Oracle® Developer Studio 12.5 リリースの新機能

Part No: E71943
2016 年 7 月

ORACLE®

Part No: E71943

ドキュメントのアクセシビリティについて

オラクルのアクセシビリティについての詳細情報は、Oracle Accessibility ProgramのWeb サイト(http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc)を参照してください。

Oracle Supportへのアクセス

サポートをご契約のお客様には、My Oracle Supportを通して電子支援サービスを提供しています。詳細情報は(http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info)か、聴覚に障害のあるお客様は (http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs)を参照してください。

目次

このドキュメントの使用法

- **概要** – Oracle Developer Studio 12.5 のこのリリースにおけるコンパイラとツールの新機能や変更点について説明します。
- **対象読者** – アプリケーション開発者、システム開発者、設計者、サポートエンジニア
- **必要な知識** - プログラミング経験、ソフトウェア開発テスト、ソフトウェア製品を構築およびコンパイルできる能力

製品ドキュメントライブラリ

この製品および関連製品のドキュメントとリソースは http://docs.oracle.com/cd/E60778_01 で入手可能です。

フィードバック

このドキュメントに関するフィードバックを http://www.oracle.com/goto/docfeedback からお聞かせください。

♦ ♦ ♦ 第 1 章

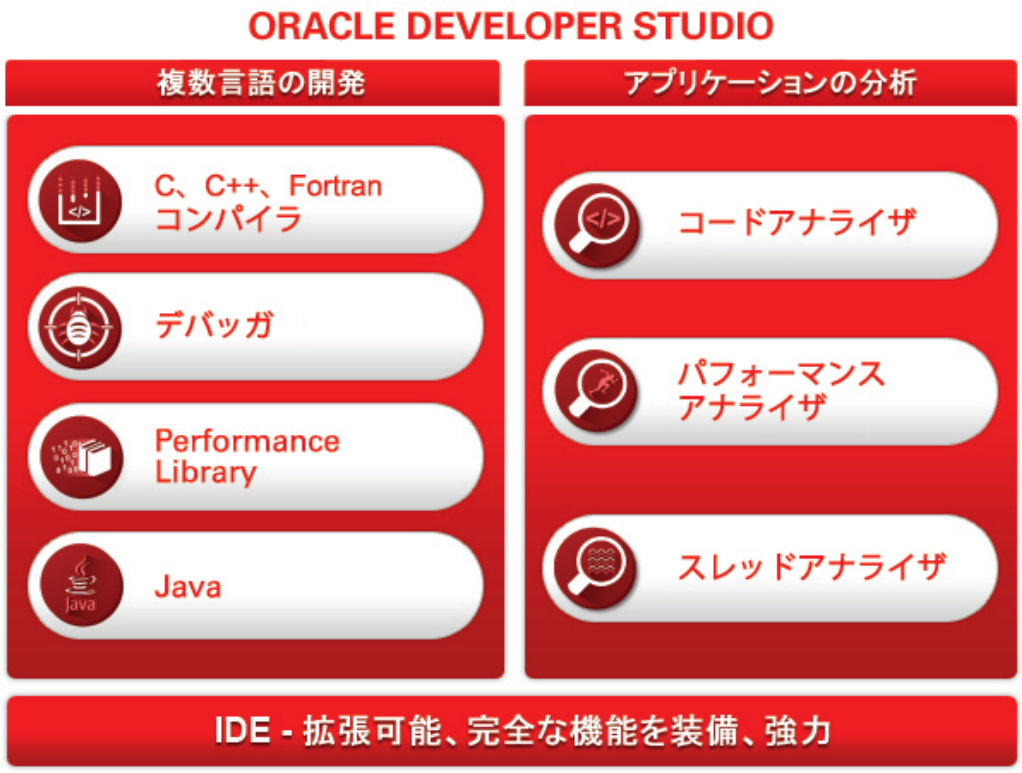
1 Oracle Developer Studio 12.5 リリースの紹介

この章では、このリリースの重要な更新の概要について説明します。

Oracle Developer Studio の概要

Oracle Developer Studio には、コンパイラスイート、分析スイート、および両方のスイートからコンパイラおよびツールとともに使用するように調整されたグラフィカル統合開発環境 (IDE) が組み込まれています。これらを組み合わせることで、Oracle Sun ハードウェア上で最高のパフォーマンスを発揮するアプリケーションの開発用に最適化された開発環境が提供されます。

図 1 IDE に統合されたコンパイラスイートおよび分析スイートを示す図



このリリースでの主な特長

Oracle Developer Studio 12.5 は、Oracle Solaris および Linux オペレーティングシステム用の高速で、信頼性の高い、セキュアなアプリケーションを容易に開発するのための高度に最適化されたコンパイラ、高度な分析ツール、および複数言語に対応した IDE を提供します。Oracle Developer Studio ツールは、完全なハードウェアおよびソフトウェアスタックを補完するように最適化されているため、開発チームはより適切なコードをより迅速に記述できます。

- **セキュリティーのレベルの向上**
  - 開発者ツール内の Oracle SPARC M7 Silicon Secured Memory (SSM) 統合はリアルタイムのメモリーアクセス検査を可能にし、新しいライブラリによりアプリケーション

はカスタムメモリーアロケータで SSM を使用できます。詳細は、第4章「コード分析ツール」を参照してください。

- ソースコードのコンパイル中の組み込みセキュリティーチェックは、開発およびテストプロセスの早い段階でセキュリティーの脆弱性を識別するのに役立ちます。詳細は、第2章「C++ コンパイラ」、第3章「C コンパイラ」、および 45 ページの「コンパイラに対する変更」を参照してください。
- アプリケーション実行時の自動スタックオーバーフロー保護により、潜在的なセキュリティーの脆弱性が最小限に抑えられます
- セキュアでない非推奨の機能の使用を検出し、よりセキュアな代替機能についての提案を提供するための、IDE でのセキュアなコーディングのヒント。詳細は、第7章「Oracle Developer Studio IDE」を参照してください。

■ **生産性の向上**

- 拡張された GNU サポートとバイナリ互換性による、オープンソースアプリケーションの容易なコンパイル。詳細は、『Oracle Developer Studio 12.5: GCC 互換性ガイド』を参照してください。
- 一般的な C++14 機能と、並列性や不可分ライブラリを含む C++11 および C11 の完全なサポート
- 機能強化されたテストおよび最新の Boost ライブラリのサポート
- Java、C、C++ アプリケーションの正確で、容易なパフォーマンス分析。詳細は、第5章「パフォーマンス解析ツール」を参照してください。
- 大規模なエンタープライズアプリケーションの効率的なコード編集により、解析時間がオープンソースアプリケーションに比べて最大 7 倍高速。詳細は、第7章「Oracle Developer Studio IDE」を参照してください。

■ **パフォーマンスの最大化**

- SPARC S7 を含む最新の Oracle システム (SPARC および x86) のためのパフォーマンス最適化により最大 4.5 倍高速なコードを生成します
- 最新の SPARC ハードウェアカウンタで更新されたパフォーマンスアナライザ。詳細は、第5章「パフォーマンス解析ツール」を参照してください。
- すべてのツールが 64 ビットバイナリとして提供されます

♦ ♦ ♦ 第 2 章

2 C++ コンパイラ

この章では、このリリースの Oracle Developer Studio の C++ コンパイラの新機能と変更された機能について説明します。

C++ コンパイラについて

このセクションでは、Oracle Developer Studio 12.5 の C++ 5.14 コンパイラリリースに導入されている新機能や変更点のサマリーを記載します。

C++ コンパイラ (cc) は、指定されたコマンド行オプションに従って、特定のオペレーティングシステム、プロセッサ、アーキテクチャー、メモリーモデル (32 ビットおよび 64 ビット)、浮動小数点演算などを対象とするコードを生成します。コンパイラは、シリアルソースコードに対して自動並列化を実行し、マルチコアシステムで優れたパフォーマンスを発揮するバイナリを生成するほか、ほかの Oracle Developer Studio ツールによる強化されたデバッグまたは分析用にバイナリを作成することもできます。また、コンパイラは GNU C/C++ 互換機能もサポートします。

C++14 標準の部分的なサポート

新しい C++14 標準は C++ を強化して、コードを無駄のない安全なものにすることができる追加のツールを提供します。コンパイラは、Oracle ハードウェア上で SPARC および x86 の高速化されたパフォーマンスを維持します。

これは、C++14 標準のサポートが含まれるはじめての Oracle Developer Studio リリースです。C++14 標準の次の機能がサポートされています。

- バイナリリテラル

- サイズ割り当て解除
- `deprecated` 属性
- 単一引用符の桁区切り文字
- メンバーの初期化と集合体

C++ コンパイラのその他の変更点

次に、C コンパイラに固有のバージョン 5.14 のこのリリースにおける新機能と変更された機能を列挙します。

C++ コンパイラの変更点には、45 ページの「全コンパイラに共通の新機能および変更点」で説明する変更点も含まれます。

詳細は、『Oracle Developer Studio 12.5: C++ ユーザーズガイド』および `cc`(1) のマニュアルページを参照してください。

- **デフォルトコンパイルモードの変更** — Oracle Solaris でのデフォルトのコンパイルモードは、`-library=Cstd` が指定された `-compat=5` (Sun ABI および `libCstd` ライブラリを使用する C++03 モード) です。Linux でのデフォルトのコンパイルモードは、`-std=c++03` (g++ ABI と実行時ライブラリを使用する C++03 モード) です。
- **C++11 標準の機能のサポート**

  Oracle Developer Studio 12.5 C++ は、次の項目を追加して C++11 のサポートを実行します。

  - 並行性および不可分操作

    注記 - 不可分機能を使用する際は、実行時サポートに特に注意を払う必要があります。詳細は、『Oracle Developer Studio 12.5: C ユーザーズガイド』の「バンドルされている不可分ライブラリ」を参照してください。

  - ユーザー定義リテラル
- **コンパイラの新しいオプション:**
  - `-pedantic` — デフォルトでは受け入れられるものの C++ 標準には準拠していないコードに対して警告またはエラーを出力します。
  - `-abiopt=[mangle5|mangle6]` — `-compat=5` モードでのみ使用可能です。デフォルトは、`mangle6` (正しい名前符号化) です。`-m32` オプションを使用する Oracle Solaris SPARC および Oracle Solaris x86 では、古いコンパイラのバグの可能性がある名前符号化との互換性のために `mangle5` を指定できます。
  - `-xcheck=noreturn` — `does_not_return` と記述されているルーチンが復帰した場合に実行時エラーを発生させるコードを追加するようコンパイラに通知します。
  - `-xatomic` は、リンクされる不可分サポート実行時ライブラリを指定します。

♦ ♦ ♦ 第 3 章

3 C コンパイラ

この章では、Oracle Developer Studio 12.5 リリースでの C コンパイラの変更点について説明します。

-xc99 フラグから -std フラグへの移動

C11 のサポートの導入に伴い、言語の文法は C89 と C99 のどちらか一方のバイナリを選択すればよいという単純なものではなくなり、3 つめの選択肢 C11 が加わりました。

Oracle Solaris Studio 12.3 では、選択肢は C99 と C89 のみで、-xc99 フラグによって制御されていました。

フラグ	言語の文法
-xc99=all	C99 言語
-xc99=none	C89 言語

Oracle Solaris Studio 12.4 および Oracle Developer Studio 12.5 では、言語文法の選択肢 (C89、C99、または C11) は -std フラグによって制御されます。

フラグ	言語の文法
-std=c11	C11 言語
-std=c99	C99 言語
-std=c89	C89/C90 言語

-xc99 フラグと -std フラグの簡単なマッピングは次のとおりです。

-xc99 フラグ	-std フラグ
-xc99	-std=c99
-xc99=all	-std=c99
-xc99=all,lib	-std=c99

-xc99 フラグ	-std フラグ
-xc99=all,no_lib	-std=c99 -xlang=c89
-xc99=none,lib	-std=c89 -xlang=c99
-xc99=none,no_lib	-std=c89
-xc99none	-std=c89

言語文法を制御するための -Xc、-Xa、-Xt、-xc99オプションは、Oracle Developer Studio の C コンパイラでは非推奨か、今後非推奨になる予定です。

- -xc99: ISO C99 または C89 言語のどちらかを選択します
- -Xc: ISO C にない構文を使用しているプログラムに対してエラーや警告を発行します
- -Xa: ISO C と、C 言語の拡張機能を受け入れます
- -Xt: ISO C と K&R C 互換性拡張機能を受け入れます
- -Xs: K&R C を受け入れます

代わりに、-std オプションを使用するようにしてください。-Xa を使用していた場合は、-pedantic オプションも一緒に使用してください。-Xt または -Xs を使用するレガシーコードは、ISO C 文法への変換が必要になります。

-xlang オプションを使用すると、標準への準拠に関連する特定の libc 関数の動作を制御できます。Oracle Solaris Studio 12.4 の C コンパイラでは、デフォルトの動作は C11 言語構造および C89 ライブラリ動作でした。このデフォルトモードでは、___STDC_VERSION__ (199409L) は C89 標準を示します。

Oracle Developer Studio 12.5 の C コンパイラでは、言語機能とライブラリ動作の両方がデフォルトで C11 モードとなり、__STDC_VERSION__ (201112L) は C11 を示します。

注記: Oracle Solaris Studio 12.3 では、この動作はサブオプション -xc99=lib によって制御されていました。

Oracle Solaris での C コンパイラの新しいデフォルト

次の機能は、Oracle Solaris での C コンパイラの新しいデフォルトです。

- C コンパイラのデフォルトモードが変更されました
- 新しいデフォルトモードはアプリケーションに影響を与える可能性があります
- 必要に応じて、古いモードも使用できます

__STDC_VERSION__ の変更点

次の変更点は、__STDC_VERSION__ の新機能について説明したものです。

- デフォルトでは、Oracle Solaris での以前のバージョンの C コンパイラは認識しているすべての C99 および C11 機能を受け入れましたが、C89 への準拠は `__STDC_VERSION__` を 199409L に事前に定義することによってのみ示しました。
- 新しい C コンパイラはデフォルトで、`__STDC_VERSION__` を 201112L に事前に定義することで C11 への準拠を要求します。

Oracle Solaris でのユーザーアプリケーションへの影響

次に、Oracle Solaris プラットフォームでのユーザーアプリケーションへの影響について説明します。

- `_XOPEN_SOURCE`、`_POSIX_SOURCE`、`_POSIX_C_SOURCE` などのマクロを使用する、インクルードされたファイルや機能テストは、別の方法で解決する可能性があります。

  たとえば、`_POSIX_SOURCE` を使用した場合、このエラーは次のようになります。

  ```
  Compiler or options invalid for pre-UNIX 03 X/Open applications
  ```

  この問題は、standards(5) のマニュアルページに記載されているとおり、`_POSIX_SOURCE` が理論上は C89 コンパイラを要求していることです。

  使用しているアプリケーションが古いバージョンの標準 (`_POSIX_SOURCE` など) についてテストしている場合は、コードの変更を考慮して新しいバージョン (この例では `_XOPEN_SOURCE=600`) を試してください。
- `__STDC_VERSION__` をテストするユーザーアプリケーションは別の方法で解決する可能性があります

-xlang=c89 の一時的な回避方法

`-std=c89` を明示的に選択しない可能性があります。選択すると C99 および C11 の機能が無効になるためです。

代わりに、`-xlang=c89` を選択した場合は、新しい C コンパイラで以前のバージョンの C コンパイラと同じプログラムを受け入れ、同じ `__STDC_VERSION` を定義できるようになります。

C コンパイラのその他の変更点

C コンパイラの変更点には、45 ページの「全コンパイラに共通の新機能および変更点」に記載されている変更点と、次の追加の変更点が含まれます。

- SPARCACE および SPARCACE+ 向けに、多数の SIMD (`__m128{d|i}`) 組み込み関数が提供されます。

- コンパイラの新しいオプション:
  - `-xcheck=noreturn` は、「do not return」と記述されているルーチンが復帰した場合に実行時エラーを発生させるコードを追加するようコンパイラに通知します。
  - `-xatomic` は、リンクされる不可分サポート実行時ライブラリを指定します。
- 次の C11 機能のサポート
  - 不可分オブジェクト

    注記 - 不可分機能を使用する際は、実行時サポートに特に注意を払う必要があります。詳細は、『Oracle Developer Studio 12.5: C ユーザーズガイド』の「バンドルされている不可分ライブラリ」を参照してください。

  - 型総称式 (`_Generic`)

詳細は、cc(1) のマニュアルページおよび『Oracle Developer Studio 12.5: C ユーザーズガイド』を参照してください。

♦ ♦ ♦ 第 4 章

4 コード分析ツール

コード分析ツールスイートは、メモリーリークやアクセス違反といった一般的なコーディングエラーを検出することでアプリケーションの信頼性および安定性を確保し、開発者が優れたコードを少ないエラーで迅速に記述できるようにします。

この章では、Oracle Developer Studio のこのリリースにおけるコード分析ツールの新機能や変更点について説明します。次のセクションで構成されています。

コード分析ツールについて

コード分析ツールを使用すると、静的分析、動的分析、およびコードカバレージ分析を使用して、メモリーリークやメモリーアクセス違反など一般的なコーディングエラーの多くを検出することにより、アプリケーションの信頼性を高めることができます。Previse ではコンパイル時に静的分析を実行し、discover ではアプリケーション実行時に動的分析を実行することにより、コード品質の問題を特定します。uncover ツールは、コードカバレージデータを分析して、テストスイートの対象にならない関数に関する情報と、それらの関数を対象とすることで得られる利点について通知します。

コードアナライザグラフィックツールまたは codean コマンド行ユーティリティーを使用すると、3 種類の分析を表示してアプリケーションの脆弱性について包括的な見地を得ることにより、妥当性および信頼性を向上させることができます。

codean の新機能

次の機能がコードアナライザのコマンド行ツール codean に追加されました。

- **新しいラベル機能**
  - Previse、discover、および uncover から報告される問題に、false_positive、wont_be_fixed、または verified のラベルを付けられるようになりました。ラベル付けは、コード分析ツールで検出された問題を管理する際に役立ちます。
  - ラベルを使用して問題を抑制できます。
- **新しいテストスイート**
  - -a オプションで保存される discover からのレポートを codean によって蓄積することで、アプリケーションをテストスイートにかけたときに検出された問題を反映できます
  - サマリーページには、テストスイートの対象にならない関数の割合が表示されます

コードアナライザ全般についての詳細は、コードアナライザ内のヘルプ、『Oracle Developer Studio 12.5: コードアナライザユーザーズガイド』、『Oracle Developer Studio 12.5: コードアナライザチュートリアル』、codean(1) のマニュアルページ、および code-analyzer(1) のマニュアルページを参照してください。

discover の新機能

このリリースでは、次の機能が discover メモリー分析ツールに追加されました。詳細は、discover(1) のマニュアルページおよび『Oracle Developer Studio 12.5: Discover および Uncover ユーザーズガイド』を参照してください。

- **Application Data Integrity (ADI) を使用したハードウェア支援検査** — これにより、SPARC M7 プラットフォームでのメモリーアクセス検査の速度が向上します。この機能は Oracle Solaris Studio 12.4 04/15 PSE で導入されました。詳細は、『Oracle Developer Studio 12.5: Discover および Uncover ユーザーズガイド』の「Silicon Secured Memory (SSM) を使用したハードウェアアシスト検査」を参照してください。
- **新しい ADI ヘルパーライブラリ** — この新しいライブラリは、メモリー管理に標準の malloc() および free() 呼び出しを使用しないプログラムに対して ADI 機能を使用するためのものです。『Oracle Developer Studio 12.5: Discover および Uncover ユーザーズガイド』の「カスタムメモリーアロケータおよび discover ADI ライブラリ」に記載されている ADI API を使用して、libadihelpder.so ライブラリとリンクするようにしてください。詳細は、『Oracle Developer Studio 12.5: 概要』の「Oracle Developer Studio コードセキュリティー検査 — ADI の検出」および libadiplugin(3) のマニュアルページを参照してください。
- **UMR および PIR の誤検出の減少** — 誤検出の一般的な原因は、discover ツールが認識していない、Oracle Developer Studio 以外で構築されたライブラリやシステムコールが存在することでした。
- **SIGCHLD シグナル処理の向上** — プログラムで SIGCHLD シグナル処理をインストールした場合は、discover で計測済みのバイナリが失敗しなくなります。
- **メモリー使用量の減少** — discover で計測済みのバイナリが使用するメモリー量が少なくなりました。

- **対話型デバッグの向上** — `dbx` 使用時の `discover` 計測済みバイナリの対話型デバッグが向上しました。
- **アドレス空間レイアウトのランダム化 (ASLR) 処理** — **discover** ユーティリティーでは、アドレス空間レイアウトのランダム化 (ASLR) が有効になっているアプリケーションを処理できます。

uncover の新機能

このリリースでは、次の機能が Uncover コードカバレージツールに追加されました。

- **計測時間とメモリー消費量の改善**

詳細は、`uncover`(1) のマニュアルページおよび『Oracle Developer Studio 12.5: Discover および Uncover ユーザーズガイド』を参照してください。

♦ ♦ ♦ 第 5 章

5 パフォーマンス解析ツール

パフォーマンス解析ツールを組み合わせて使用することで、ユーザーはアプリケーションの動作を解析し、パフォーマンスに影響を及ぼす問題点を見つけることができます。

この章では、Oracle Developer Studio のこのリリースにおけるパフォーマンス解析ツールの新機能や変更点について説明します。

パフォーマンスアナライザについて

パフォーマンスアナライザは、アプリケーションの動作をユーザーが把握できるようにすることで、コード内で問題となっている領域を見つけることができます。パフォーマンスアナライザは、システムリソースをもっとも多く使用している関数、コードセグメント、およびソース行を識別します。パフォーマンスアナライザは、単一スレッド、マルチスレッド、およびマルチプロセスのアプリケーションのプロファイルを作成でき、プロファイリングデータを提供することで、アプリケーションのパフォーマンスを改善できる場所を特定するのに役立てることができます。

パフォーマンスアナライザは一連のコマンドおよびツールで構成されています。具体的には、ユーザーレベルのプログラム上のプロファイリングデータを収集する collect ユーティリティー、Oracle Solaris カーネル上のプロファイリングデータを収集する `er_kernel` ユーティリティー、プロファイリング情報をテキスト形式で表示する `er_print` ユーティリティー、プロファイリング情報をグラフィカルに表示するパフォーマンスアナライザ GUI などがあります。

スレッドアナライザは、マルチスレッドの問題に的を絞ることが可能な関連ツールです。

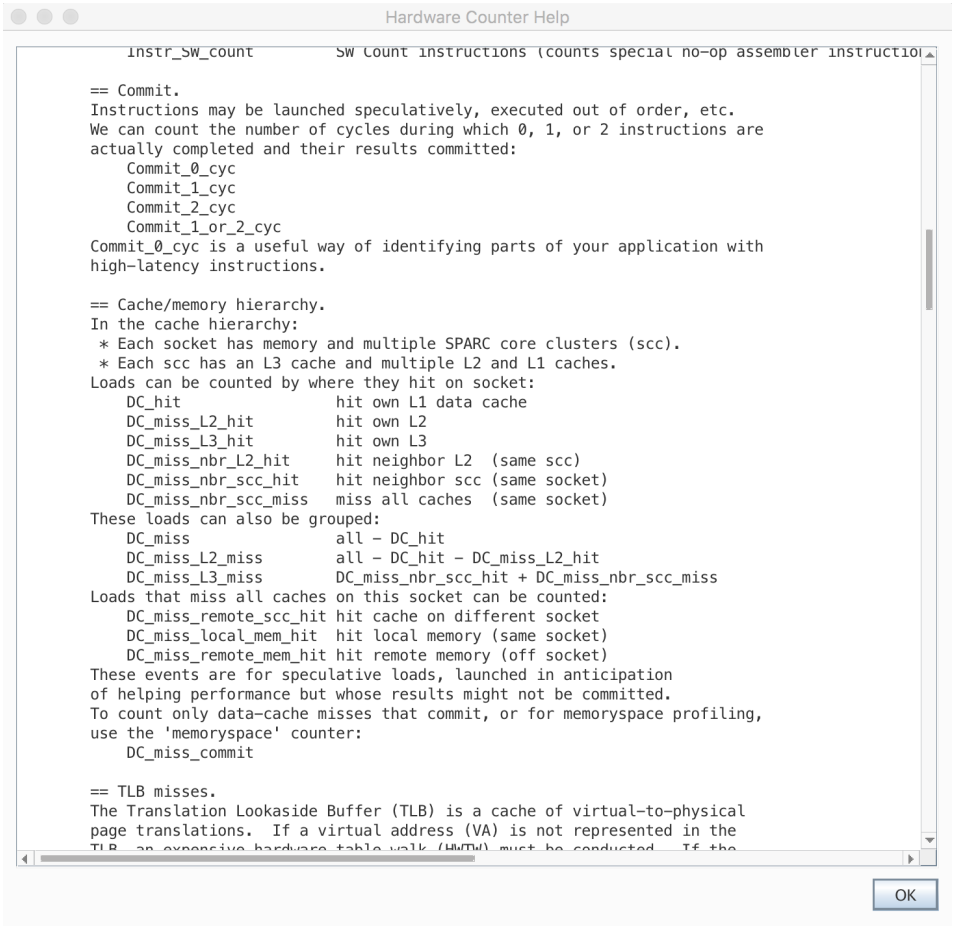
詳細は、『Oracle Developer Studio 12.5: パフォーマンスアナライザ』および『Oracle Developer Studio 12.5: パフォーマンスアナライザチュートリアル』を参照してください。

パフォーマンスアナライザの新機能

このセクションでは、パフォーマンスアナライザおよび関連ツールのこのリリースでの新機能の概要を示します。詳細は、パフォーマンスアナライザのヘルプを参照してください。

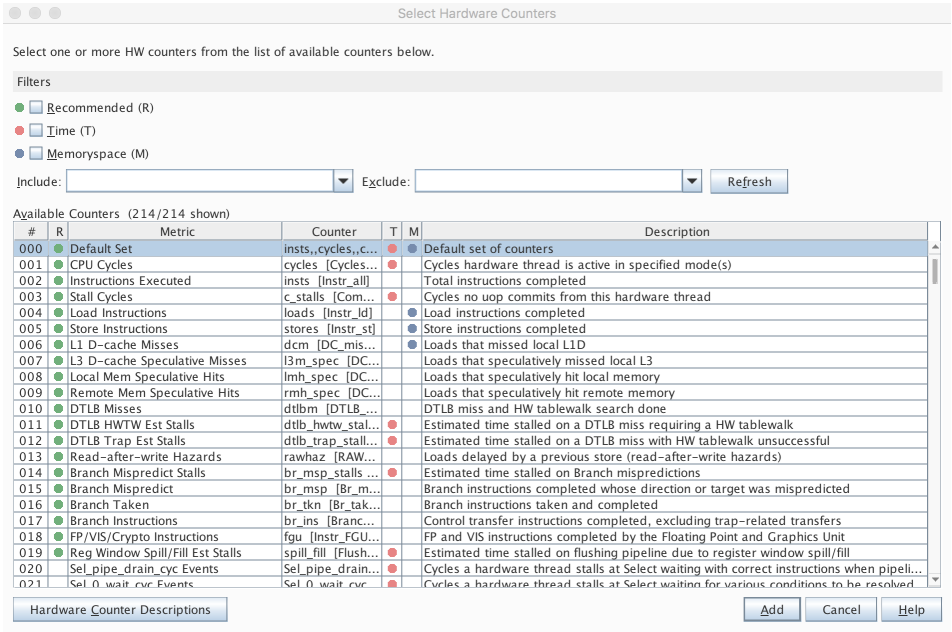
■ **簡略化したハードウェアカウンタプロファイリング**

- プロセッサ固有の新しいヘルプには、主要なハードウェアカウンタの概要が表示されます (SPARC のみ)

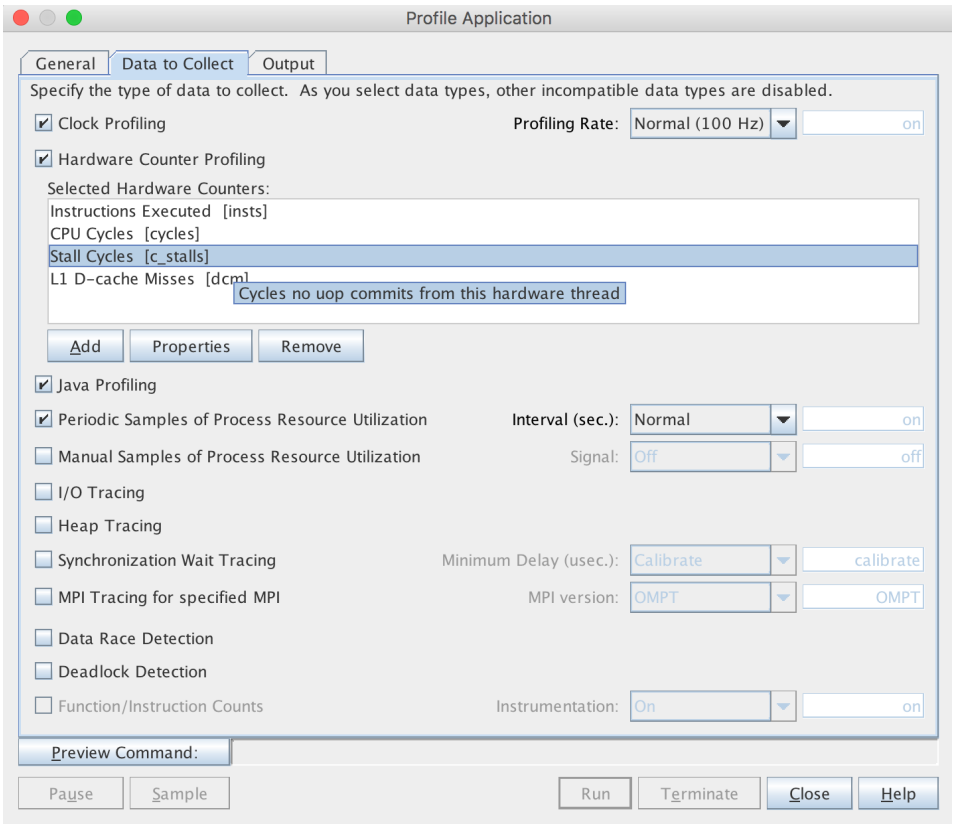



```
    Instr_SW_count          SW Count instructions (counts special no-op assembler instructio

== Commit.
Instructions may be launched speculatively, executed out of order, etc.
We can count the number of cycles during which 0, 1, or 2 instructions are
actually completed and their results committed:
    Commit_0_cyc
    Commit_1_cyc
    Commit_2_cyc
    Commit_1_or_2_cyc
Commit_0_cyc is a useful way of identifying parts of your application with
high-latency instructions.

== Cache/memory hierarchy.
In the cache hierarchy:
 * Each socket has memory and multiple SPARC core clusters (scc).
 * Each scc has an L3 cache and multiple L2 and L1 caches.
Loads can be counted by where they hit on socket:
    DC_hit                  hit own L1 data cache
    DC_miss_L2_hit          hit own L2
    DC_miss_L3_hit          hit own L3
    DC_miss_nbr_L2_hit      hit neighbor L2  (same scc)
    DC_miss_nbr_scc_hit     hit neighbor scc (same socket)
    DC_miss_nbr_scc_miss    miss all caches  (same socket)
These loads can also be grouped:
    DC_miss                 all - DC_hit
    DC_miss_L2_miss         all - DC_hit - DC_miss_L2_hit
    DC_miss_L3_miss         DC_miss_nbr_scc_hit + DC_miss_nbr_scc_miss
Loads that miss all caches on this socket can be counted:
    DC_miss_remote_scc_hit hit cache on different socket
    DC_miss_local_mem_hit  hit local memory (same socket)
    DC_miss_remote_mem_hit hit remote memory (off socket)
These events are for speculative loads, launched in anticipation
of helping performance but whose results might not be committed.
To count only data-cache misses that commit, or for memoryspace profiling,
use the 'memoryspace' counter:
    DC_miss_commit

== TLB misses.
The Translation Lookaside Buffer (TLB) is a cache of virtual-to-physical
page translations.  If a virtual address (VA) is not represented in the
TLB, an expensive hardware table walk (HWTW) must be conducted.  If the
```

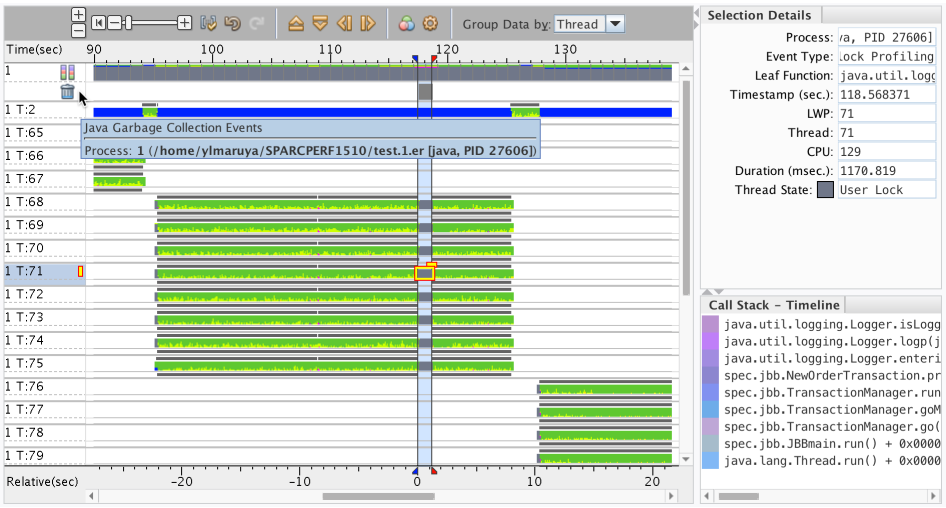
- 新しいダイアログボックス「ハードウェアカウンタの選択」には、カウンタについての詳細な説明が表示されます。さらに、カウンタをフィルタリングしたり、カウンタをより簡単に追加したりできます。



- 適切なプロファイルレートを自動的に選択するための新しい `auto` オプション。
- サポートされているハードウェア用のデフォルトのハードウェアカウンタセットが更新されました。
- サポートされているプラットフォーム用のメモリー領域プロファイリングのビューが更新されました。(x86 システムでは、メモリー領域プロファイリングには Oracle Solaris 11.3 以上が必要です)
- ハードウェアカウンタを選択するためのワークフローが簡略化されました



- **Java プロファイリングの機能強化**
  - Java ガベージコレクタのイベントに関する情報が、概要およびタイムラインに表示されるようになりました。

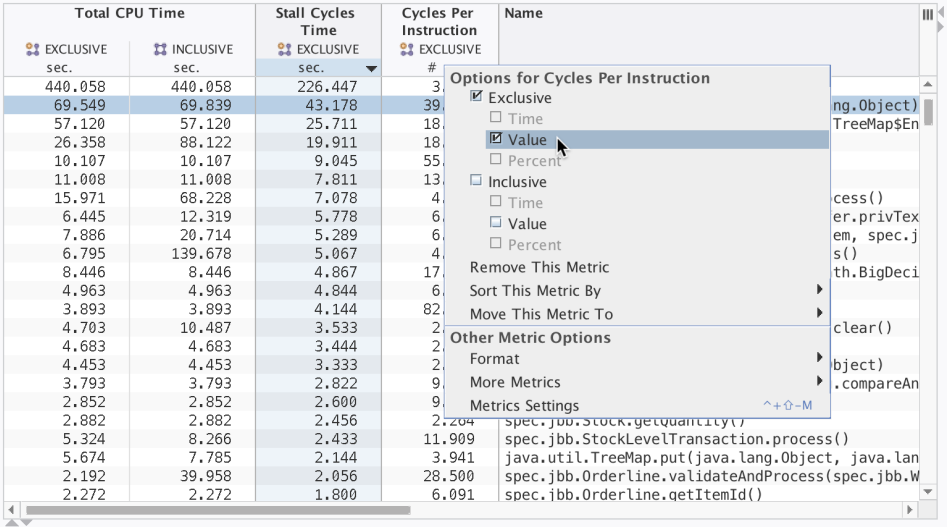


- 関数、ソース、バイトコード、およびマシンコードに対するパフォーマンスメトリックの起因関係が改善されました。

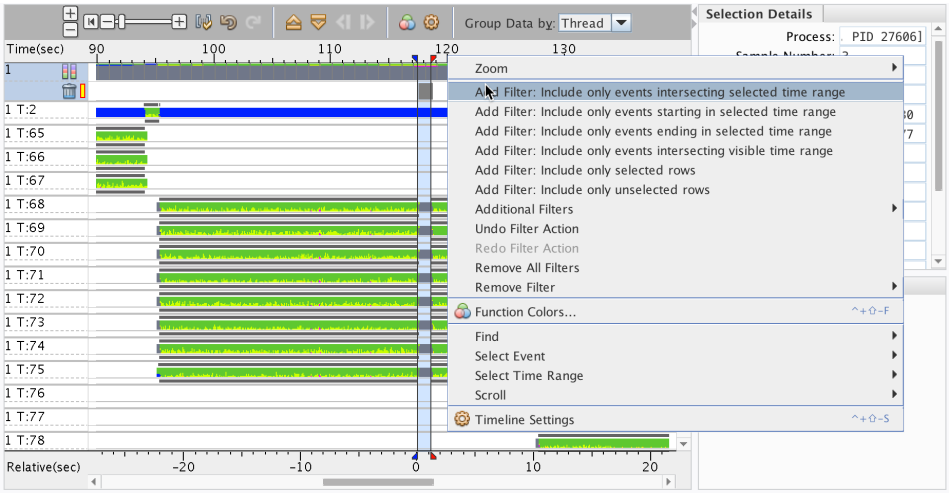
■ **メトリックの表示方法の機能強化**

ほとんどのデータビューで、メトリックの表示方法が向上しており、データの表示方法をより便利に制御できます。

- データ列は、包括的メトリックおよび排他的メトリックをより明確に表示するようにグループ化されます。
- 列ヘッダーの上部隅にコントロールが提供され、それをクリックして、メトリックを構成したり、ビューからまとめて削除したりできます。
- 以前「概要」画面にあったメトリックの表示の時間や割合を選択するためのコントロールは、列ヘッダーコントロールで使用できるようになりました。

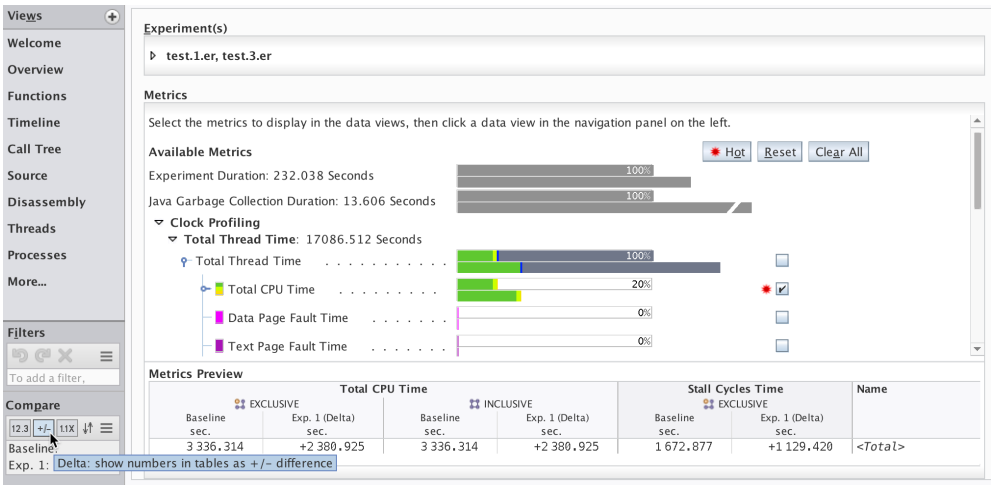


- タイムラインの機能強化
  - 時間範囲をルーラーから選択できるようになりました。
  - 時間または行によるフィルタリングに、ルーラーのコンテキストメニューからアクセスできるようになりました。



- 実験の比較の機能強化

パフォーマンスアナライザのメインウィンドウの新しい「比較」パネルでは、ボタンを使用して、比較対象データの表示モードを絶対、デルタ、および比率の間で切り替えられます。



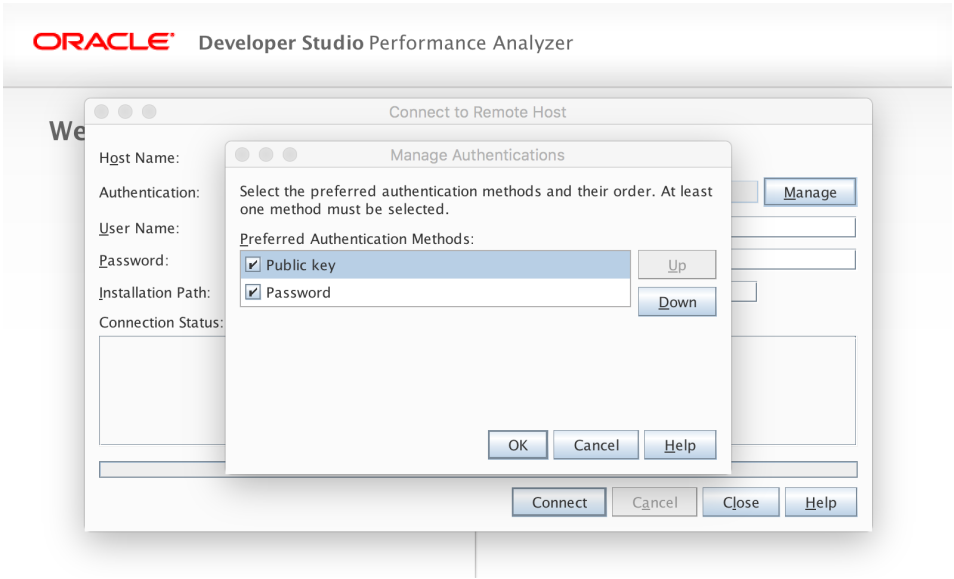
詳細は、「実験の比較」を参照してください。

- **呼び出しツリーの機能強化**

  「呼び出しツリー」ビューには新しい「すべてコピー」オプションがあり、呼び出しツリーをテキスト形式でコピーして、それをテキストファイル内にペーストできます。

- **リモートアナライザの機能強化**

  - リモートアナライザでは、複数の認証方式をサポートするようになりました。



- リモートアナライザに表示されるエラーメッセージが改善され、そのパフォーマンスが向上しました。

コマンド行ツールの変更点

このセクションでは、コマンド行のさまざまなパフォーマンス解析ツールに対して行われた変更について説明します。詳細は、各コマンド行ツールの対応するマニュアルページを参照してください。

データ収集ツールの変更点

データ収集ツールには、`collect` コマンド、`dbx collector` コマンド、および `er_kernel` コマンドがあります。これらの各ツールは、プログラムをプロファイルしてデータを収集し、パフォーマンスアナライザまたは `er_print` によって読み取ることが可能な実験を作成するために使用されます。すべてのデータ収集ツールには次の変更点があります。

- Java 実験では、Java ガベージコレクションのトレースが自動的に行われます。
- Java の場合、synctrace 機能がオプションの `<scope>` 修飾子で拡張されました。`<scope>` は、ネイティブな API トレースを表す `n`、Java API トレースを表す `j`、または両方の API を表す `nj` にできます。デフォルトは `nj` です。

- Oracle SPARC および x86 (Haswell-E/EP を含む) でのハードウェアカウンタのサポートおよびメモリー領域ビューが更新されました。
- ハードウェアカウンタの新しいオプション `-auto` は、適切なプロファイル速度を選択するのに役立ちます。
- SPARC ハードウェアカウンタの使用法に関する新しいガイダンスは、`collect -h` および `er_kernel-h` を使用して入手できます。
- `dlopen()`、`dlmopen()`、`dlclose()`、`exit()`、および `Exit()` に関する問題が修正されました。
- `CLONE_VM` が指定されているときに Linux で作成されたスレッドは追跡されなくなります。
- スレッドの最大数の制限が削除されました。

collect ユーティリティーの変更点

`collect` ユーティリティーは、アプリケーションの実行中にアプリケーションをプロファイルしてデータを収集し、パフォーマンスアナライザまたは `er_print` によって読み取り可能な実験を作成するために使用するツールです。

すべてのデータ収集ツールに共通する変更点に加え、`collect` ユーティリティーはこのリリースでは次のように変更されています。

- Java ターゲットへのヒープトレースが可能になりますが、Java 割り当てではなく、ネイティブな割り当てのみがトレースされます。
- `-R` 引数が認識されなくなります。

dbx collector の変更点

`dbx collector` は、パフォーマンスデータ収集に使用できる `dbx` デバッガのサブコマンドです。詳細は、`collector(1)` のマニュアルページを参照してください。

すべてのデータ収集ツールに共通する変更点に加え、`dbx collector` コマンドはこのリリースでは次のように変更されています。

- 12.4 バージョンのいくつかのバグが修正されました。

er_kernel ユーティリティーの変更点

`er_kernel` コマンドは Oracle Solaris カーネルをプロファイルし、パフォーマンスアナライザまたは `er_print` で調査できる実験を生成します。

すべてのデータ収集ツールに共通する変更点に加え、`er_kernel` ユーティリティーは次のように変更されています。

- プロセスの作成と終了がより正確に追跡されます。

- er_kernel -h 出力のフォーマットが向上しました。

詳細は、er_kernel(1) のマニュアルページを参照してください。

er_print ユーティリティーの変更点

er_print ユーティリティーは、パフォーマンスアナライザで提供されるデータビューのプレーンテキストバージョンを生成します。その出力は、標準出力に表示されます。

er_print ユーティリティーはこのリリースでは次のように変更されています。

er_print ユーティリティーは次のように変更されています。

- 12.4 バージョンのいくつかのバグが修正されました。
- Oracle Developer Studio でのコンパイル時に使用されたフラグがソースおよび逆アセンブリレポートに表示されます。
- マシンモデル情報が実験のヘッダーで報告されます。
- overview コマンドが実装されました。

詳細は、er_print(1) のマニュアルページを参照してください。

その他のコマンドと API の変更点

libcollector API には次の更新があります。

- libcollectorAPI.a および libfcollector.a の静的なバージョンが使用できるようになりました。
- マニュアルページが書き換えられ、Java API に関する記述がよりわかりやすくなりました。libcollector(3) を参照してください。

♦ ♦ ♦ 第 6 章

6 デバッグツール

Oracle Developer Studio には、コマンド行の dbx デバッガ、および dbx を使用するための dbxtool グラフィカルツールが用意されています。また、デバッガは IDE に統合されています。IDE を使用したデバッグの詳細は、第7章「Oracle Developer Studio IDE」を参照してください。

この章には、デバッグツールの新機能に関する次のトピックが含まれます。

dbx デバッガについて

dbx デバッガは、対話型でソースレベルの事後およびリアルタイムのデバッグツールです。コマンド行、dbxtool グラフィカルインタフェース、および Oracle Developer Studio IDE で使用できます。dbx デバッガは、スクリプティング可能で、マルチスレッドに対応しています。

dbx の新機能と変更された機能

次の機能が dbx で追加または変更されました。詳細は、『Oracle Developer Studio 12.5: dbx コマンドによるデバッグ』、dbx(1) のマニュアルページ、および dbx のヘルプファイルを参照してください。dbx のヘルプファイルにアクセスするには、次のように入力します。

```
% dbx
(dbx) help
```

- dbx のバージョンが 8.0 から 8.1 に更新。
- 圧縮されたデバッグ情報のサポート。『Oracle Developer Studio 12.5: dbx コマンドによるデバッグ』の「圧縮デバッグセクション (Oracle Solaris のみ)」を参照してください。
- whatis コマンドの新しいオプション -a。-a オプションはデータメンバーのみを出力します。
- 新しい dbx 環境変数 output_data_member_only。on に設定されている場合は、データメンバーのみを出力します。

- Position Independent Executables (PIE) を処理するためのサポートが追加。
- Oracle Solaris 11 の SPARC での Secured Silicon Memory (SSM) のサポートが追加。この機能を使用するには、`dbx adi` コマンドを使用します。
- C++11 機能であるユーザー定義リテラルのサポート。
- C++14 機能であるバイナリリテラル、単一引用符の桁区切り文字のサポート。
- C11 機能である型総称式のサポート。
- Intel プラットフォーム全体の一貫性を向上させるために、次のレジスタ名を変更。

以前の名前	新しい名前
`fs_base`	`fsbase`
`gs_base`	`gsbase`
`fcwd`	`fcw`
`fswd`	`fsw`
`ftw`	`fctw`
`mxcr_mask`	削除済み
`uesp`	削除済み

詳細は、`dbx` で `help changes` コマンドを発行して、`dbx` のヘルプファイルにアクセスしてください。

♦ ♦ ♦ 第 7 章

7 Oracle Developer Studio IDE

Oracle Developer Studio IDE は、グラフィカルプログラミング環境を好むユーザー向けに Oracle Developer Studio の多くのコンポーネントを統合します。

この章は次の各節から構成されています。

Oracle Developer Studio IDE について

Oracle Developer Studio では、NetBeans プラットフォーム上に構築されたグラフィカル統合開発環境 (IDE) を提供します。この IDE は、Oracle Developer Studio C、C++、および Fortran コンパイラ、`dmake` 分散 make コマンド、および `dbx` デバッガを使用するように構成されます。また、IDE は分析スイートの一部のアナライザツールを統合するため、IDE から離れることなくコードを分析することができます。

IDE を起動するコマンドは `devstudio` です。このコマンドの詳細は、`devstudio`(1) のマニュアルページを参照してください。

IDE の完全なドキュメントは、IDE のヘルプを参照してください。IDE の基本機能を使用する手順については、『Oracle Developer Studio 12.5: IDE クイックスタートチュートリアル』を参照してください。

IDE の新機能および変更された機能

Oracle Developer Studio IDE で追加または変更された機能は次のとおりです。

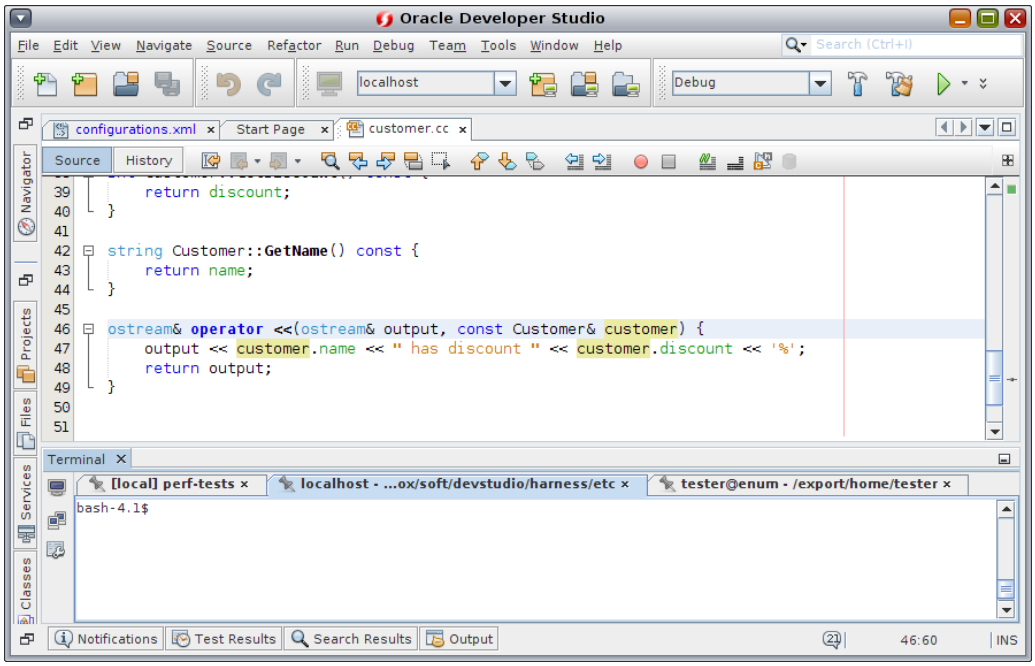
- **新しい固定可能ターミナル** - 詳細は、36 ページの「固定可能ターミナル」を参照してください。

- **既存のソースを使用する新規プロジェクトウィザードで事前ビルドおよびビルド手順がより明確かつ理解しやすくなりました。** - 詳細は、37 ページの「新規プロジェクトウィザード」を参照してください。
- **リモート開発が改善されました。** - 詳細は、38 ページの「リモート開発の改善」を参照してください。
- **混在開発 (Java および C/C++) のサポート。** - 詳細は、39 ページの「混在開発 (Java および C/C++) のサポート」を参照してください。
- **複数ファイルのプロパティーの編集** - 詳細は、40 ページの「複数ファイルのプロパティーの編集」を参照してください。
- **Doxygen C++ コメントのサポート** - 詳細は、41 ページの「Doxygen C++ コメントのサポート」を参照してください。
- **複合文のコード折りたたみ** - 詳細は、41 ページの「複合文のコード折りたたみ」を参照してください。
- **エディタ内のコンパイラのヒントから出力ログへのナビゲート** - 詳細は、コンパイラのヒントからビルドログへのナビゲートに関する IDE のヘルプトピックを参照してください。
- **SendTo ユーティリティー** - 詳細は、SendTo ユーティリティーに関する IDE のヘルプトピックを参照してください。
- **不足している switch 節の生成**
- **コールグラフの機能拡張** - 詳細は、コールグラフの使用に関する IDE のヘルプトピックを参照してください。
- **新しい監査およびヒント - セキュリティーコーディングの推奨を含む** - IDE で「オプション」->「エディタ」->「ヒント」の順に選択し、C/C++ 言語を選択して、利用可能なさまざまな新しいヒントを参照してください。詳細は、IDE のヘルプトピック「静的コードアナライザの使用」を参照してください。
- **新しいリファクタリング** - 詳細は、リファクタリングによる関数と変数の導入に関する IDE のヘルプトピックを参照してください。

固定可能ターミナル

「タブの固定」アクションが「ターミナル」ポップアップメニューに追加されました。

図 2　　固定可能ターミナル

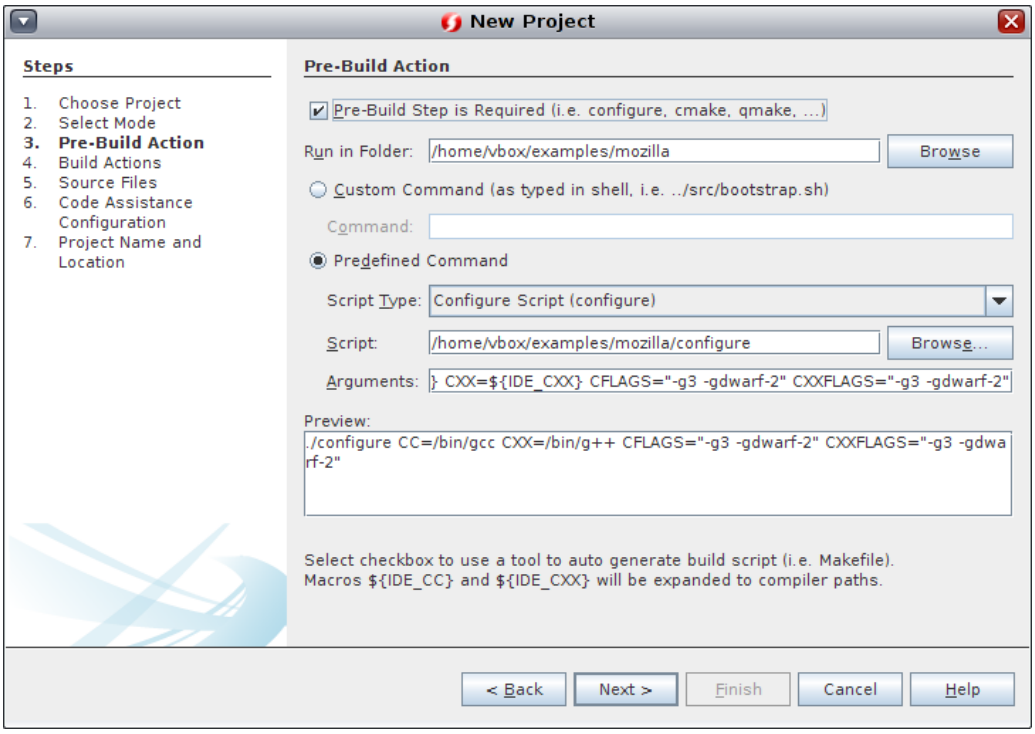


詳細は、ターミナルの固定に関する IDE のヘルプトピックを参照してください。

新規プロジェクトウィザード

既存のソースを使用する新規プロジェクトウィザードでは、あとで新しい「事前ビルド」プロパティータブを使用して事前ビルド手順をカスタマイズできます。このタブには、「プロジェクト・プロパティ」->「ビルド」->「事前ビルド」でアクセスできます。次の図は、新規プロジェクトウィザードの「事前ビルド・アクション」タブを示しています。

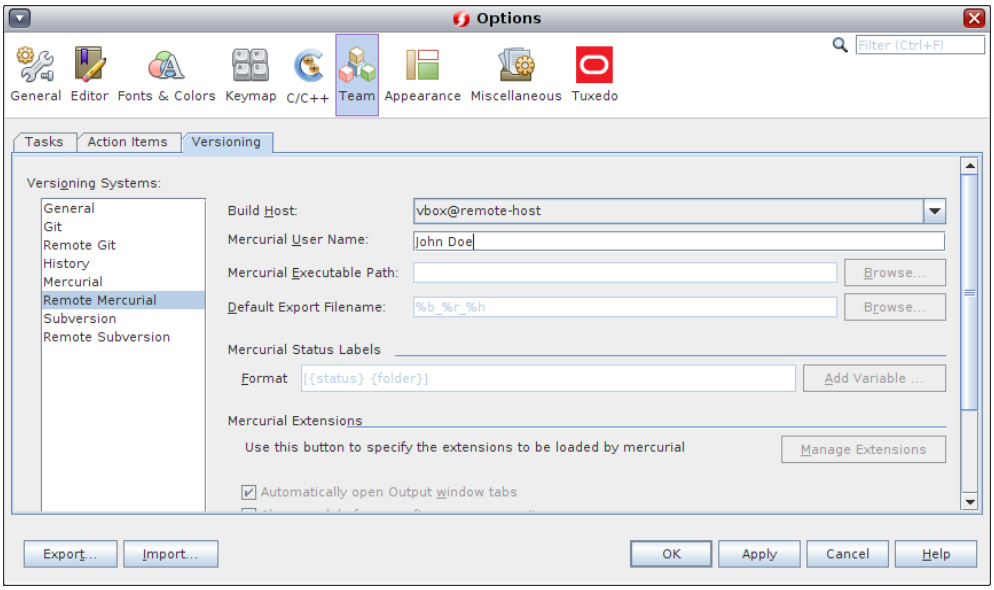
図 3 新規プロジェクトウィザード



リモート開発の改善

次の機能がリモート開発に追加されました。

- リモートモードでの SVN、Git、および Mercurial のサポート - この機能により、VCS をローカルユーザーとして完全リモートモードで使用できます。リモート VCS プリファレンスのカスタマイズは、「ツール」->「オプション」->「チーム」->「リモートMercurial/Git/Subversion」および「ツール」->「オプション」->「フォントと色」->「リモートMercurial/GIT/Subversion」で実行できます。次のスクリーンショットは、「オプション」ウィンドウのバージョン管理オプションを示しています。



- 完全リモートプロジェクトでのコード支援のサポート
- 新規リモートホストの設定ウィザード用の新しいオプション

詳細は、バージョン管理システムでの完全リモートモードの使用および C/C++/Fortran 開発用のリモートホストの構成に関する IDE のヘルプトピックを参照してください。

混在開発 (Java および C/C++) のサポート

IDE では、C/C++ プロジェクトを、Java Native Interface (JNI) および Java Native Access (JNA) テクノロジを使用する Java プロジェクトと統合できます。これらは、Java プログラムからネイティブ C および C++ プログラムを呼び出すことを可能にするテクノロジです。

JNI メソッドを使用する Java プロジェクトがある場合は、Java プロジェクトから C++ JNI プロジェクトを生成できます。JNI プロジェクトでは、Java コード内のすべてのネイティブインタフェースが実装されます。C++ JNI ライブラリプロジェクトの作成方法および関連する Java プロジェクトやネイティブプロジェクトでサポートされている IDE 機能については、混在開発プロジェクトでの JNI の操作および混在開発での JNA の操作に関連する IDE のヘルプトピックを参照してください。

混在開発を使用するために、次の必須の手順を実行する必要があります。

1. **Java JDK があり、Java JDK がシステムにインストールされている状態で IDE を実行してこの機能を使用しているか確認します。**

IDE では通常、Java JRE のみを必要とします。

2. **IDE アップデートセンターから Java SE プラグインをインストールします。**

   インストール対象のモジュールは 5 つあります。

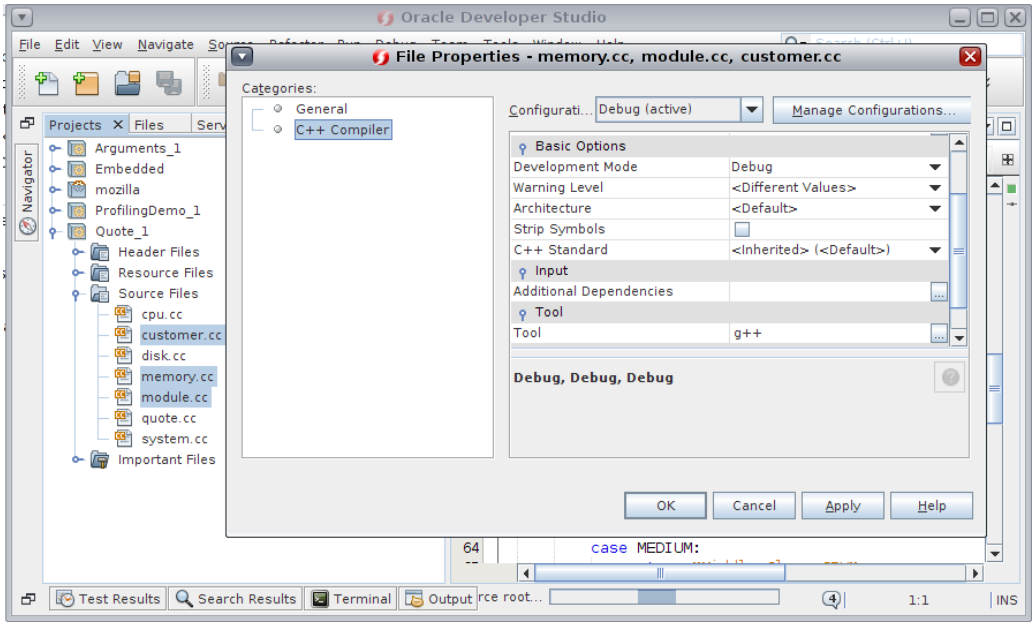
3. **ユーザーディレクトリに注意し、次のことを認識しておきます。**

   - その他のすべての必須モジュールが、指定されたユーザーディレクトリにインストールされます。
   - 混在開発は、追加のモジュールが関連付けられているユーザーディレクトリを使用して IDE を実行した場合にのみ機能します。
   - 正しいユーザーディレクトリにインストールしていない場合は、この機能が正常に動作しなかったり、新しいユーザーディレクトリを使用した Java SE の再インストールが必要になったりします。

複数ファイルのプロパティーの編集

プロジェクト内の複数のファイルのプロパティーを編集できるようになりました。「プロジェクト」ウィンドウで Shift キーを押しながら複数のファイルをクリックして選択し、右クリックしながら「プロパティ」を選択します。「プロパティ」ウィンドウが表示されます。

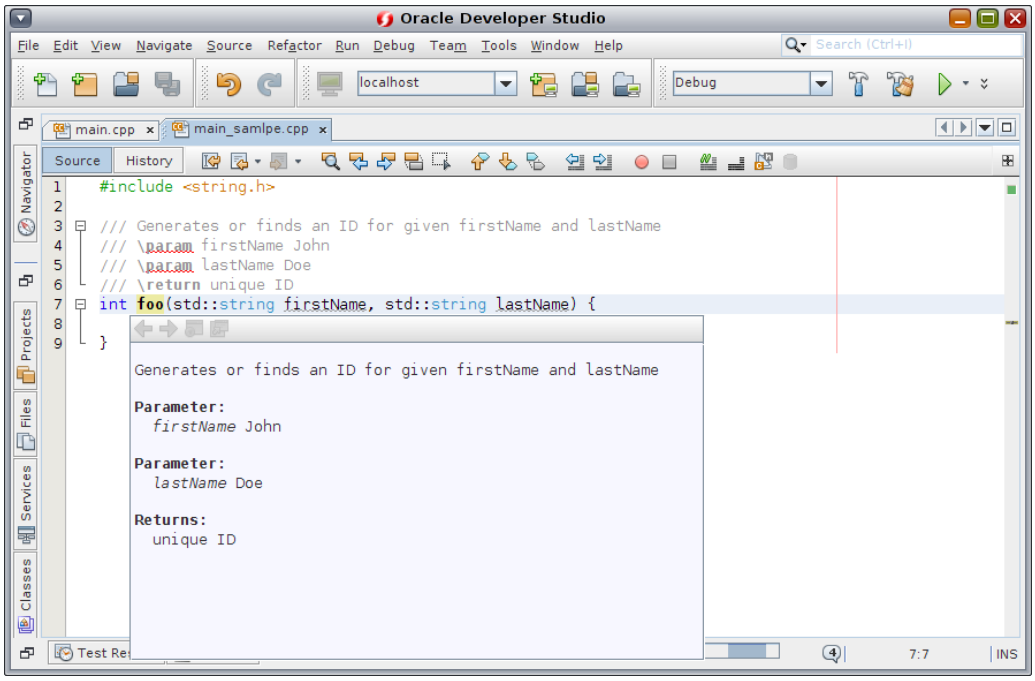
図 4 複数ファイルのプロパティーの編集



Doxygen C++ コメントのサポート

プロジェクトでは、doxygen 形式の 1 行の「///」コメントをドキュメントに使用できるようになりました。

図 5 Doxygen C++ コメントの使用

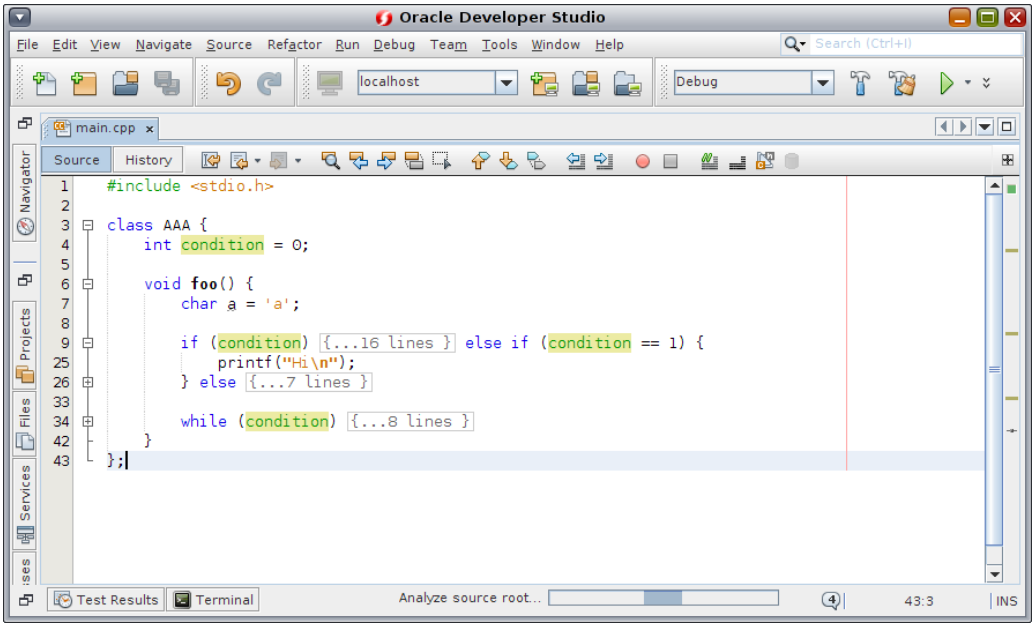


詳細は、コードへのドキュメントの追加に関する IDE のヘルプトピックを参照してください。

複合文のコード折りたたみ

if-else や do-while などの複合文を折りたためるようになりました。

図 6 IDE でのコード折りたたみ



詳細は、C および C++ ファイル内のコードのブロックの折りたたみに関する IDE のヘルプトピックを参照してください。

♦ ♦ ♦ 第 8 章

8 OpenMP API

この章では、Oracle Developer Studio のこのリリースにおける OpenMP API サポートの変更点について説明します。

OpenMP

このセクションでは、OpenMP API の新機能および更新について説明します。

- OpenMP および `autopar` プログラム用のスレッド数のデフォルト値は、`cores_per_chip` の倍数です。スレッドの数を計算するためのアルゴリズムは `cores_per_chip` から始まり、その数値の後続の倍数をチェックし、32 以下で、マシン上のコア数を超えない最大の倍数を選択します。

OpenMP の詳細は、『Oracle Developer Studio 12.5: OpenMP API ユーザーズガイド』を参照してください。

♦ ♦ ♦ 第 9 章

9 その他の変更点

この章では、Oracle Developer Studio ソフトウェアのその他のコンポーネントについて、新機能および変更された機能について説明します。

コンパイラに対する変更

次のセクションではコンパイラに対して行われた変更について説明し、内容は次のとおりです。

全コンパイラに共通の新機能および変更点

C、C++、および Fortran コンパイラに対して前のリリースから次の変更が行われました。詳細は、コンパイラのマニュアルページを参照してください。C++ コンパイラに固有の変更点は、第2章「C++ コンパイラ」に詳しく説明されています。C コンパイラに固有の変更点は、第3章「C コンパイラ」に詳しく説明されています。

新しいハードウェア上でのアプリケーションパフォーマンス

Oracle Developer Studio のすべてのリリースには、Oracle Sun ハードウェアサーバー用のパフォーマンス向上が含まれています。このリリースには、SPARC M6、SPARC M7、SPARC T7、SPARC S7、および Intel Broadwell/avx2_i プロセッサの拡張サポートが含まれています。

コンパイラのその他の変更点

次に、C、C++、および Fortran コンパイラに影響を及ぼす、その他の変更点の一覧を示します。

- SPARC M6、M7、T7、および S7 プロセッサのサポート。
- x86 データ領域プロファイリングのサポート。
- `-xcheck` には `-xcheck=stkovfl` という新しいデフォルトがあります。
- コンパイラの新しいオプション:
  - `-features=[no]mergestrings` を指定すると、コンパイラは文字列リテラルやその他の適切な定数または読み取り専用データをバイナリの特殊セクションに入れ、そこでリンカーによって重複した文字列が削除されます。このオプションは SPARC でのみ使用できます。
  - `-xsecure_code_analysis` を指定すると、コンパイル時に、可能性のあるメモリーのセキュリティー違反を検出して表示する、コンパイラセキュアコード分析が有効になります。
  - `-xtarget=S7`、`-xchip=S7` は、コンパイラドライバでサポートされています。

Fortran コンパイラ

Fortran コンパイラは、Fortran77、Fortran90、および Fortran95 規格のための、過去最高のランタイムパフォーマンスおよび互換性オプションにより、科学技術アプリケーション開発をサポートします。Fortran 2003 の機能と OpenMP 4.0 のサポートの大部分が含まれます。Fortran コンパイラでは C および C++ コンパイラと同様のハイパフォーマンスなコード生成テクノロジが使用され、最新の SPARC および x86 ベースの Oracle システムのための最大パフォーマンスの並列コードがアプリケーションで生成されることを保証します。

Fortran コンパイラの変更点には、45 ページの「全コンパイラに共通の新機能および変更点」に記載されている変更点と、次の変更点が含まれます。

- 自由プログラム形式の行の最大長が 132 文字から 250 文字に増えました。

詳細は、`f95(1)` のマニュアルページおよび『Oracle Developer Studio 12.5: Fortran ユーザーズガイド』を参照してください。

新しい静的解析機能

C および C++ コンパイラではデフォルトで、コンパイル時のメモリーの安全性検査に関する警告が生成されます。

次のメッセージタグが含まれています。

- SEC_UNINITIALIZED_MEM_READ
- SEC_UNINITIALIZED_BITOP
- SEC_UNDEFINED_RETURN_VALUE
- SEC_ARR_OUTSIDE_BOUND_READ
- SEC_ARR_OUTSIDE_BOUND_WRITE
- SEC_FREED_PTR_RETURN
- SEC_READ_FREED_PTR
- SEC_WRITE_FREED_PTR
- SEC_NULL_PTR_DEREF

これらの警告は、コンパイル時のほかのすべてのエラーや警告とともに stderr に書き出されます。それらは、コンパイル時のほかのすべてのメッセージと同様に、-erroff、-errtags、および -errwarn コマンド行オプションや error_messages() プラグマによって制御されます。

-errwarn コマンド行オプションを使用する方法

-errwarn コマンド行オプションを使用すると、メモリーの安全性検査の警告を致命的エラーに変えることができます。例:

- -errwarn=SEC_NULL_PTR_DEREF を使用すると、すべての NULL ポインタ間接参照が致命的エラーになります。
- -errwarn=%all を使用すると、コンパイル時のすべての警告が致命的エラーになります。

静的エラー検査を無効にする方法

静的エラー検査は、コンパイラのバックエンド処理と並列して実行されます。高い負荷のかかっているシステムや、きわめて複雑なフロー制御を含む非常に大きなモジュールなどの特定の状況では、コンパイル時間を増やすことができます。コンパイル時間がきわめて重要なシナリオでは、コマンド行オプション -xsecure_code_analysis=no によって可能性のある重大な診断メッセージを抑制して、静的エラー検査を無効にできます。

あるいは、Oracle Developer Studio のデフォルトの config ファイル機能を使用して、サイト全体の静的エラー検査を無効にすることもできます。たとえば、構成ファイル cc.defaults または CC.defaults を使用できます。さらに、c89.defaults または c99.defaults も使用できます。インストールパスは *install-dir*/lib/compilers/etc/config. です。

SPRO_DEFAULTS_PATH 環境変数を使用すると、メイクファイルを変更しなくてもデフォルトの静的エラー検査を無効にできます。SPRO_DEFAULTS_PATH=*path* の path は、デフォルトのファイルが含まれているディレクトリです。

error_messages() プラグマを使用した SEC メッセージの有効化と無効化

error_messages() プラグマを使用すると、指定されたソースファイルの領域に従って SEC メッセージを選択的に有効および無効にできます。

次の例は、ある特定の文で SEC の特定の警告を無効にする方法を示しています。

```
cat foo.c:
     <some code>
#pragma error_messages (off, tag-of-interest)
   <statement of interest>
#pragam error_messages (default|on, tag-of-interest)
   <remainder of code>
```

ここで、*tag-of-interest* は SEC のフロントエンドで生成される警告の 1 つです。

コードの領域内のすべての SEC 警告を無効にするには:

```
    cat foo.c:
     <some code>
#pragma error_messages (off, SEC_UNINITIALIZED_MEM_READ,SEC_UNINITIALIZED_BITOP,
SEC_UNDEFINED_RETURN_VALUE,SEC_ARR_OUTSIDE_BOUND_READ,
SEC_ARR_OUTSIDE_BOUND_WRITE,SEC_DOUBLE_FREE,SEC_FREED_PTR_RETURN,SEC_READ_FREED_PTR,
SEC_WRITE_FREED_PTR,SEC_NULL_PTR_DEREF
)
   line-or-lines-of-interest
#pragam error_messages (default|on, SEC_UNINITIALIZED_MEM_READ,SEC_UNINITIALIZED_BITOP,
SEC_UNDEFINED_RETURN_VALUE,
SEC_ARR_OUTSIDE_BOUND_READ,SEC_ARR_OUTSIDE_BOUND_WRITE,SEC_DOUBLE_FREE,SEC_FREED_PTR_RETURN,
SEC_READ_FREED_PTR,SEC_WRITE_FREED_PTR,
SEC_NULL_PTR_DEREF
)
   <remainder of code>
```

パフォーマンスライブラリの変更点

Oracle Developer Studio パフォーマンスライブラリは、線形代数や大量の数値計算を伴う問題を解くために最適化された高速な数学サブルーチンのセットです。Oracle Developer Studio パフォーマンスライブラリは、http://www.netlib.org の Netlib から入手できるパブリックドメインサブルーチンのコレクションに基づいています。Oracle はこれらのパブリックドメインサブルーチンを強化し、Oracle Developer Studio パフォーマンスライブラリとしてバンドルしました。

このリリースでは次の変更が行われました。

- Oracle Developer Studio パフォーマンスライブラリの LAPACK がバージョン 3.5.0 にアップグレードされました。次の機能を含む LAPACK 3.5.0 のすべての新機能が Oracle Developer Studio パフォーマンスライブラリに実装されています。
  - Fortran 95 インタフェース用のオプションの引数の削除。
  - LAPACK 3.5.0 に適合するように更新された LAPACK の基本ルーチン
- `STACKSIZE` 環境変数の必要な最小値が 1 スレッドあたり 8M バイトに新たに設定されました。

  Oracle Developer Studio パフォーマンスライブラリは、大きなデータをハードウェアのキャッシュ内に収まるブロックに分割し、それらのブロックをスタック内にコピーすることでデータの局所性を高めます。高速なハードウェアほどキャッシュが大きくなるため、ブロックやスタックサイズも大きくなります。現在のハードウェアをうまく適応させるには、スレッドごとの必要な最小スタックサイズをすべてのプラットフォームで 8M バイトにします。

詳細は、『Oracle Developer Studio 12.5: パフォーマンスライブラリユーザーズガイド』を参照してください。

数学ライブラリの変更点

Oracle Solaris Studio 12.4 以前のリリースのコンパイラでは、`<sunmath.h>` ヘッダーファイルに「ラッパー」関数の C/C++ プロトタイプが含まれていました。これらは、libsunmath 内のさまざまな非標準の数学関数や浮動小数点ユーティリティールーチンを呼び出すために Fortran から呼び出されていたものです。これらの「ラッパー」関数自体は、C/C++ から直接呼び出すことができる対応する関数を呼び出すだけなので、C/C++ プログラムでそれらを使用する必要はありません。たとえば、「ラッパー」関数 `r_atan2d_` は、C 関数 `atan2df` と同等です。

Oracle Developer Studio 12.5 では、これらの「ラッパー」関数は非推奨です。`<sunmath.h>` 内のそれらのプロトタイプは、プリプロセッサマクロ `__SUNMATH_DEPRECATED` によって監視されるようになり、C プログラムでこれらの関数を使用すると、警告が生成されます。

```
example% cat func.c
    #include <sunmath.h>
    float func(float x, float y) {
        return r_atan2d_(&x, &y);
    }
    example% cc -c func.c
    "func.c", line 4: warning: implicit function declaration:
    r_atan2d_
    example%
```

結果となるプログラムによって間違った結果が生成される可能性があります。このようなプログラムは、対応する C 関数を直接使用して書き直すようにしてください。

```
return atan2df(x, y);
```

あるいは、<sunmath.h> の組み込みに #define __SUNMATH_DEPRECATED prior を追加したり、コマンド行で指定されたコンパイラフラグに -D__SUNMATH_DEPRECATED を追加したりできますが、「ラッパー」関数は今後のリリースですべて削除される可能性があることに注意してください。

索引