123	Book titles, new words or terms, words to be emphasized	Read Chapter 6 in the *User's Guide*. These are called *class* options. You *must* be superuser to do this.
AaBbCc123	Command-line placeholder text; replace with a real name or value	To delete a file, type **`rm`** *filename*.

Shell Prompts

Shell	Prompt
C shell	%
Bourne shell and Korn shell	$
C shell, Bourne shell, and Korn shell superuser	#

Accessing Forte Developer Development Tools and Man Pages

The Forte Developer product components and man pages are not installed into the standard `/usr/bin/` and `/usr/share/man` directories. To access the Forte Developer compilers and tools, you must have the Forte Developer component directory in your `PATH` environment variable. To access the Forte Developer man pages, you must have the Forte Developer man page directory in your `MANPATH` environment variable.

For more information about the `PATH` variable, see the `csh`(1), `sh`(1), and `ksh`(1) man pages. For more information about the `MANPATH` variable, see the `man`(1) man page. For more information about setting your `PATH` and `MANPATH` variables to access this Forte Developer release, see the installation guide or your system administrator.

Note – The information in this section assumes that your Forte Developer products are installed in the `/opt` directory. If your product software is not installed in the `/opt` directory, ask your system administrator for the equivalent path on your system.

Accessing Forte Developer Compilers and Tools

Use the steps below to determine whether you need to change your `PATH` variable to access the Forte Developer compilers and tools.

▼ To Determine Whether You Need to Set Your PATH Environment Variable

1. **Display the current value of the PATH variable by typing the following at a command prompt:**

```
% echo $PATH
```

2. **Review the output for a string of paths that contain /opt/SUNWspro/bin/.**

   If you find the path, your PATH variable is already set to access Forte Developer development tools. If you do not find the path, set your PATH environment variable by following the instructions in the next section.

▼ To Set Your PATH Environment Variable to Enable Access to Forte Developer Compilers and Tools

1. **If you are using the C shell, edit your home .cshrc file. If you are using the Bourne shell or Korn shell, edit your home .profile file.**

2. **Add the following to your PATH environment variable:**

   /opt/SUNWspro/bin

Accessing Forte Developer Man Pages

Use the following steps to determine whether you need to change your MANPATH variable to access the Forte Developer man pages.

▼ To Determine Whether You Need to Set Your MANPATH Environment Variable

1. **Request the dbx man page by typing the following at a command prompt:**

```
% man dbx
```

2. **Review the output, if any.**

   If the dbx(1) man page cannot be found or if the man page displayed is not for the current version of the software installed, follow the instructions in the next section for setting your MANPATH environment variable.

▼ To Set Your MANPATH Environment Variable to Enable Access to Forte Developer Man Pages

1. **If you are using the C shell, edit your home `.cshrc` file. If you are using the Bourne shell or Korn shell, edit your home `.profile` file.**
2. **Add the following to your `MANPATH` environment variable:**

   `/opt/SUNWspro/man`

Accessing Forte Developer Documentation

You can access Forte Developer product documentation at the following locations:

- The product documentation is available from the documentation index installed with the product on your local system or network at `/opt/SUNWspro/docs/index.html`.

  If your product software is not installed in the `/opt` directory, ask your system administrator for the equivalent path on your system.

- Most manuals are available from the `docs.sun.com`sm web site. The following titles are available through the documentation index in your installed product (and not through the `docs.sun.com` web site):
  - *Standard C++ Library Class Reference*
  - *Standard C++ Library User's Guide*
  - *Tools.h++ Class Library Reference*
  - *Tools.h++ User's Guide*

  The `docs.sun.com` web site (`http://docs.sun.com`) enables you to read, print, and buy Sun Microsystems manuals through the Internet. If you cannot find a manual, see the documentation index installed with the product on your local system or network.

Note – Sun is not responsible for the availability of third-party web sites mentioned in this document and does not endorse and is not responsible or liable for any content, advertising, products, or other materials on or available from such sites or resources. Sun will not be responsible or liable for any damage or loss caused or alleged to be caused by or in connection with use of or reliance on any such content, goods, or services available on or through any such sites or resources.

Product Documentation in Accessible Formats

Forte Developer 7 product documentation is provided in accessible formats that are readable by assistive technologies for users with disabilities. You can find accessible versions of documentation as described in the following table. If your product software is not installed in the `/opt` directory, ask your system administrator for the equivalent path on your system.

Type of Documentation	Format and Location of Accessible Version
Manuals (except third-party manuals)	HTML at `http://docs.sun.com`
Third-party manuals: • *Standard C++ Library Class Reference* • *Standard C++ Library User's Guide* • *Tools.h++ Class Library Reference* • *Tools.h++ User's Guide*	HTML in the installed product through the documentation index at `file:/opt/SUNWspro/docs/index.html`
Readmes and man pages	HTML in the installed product through the documentation index at `file:/opt/SUNWspro/docs/index.html`
Release notes	Text file on the product CD at `/cdrom/devpro_v10n1_sparc/release_notes.txt`

Sending Your Comments

Sun is interested in improving its documentation and welcomes your comments and suggestions. Email your comments to Sun at this address:

`docfeedback@sun.com`

Forte Developer 7 New Features

This chapter describes the new features of the Forte™ Developer 7 compilers and command-line tools. The primary focus of this release is improved performance, additional OpenMP support, and the ability to debug code written in the Java™ programming language (Java code) as well as in the C programming language, C++ programming language, and Fortran programming language.

This chapter has the following sections:

- C Compiler
- C++ Compiler
- Fortran Compiler
- `dbx` Command-Line Debugger
- Sun Performance Library
- Interval Arithmetic
- Performance Analysis Tools
- Documentation

In most sections, there is a table that lists the new features of that component. The table has either two columns or three columns:

- **Two-column table.** The left-hand column provides a short description of the feature, and the right-hand column has a longer description.
- **Three-column table.** The left-hand column provides a short description of the feature, the middle column lists the relevant command or option, and the right-hand column has a longer description of the new feature.

Note – To find the Forte Developer documentation described in this chapter, see the documentation index installed with the product software at `/opt/SUNWspro/docs/index.html`. If your software is not installed in the `/opt` directory, contact your system administrator for the equivalent path on your system or network.

C Compiler

TABLE 1 lists the new features for this release of the C compiler. For more information about each of the new features listed in TABLE 1, see the index of the *C User's Guide*.

TABLE 1 C Compiler New Features

Feature	Option	Description
Enhanced interprocedural optimizations (SPARC)	`-xipo=2`	Performs inlining across all source files or performs interprocedural aliasing analysis as well as optimizations of memory allocation and layout to improve cache performance.
New prefetch levels (SPARC)	`-xprefetch_level=`*n*	Allows you to control the aggressiveness of the automatic insertion of prefetch instructions as determined by `-xprefetch=auto`.
Stack-overflow checking (SPARC)	`-xcheck=stkovf`	Adds a runtime check for stack overflow of the main thread in a singly-threaded program as well as slave-thread stacks in a multithreaded program.
Labels as values (computed `goto`)		Enables runtime determination of branching destinations. The address of a label can be acquired by using the && operator and assigned to a pointer of type `void *`.
Support for additional C99 features	`-xc99`	Enables support of various implemented C99 features.

TABLE 1 C Compiler New Features *(Continued)*

Feature	Option	Description
Improved formatting control of error messages	`-errfmt` `-errshort`	The compiler now offers two new commands that you can use to change the formatting of error messages: `-errfmt` and `-errshort`.
Support for UltraSPARC IIICu processor		The compiler's `-xtarget`, `-xchip`, and `-xcache` options support the UltraSPARC® IIICu processor.
Compiling for the Native Connector Tool	`-xnativeconnect`	Includes interface information inside object files and subsequent shared libraries so that the shared library can interface with Java code. The Native Connector Tool (NCT) enables the automatic generation of Java code and the Java™ Native Interface so that C shared libraries can be called from Java code.

C++ Compiler

TABLE 2 lists the new features available with this release (version 5.4) of the C++ compiler. All of the features are described in the *C++ User's Guide*.

TABLE 2 C++ Compiler New Features

Feature	Option	Description
Support for OpenMP (SPARC)	`-xopenmp`	Implements the OpenMP interface for explicit parallelization, including a set of source code directives, run-time library routines, and environment variables.
Type-based alias analysis and optimizations (SPARC)	`-xalias_level`	Allows the compiler to perform type-based alias analysis and optimizations.
Compiling for the Native Connector Tool	`-xnativeconnect`	Includes interface information inside object files and subsequent shared libraries so that the shared library can interface with Java code. The Native Connector Tool enables the automatic generation of Java code and the Java Native Interface so that C++ shared libraries can be called from Java code.
Enhanced interprocedural optimizations (SPARC)	`-xipo=2`	Performs inlining across all source files or performs interprocedural aliasing analysis as well as optimizations of memory allocation and layout to improve cache performance.
New prefetch levels (SPARC)	`-xprefetch_level=`*n*	Allows you to control the aggressiveness of the automatic insertion of prefetch instructions as determined by `-xprefetch=auto`.
Stack-overflow checking (SPARC)	`-xcheck=stkovf`	Adds a runtime check for stack overflow of the main thread in a singly-threaded program as well as slave-thread stacks in a multithreaded program.
Support for STLport Standard Library implementation version 4.5.2	`-library=stlport4`	Supports STLport's Standard Library implementation version 4.5.2.

TABLE 2 C++ Compiler New Features *(Continued)*

Feature	Option	Description
`+w` behavior change	`+w`	No longer reports that a function is too large to inline and no longer reports that parameters are unused in order to reduce output messages and simplify the use of the `+w` option in routine builds. These warnings are still available with the `+w2` option.
`+w2` behavior change	`+w2`	No longer reports the use of implementation-dependent constructs in the system header files in order to simplify the use of the `+w2` option in routine builds.
`#error` directive aborts compilation		Issues an error message and immediately halts compilation. The compiler quits and reports the failure.

Fortran Compiler

The Forte Developer 7 release provides a Fortran 95 compiler, `f95`, with compatibility support for legacy Fortran 77 programs. See the chapter "FORTRAN 77 Compatibility: Migrating to Fortran 95" in the *Fortran User's Guide* for details on porting legacy Fortran 77 programs to the Fortran 95 compiler.

TABLE 3 lists the new features of the Fortran 95 compiler. See the *Fortran User's Guide*, *Fortran Programming Guide*, and the *Fortran Library Reference* for details.

TABLE 3 Fortran 95 Compiler New Features

Feature	Option	Description
Fortran 77 compatibility flag	`-f77`	Accepts many Fortran 77 constructs that are normally incompatible with Fortran 95.
Forcing implicit USE statements	`-use`	Forces one or more implicit USE statements into each subprogram or module subprogram compiled with the `-use` flag.
Enhanced `MODULE` features	`-moddir=`*directory-name*	Controls where `f95` writes compiled `MODULE` subprograms (`.mod` files). There is a new `MODDIR` environment variable. The `-M` option now accepts an archive (`.a`) file or module (`.mod`) file as an argument, as well as a directory name. The way the compiler searches for modules has also changed, and the way the compiler records module and include file dependences in makefiles with the `.KEEP_STATE` target has also improved.
`-Xlist` enhancements	`-XlistMP`	Provides verification of OpenMP directives. Global program analysis with the `-Xlist` flag has a number of new checks.
Identify known libraries	`-xknown_lib=`*keywords*	Directs the compiler to treat references to certain known libraries as intrinsics, ignoring any user-supplied versions. This enables the compiler to perform optimizations over calls to library routines based on special knowledge of the library.
Ignoring dummy argument type in interfaces	`!$PRAGMA IGNORE_TKR {`*name* `{, ` *name*`} ...}`	Causes the compiler to ignore the type, kind, and rank of the specified dummy argument names appearing in a generic procedure interface when resolving a specific call.

TABLE 3 Fortran 95 Compiler New Features *(Continued)*

Feature	Option	Description
`-xalias` for permitting nonstandard aliasing	`-xalias`	Advises the compiler about how far the program deviates from the aliasing restrictions required by the Fortran 95 standard. Useful when porting legacy Fortran 77 code that utilizes nonstandard programming techniques.
Enhancements to `-C` runtime checking	`-C`	Now also performs runtime array conformance checking.
The `BYTE` data type		Accepts the `BYTE` data type and treats `BYTE` as `INTEGER *1`.
Fortran 2000 features		New formatted I/O specifiers implemented in `f95`: `DECIMAL=`, `ROUND=`, and `IOMSG=`.
Rounding in formatted I/O	`-iorounding`	Sets the default rounding mode for formatted I/O to `processor-defined` or `compatible`. These modes correspond to the `ROUND=` specifier described in the Fortran 2000 features in this table.
Stack-overflow checking (SPARC)	`-xcheck=stkovf`	Adds a runtime check for stack overflow of the main thread in a single-threaded program as well as slave-thread stacks in a multithreaded program.
New default thread stack size		Increased to 4 megabytes on SPARC V8 platforms and 8 megabytes on SPARC V9 platforms.
Enhanced interprocedural optimizations (SPARC)	`-xipo=`*n*	Performs inlining across all source files or performs interprocedural aliasing analysis as well as optimizations of memory allocation and layout to improve cache performance.
New prefetch levels (SPARC)	`-xprefetch_level=`*n*	Allows you to control the automatic insertion of prefetch instructions.

dbx Command-Line Debugger

TABLE 4 lists the new features in this release of the dbx command-line debugger. For more information about these features, see the *Debugging a Program With* dbx manual.

TABLE 4 dbx New Features

Feature	Description
Debug programs with mixed-language code	The dbx command-line debugger can now debug applications running in the Solaris operating environment that are a mixture of Java code and C code or C++ code using the Java Native Interface (JNI). The JNI allows Java code and applications and libraries written in native languages to operate together.
Changes to dbx collector command	You can set the approximate limit to the amount of data collected. Address space data is no longer supported. There are subcommands to record a sample and to control whether dbx records a sample when it stops the target process. For more information about new features of the performance analysis tools, see TABLE 7.
dbx environment variables removed	The following environment variables have been removed: allow_critical_exclusion, aout_cache_size, locache_enable, and delay_xs.
Changes to loadobject command	There is a new loadobject command to control and list the symbols loaded by dbx. The new command replaces both the previous loadobject and loadobjects commands.
Change to default dbx prompt	The default dbx prompt is now the last path component of argv[0] instead of the entire string in argv[0].

Sun Performance Library

Sun Performance Library™ is a set of optimized, high-speed mathematical subroutines for solving linear algebra problems and other numerically intensive problems. Sun Performance Library is based on a collection of public domain applications available from Netlib (at `http://www.netlib.org`). These routines have been enhanced and bundled as the Sun Performance Library.

TABLE 5 lists the new features in this release of the Sun Performance Library. See the *Sun Performance Library User's Guide* and the section 3p man pages for more information.

TABLE 5 Sun Performance Library New Features

Feature	Description
New FFT interfaces supersede a subset of FFTPACK and VFFTPACK Routines	See the `fft`(3p) man page and the *Sun Performance Library User's Guide* for descriptions of the new FFT routines. FFTPACK and VFFTPACK routines are included with this version of Sun Performance Library, but they are no longer supported.
Interval BLAS routines added	Interval BLAS routines operate on interval scalars, interval vectors, and interval matrices (dense, banded, and triangular).
Sort and permute routines added	The following routines have been added: `blas_dsort`, `blas_isort`, `blas_ssort`, `blas_dsortv`, `blas_isortv`, `blas_ssortv`, `blas_dpermute`, `blas_ipermute`, and `blas_spermute`.
Select sparse BLAS routines have been parallelized	All of the sparse matrix-matrix multiply (`*mm`) routines and the sparse triangular solve (`*sm`) routines have been parallelized. New sparse matrix-matrix multiply and sparse triangular solve routines have also been added to this release.

TABLE 5 Sun Performance Library New Features *(Continued)*

Feature	Description
Sparse solver support for real, complex, and double complex data types	Sparse solver routines provide support for real, complex, and double complex data types.
`LINPACK` removed from Sun Performance Library	`LINPACK` is no longer included in the library. `LAPACK` version 3.0 supersedes `LINPACK` and all previous versions of `LAPACK`. If code that calls `LINPACK` routines cannot be modified to use `LAPACK` routines, the public domain version of `LINPACK` can still be obtained from Netlib (`http://www.netlib.org/linpack`).
Legacy `so.2` and `so.3` libraries not included with Sun Performance Library	To reduce the size of the Sun Performance Library for Forte Developer 7, only the `so.4` libraries have been included, and the legacy `so.2` and `so.3` libraries have been removed from this release.

Interval Arithmetic

TABLE 6 lists the Forte Developer 7 Interval Arithmetic new features.

TABLE 6 Interval Arithmetic New Features

Feature	Description
Interval BLAS routines added to Sun Performance Library	Interval `BLAS` routines operate on interval scalars, interval vectors, and interval matrices (dense, banded, and triangular). For more information on the Interval BLAS routines, see the *Sun Performance Library User's Guide*.
Two new template classes added to C++ interval arithmetic	A C++ interface to the C++ interval arithmetic library is extended with the `nvector` and `nmatrix` template classes that operate on interval vectors and interval matrices. See the `nvector`(3C++) and `nmatrix`(3C++) man pages for more information.
Interval-specific GPC functions added	Several interval-specific checks are implemented in GPC.

Performance Analysis Tools

TABLE 7 lists the new data collection and presentation features in the Forte Developer 7 release of the performance analysis tools. For more information, see the following man pages:

- `collect`(1)
- `collector`(1)
- `er_print`(1)
- `libcollector`(3)

TABLE 7 Performance Analysis Tools New Features

Feature	Command and Option	Description
Simultaneous clock-based profiling and hardware-counter profiling		Collects clock-based profiling data and hardware-counter profiling data simultaneously.
Low-resolution profiling options added		Reduces the amount of data collected on executables that run for a long time.
Periodic sampling available with the `collect` command	`collect -S`	Collects periodic samples at the interval specified.
New sampling controls in `dbx`	`dbx collector dbxsample` `dbx collector sample record`	Records a sample and controls whether `dbx` records a sample when it stops the target process.
Approximate limit to the amount of data collected	`collect -L` `dbx collector limit`	Limits the amount of profiling data and tracing data recorded to a specified number of megabytes.
Record profiling data for Java programs through the `collect` command		Records profiling data on Java methods compiled by the Java HotSpot™ virtual machine.

TABLE 7 Performance Analysis Tools New Features *(Continued)*

Feature	Command and Option	Description
Experiments on descendant processes recorded	`collect -F`	Controls whether or not descendant processes should have their data recorded. Performance data can be collected on descendant processes created by calls to `fork`(2), `fork1`(2), `vfork`(2), `fork`(3F), and `exec`(2) and its variants. Only available from the `collect` command.
Fortran API to collector library		Has Fortran API subroutines for recording samples, pause and resume, and closing an experiment.
Support for dynamically compiled functions		C and C++ API functions provide data to the collector on dynamically compiled functions and dynamically loaded modules.
Tracing of memory allocations and deallocations (heap tracing) recorded	`collect -H` `dbx collector heaptrace`	Collects heap tracing data. Data is converted into allocation and leak metrics. Display of metrics is supported by the `leaks` and `allocs` commands in `er_print`.
Tracing of MPI calls separated from synchronization delay tracing	`collect -m` `dbx collector mpitrace`	Collects more data on the MPI calls that are traced and displays them separately from synchronization delay data.
New meaning for `collect -n`		Prints details of the experiment that would have been run. It no longer runs the target program.
Address space data collection unavailable	`collect -a` `dbx collector address_space`	The collection and display of address space data is no longer supported.
Experiments no longer kept in a hidden directory		The experiment name is now the name of an ordinary UNIX® software directory. Older versions of the tools can not read newer experiments.

TABLE 7 Performance Analysis Tools New Features *(Continued)*

Feature	Command and Option	Description
Improved robustness of data collection		When an application installs a signal handler, the collector re-installs its signal handler as the primary handler and passes signals to the application's handler so that profiling signals are not lost. The collector also prevents an application from using the hardware counters if hardware-counter overflow profiling is enabled and ensures that a application cannot interfere with clock-based profiling.
Performance tools readme available from the command line	`collect -R`	Displays the text version of the performance tools readme file.
Changed handling of load objects	`er_print object_list` `er_print object_select`	Aggregates and presents data for the load object as a whole instead of removing data for a load object.
New commands to set thresholds for highlighting high-metric code lines	`er_print sthresh` `er_print dthresh`	Sets the thresholds for flagging important lines in annotated source code and annotated disassembly code.
Function and callers-callees data printed	`er_print fsingle` `er_print csingle`	Prints function summary and callers-callees data for a single function.

Documentation

This section describes Forte Developer 7 documentation new features.

- The *OpenMP API User's Guide* is a new manual for this release. This manual summarizes the OpenMP directives and pragmas accepted by the compilers, describes implementation details, and provides guidelines for converting legacy Sun parallelization directives to OpenMP.

  The OpenMP Application Program Interface (API) supports multiplatform shared-memory parallel programming in C, C++, and Fortran. OpenMP is a portable, scalable model that gives programmers an explicit and flexible interface for developing parallel applications on shared-memory mulitprocessor platforms. In this release, the C compiler, C++ compiler, and Fortran 95 compiler feature full native support on SPARC® platforms for the OpenMP C/C++ API Version 1.0 and Fortran 95 API Version 2.0 specifications. For PDF versions of the specifications, go to the `http://www.openmp.org` web site.

- The *Fortran User's Guide* has a new chapter entitled "FORTRAN 77 Compatibility: Migrating to Fortran 95" that provides details on porting legacy Fortran 77 programs to the Fortran 95 compiler.
- The *Debugging a Program With* `dbx` manual has a new chapter entitled "Debugging a Java Application With `dbx`." The chapter describes how you can use the `dbx` command-line debugger to debug applications running in the Solaris™ operating environment that are a mixture of Java code and C code or C++ code using the Java™ Native Interface (JNI). The JNI allows Java code and applications and libraries written in native languages to operate together.
- The *Analyzing Program Performance* manual is now entitled *Program Performance Analysis Tools*.
- The following manuals have been discontinued and were not updated for this release:
  - *Sun WorkShop 6 update 2 Installation and Licensing Reference* (see the *Forte Developer 7: Installation Guide* for installation and licensing information)
  - *About Sun WorkShop 6 update 2 Documentation*
  - *Introduction to Sun WorkShop*
- Previous releases of the *FORTRAN 77 Language Reference* are available through the `docs.sun.com` web site. This manual has not been updated for this release.
- Forte Developer 7 product documentation is provided in formats that are readable by assistive technologies for users with disabilities. For more information, see "Product Documentation in Accessible Formats" on page ix.