What's New in the Oracle® Developer Studio 12.6 Release

Part No: E77784
July 2017

What's New in the Oracle Developer Studio 12.6 Release

Part No: E77784

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Référence: E77784

Accès aux services de support Oracle

Les clients Oracle qui ont souscrit un contrat de support ont accès au support électronique via My Oracle Support. Pour plus d'informations, visitez le site `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info` ou le site `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs` si vous êtes malentendant.

Contents

Using This Documentation

- **Overview** – Describes the new and changed features in the compilers and tools with this Oracle Developer Studio 12.6 release.
- **Audience** – Application developers, system developers, architects, support engineers
- **Required knowledge** – Programming experience, software development testing, aptitude to build and compile software products

Product Documentation Library

Documentation and resources for this product and related products are available at `http://www.oracle.com/pls/topic/lookup?ctx=E77782-01`

Feedback

Provide feedback about this documentation at `http://www.oracle.com/goto/docfeedback`.

♦ ♦ ♦ CHAPTER 1

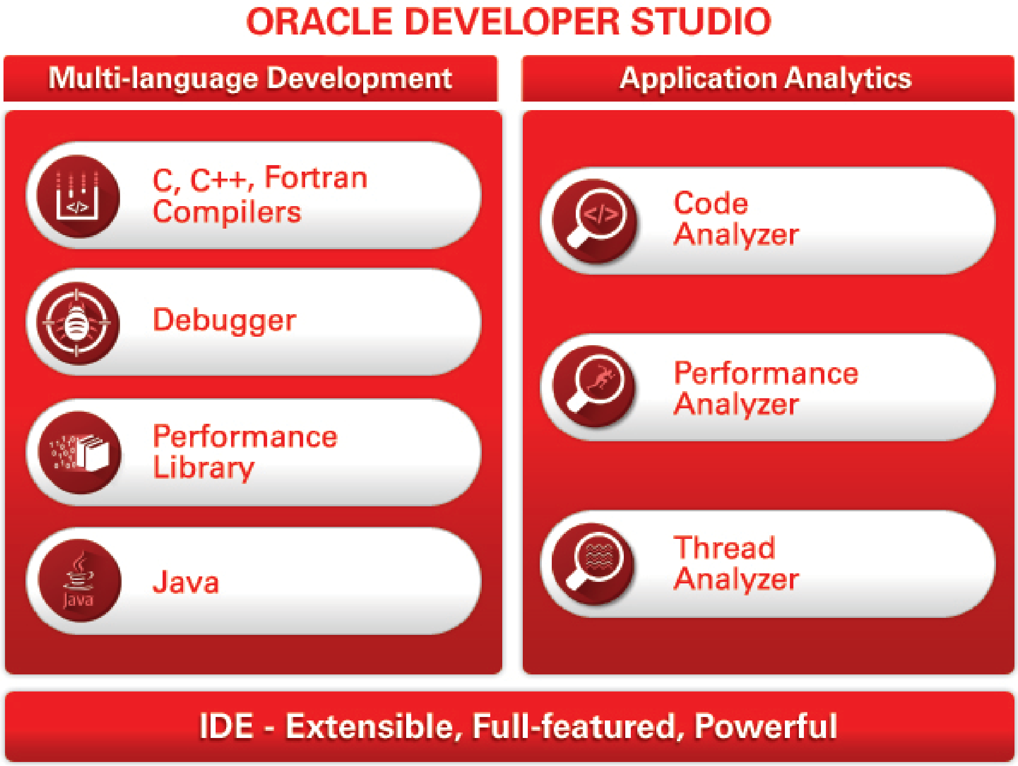
Introducing the Oracle Developer Studio 12.6 Release

This chapter provides an overview of the key updates in this release.

Overview of Oracle Developer Studio

Oracle Developer Studio includes a compiler suite, an analysis suite, and a graphical integrated development environment (IDE) that is tailored for use with the compilers and tools from both the suites. Together, they provide a development environment that is optimized for developing applications with the best performance on Oracle Sun hardware.

FIGURE 1 Diagram shows compiler and analysis suites integrated with the IDE



Key Features in This Release

Oracle Developer Studio 12.6 delivers highly optimized compilers, advanced analysis tools, and a multi-language aware IDE for easy development of fast, reliable, and secure applications for Oracle Solaris and Linux operating systems on-premise, or in the Cloud. Oracle Developer Studio tools are optimized to complement the complete hardware and software stack, and enable development teams to write better code, faster.

A summary of the key features in this release include the following:

- **Fastest Code Generation for Oracle Systems and Oracle Cloud** — Compilers, code generators, and runtime libraries are tuned to take advantage of the newest Oracle SPARC M8, Fujitsu M12, and Intel Xx86 Skylake-based servers. These new systems or processors introduce new instructions, new cache sizes, and new pipelines that the compiler takes into consideration to generate the best application performance for applications running on these new systems. For more information, see the compiler user guides.
  - SPARC M8/T8 cache and instruction scheduling optimizations
  - New pragmas to support misaligned data access on SPARC
  - SPARC M8 and Intel Skylake hardware counter information accessible through the Performance Analyzer
  - Key Performance Library routines tuned to take advantage of SPARC M8 chip capabilities
  - Fujitsu SPARC M12 optimizations
  - Intel Skylake AVX512 support
- **Simplified Development With Open Source Compatibility and Interoperability** — Support for formal language standards by compilers is critical to a developer's ability to move code across various types of systems in most modern-data enterprise data centers. In addition, the open source movement has created a wealth of applications that are being incorporated into enterprise systems. The majority of this open source code is written using the GNU toolset, so compatibility with this standard is also important. Oracle Developer Studio 12.6 supports the following:
  - Full C++ 2011 and C++ 2014 Language Standard compliance. For more information, see Chapter 2, “C++ Compiler”.
  - More than 40 gcc compatibility features, which extends the gcc compatibility features offered in our previous release. For more information, see “New and Changed Features Common to the Compilers” on page 51
  - Java 9 support in IDE and Performance Analyzer
- **Accelerated Cloud Computing With Oracle Developer Studio Cloud Service Plug-In** — Deploying applications to a Cloud is becoming more commonplace as companies move from on-premise data centers to centralized cloud providers. Developers are also moving their development environments to centralized, cloud-provided data centers. This release offers features to make it easier for developers to take advantage of this transition.
  - The Oracle Developer Studio IDE offers a new plug-in that enables you to run the IDE on your local desktop and access the Oracle Developer Studio Cloud Service, which offers issue tracking, source code management, build servers, and an easy way to deploy your application into the Oracle Cloud. For more information, see Chapter 7, “Oracle Developer Studio IDE”.
  - The Oracle Cloud Service uses ssh to access its Infrastructure as a Service Compute VMs. The Oracle Developer Studio remote Performance Analyzer supports ssh tunneling so developers can launch and analyze performance experiment results

    that were run on the Oracle Cloud Compute VMs. For more information on using Performance Analyzer remotely, see "Using Performance Analyzer Remotely" in *Oracle Developer Studio 12.6: Performance Analyzer*

  - As more developers adopt a micro-services model, they are also using container technology to host these micro-services. The Oracle Developer Studio IDE enables you to create Docker containers for later deployment to the Oracle Cloud or other systems that supports such containers. For more information, see "Docker" on page 42.

- **Increased Application Security** — Application security is a key requirement for modern applications. Oracle Developer Studio is making it easier for the developers to create more secure applications by providing the following features:
  - Secure coding assistant in the IDE which highlights code that does not adhere to the Oracle Solaris secure coding recommendations, and it also suggests alternatives that might be more secure. For more information, see "New and Changed IDE Features" on page 29.
  - The `discover` tool can leverage SPARC Silicon Secured Memory (SSM) to provide real-time information about erroneous application memory access and memory corruption, which are typical exploits for security violations. For more information, see "Hardware-Assisted Checking Using Silicon Secured Memory (SSM)" in *Oracle Developer Studio 12.6: Discover and Uncover User's Guide*.
  - Compilers can perform automated code checking with each compile. A side effect of this feature is that you may see additional warnings when building your code with the Oracle Developer Studio 12.6 compilers. There are compiler options to disable these checks.

♦ ♦ ♦ CHAPTER 2

C++ Compiler

This chapter describes the new and updated features in this release of the Oracle Developer Studio C++ compiler.

- "About the C++ Compiler" on page 13
- "C++ Compiler Changes" on page 13

About the C++ Compiler

This section provides a summary list of the new features and changed functionality introduced in the Oracle Developer Studio 12.6 C++ 5.15 Compiler release.

The C++ compiler (`CC`) produces code that is targeted for specific operating systems, processors, architectures, memory models (32-bit and 64-bit), floating-point arithmetic, and more, according to the command-line options you specify. The compiler automatically parallelizes serial source code to produce binaries with better performance on multi-core systems, and can also prepare binaries for enhanced debugging or analysis by other Oracle Developer Studio tools. The compiler also supports GNU C and C++ compatibility features.

C++ Compiler Changes

This section includes the changes that are described in "New and Changed Features Common to the Compilers" on page 51.

The new features in this release specific to the C++ compiler are as follows:

- **New `section` attribute** – The C++ compiler now provides a new `section` attribute that specifies a variable or a function present in a particular section.

- **Inlining Behavior Change for** `-xO3` – Supports automatic inlining of functions whose body is smaller than the calling overhead. In addition, the `-xO3` and `-xO4` result in the minimum code size when used with the `-xspace` option.
- **Change in** `-xlibmopt` **Option** – The `-xlibmopt` option has been enhanced with the `-%none`, `-archive`, and `-shared` sub-options.
- `libCrunG3` **Linked Statically on Linux** – `libCrunG3` is a runtime support library needed by C++ programs that are compiled with any of the options, `-compat=g`, `-std=c++03`, `-std=c++11`, or `-std=c++14`.

  On Oracle Solaris, `libCrunG3` is linked dynamically by default, when needed. It is part of the Oracle Solaris Operating System, and hence it can always be found by user programs. On Linux, you would need to supply `libCrunG3.so.1` with your program if you linked the library dynamically. To avoid this problem, `libCrunG3`, when needed, is linked statically by default on Linux.
- **Improved Error Reporting in Function Overload Resolution** – The previous compilers of Oracle Developer Studio and Oracle Solaris Studio did not provide detailed messages in reporting ambiguous or unresolvable calls to overloaded functions. In Oracle Developer Studio 12.6, the error reporting has been improved. The following examples illustrate the improved error reporting feature.

  ```
  Example 1:

  % cat ex1.cc
  void f(int n, int* ptr);
  void f(int, int, int);
  void f(int);
  void foo()
  {
      f(1, 2);
  }
  ```

  The previous compilers reported a message like:

  ```
  % CC -c ex1.cc
  "ex1.cc", line 7: Error: Could not find a match for f(int, int) needed in
  foo().
  1 Error(s) detected.
  ```

  The Oracle Developer Studio 12.6 C++ compiler provides more information about why no match could be found.

```
% CC -c ex1.cc
"ex1.cc", line 7: Error: Could not find a match for f(int, int) needed in
foo().
"ex1.cc", line 1: Note: Candidate 'f(int, int*)' is not viable: argument
'ptr' can't be converted from 'int' to 'int*'.
"ex1.cc", line 3: Note: Candidate 'f(int)' is not viable: too many arguments.
"ex1.cc", line 2: Note: Candidate 'f(int, int, int)' is not viable: too few
arguments.


Example 2:

% cat ex2.cc
struct A;
struct B {
    B(const A&);
};
struct D1 : B {};
struct D2 : B {};
struct A {
    A();
    operator  D1() const;
    operator  D2&() volatile ;
};
void foo(A& ra)
{
    A va;
    B vb  = ra;
}
```

The previous compilers reported a message like:

```
% CC -c ex2.cc
"ex2.cc", line 19: Error: Overloading ambiguity between "A::operator D1()
const" and "B::B(const A&)".
"ex2.cc", line 19: Error: Cannot use A to initialize B.
2 Error(s) detected.
```

The Oracle Developer Studio 12.6 C++ compiler provides more information about the ambiguity.

```
% CC -c ex2.cc
"ex2.cc", line 19: Error: Type conversion 'A' -> 'B' is ambiguous.
"ex2.cc", line 12: Note: Viable candidate 'A::operator D1() const'.
"ex2.cc", line 4: Note: Viable candidate 'B::B(const A&)'.
```

```
"ex2.cc", line 13: Note: Viable candidate 'A::operator D2&() volatile'.
"ex2.cc", line 19: Error: Cannot use A to initialize B.

You can disable generation of the Notes by using the CC option
    -features=no%note
```

For more information, see the *Oracle Developer Studio 12.6: C++ User's Guide* and the `CC`(1) man page.

♦ ♦ ♦ CHAPTER 3

C Compiler

This chapter describes changes to the C compiler for the Oracle Developer Studio 12.6 release.

- "About the C Compiler" on page 17
- "C Compiler Changes" on page 17

About the C Compiler

This section provides a summary list of the new features and changed functionality introduced in the Oracle Developer Studio 12.6 C 5.14 Compiler release.

The C compiler (`cc`) is the interface to the C compilation system. The compilation process incorporates a preprocessor, compiler, code generator, optimizer, assembler, and link editor. The C compiler processes the supplied options and then executes the various components with the proper arguments. For more information, see the `cc`(1) man page.

C Compiler Changes

The C compiler changes include the changes that are described in "New and Changed Features Common to the Compilers" on page 51, and also the following changes:

- The new compiler options are as follows:
  - `-fcommon` and `-fno-common`
  - `-features=[no%]gcc_enums`
  - `-fexceptions`
  - `-fsemantic-interposition`
  - `-fno-semantic-interposition`

  - `-fshort-enums`
  - `-fvisibility`
  - `-shared`
  - `-std=gnu11 -std=gnu99 -std=gnu90 -std=gnu89 -std=c90` and equivalent aliases
- The new `lint` options are as follows:
  - `-features=[no%]gcc_enums`
  - `-fshort-enums`
- The features associated with the improved compatibility with gcc are as follows:
  - Zero-size structs in types
  - `__builtin_offsetof` intrinsic
  - `__builtin_expect` intrinsic
  - Accepts white space after line-continuation character (backslash `\`)
  - Handles BOM character properly
  - Allows char and short enums under the `-fshort-enums` option
  - Allows unsigned int enums with `-features=[no%]gcc_enums`
  - Allows long and unsigned long enums with `-features=[no%]gcc_enums`
  - Allows packed attribute on enums
- The new attributes supported are as follows:
  - `__packed__`
  - `section(…)`
- Added `__cpuid` instrinsic
- Added support of C11 string literals, `u8""`, `u""`, and `U""`

For more information, see the `cc`(1) man page and the *Oracle Developer Studio 12.6: C User's Guide*.

♦ ♦ ♦ CHAPTER 4

Code Analysis Tools

The code analysis tool suite ensures application reliability and stability by detecting common coding errors, including memory leaks and access violations. This enables the developers to write better code, and code faster with minimal errors.

This chapter describes the new and changed features in the code analysis tools in this Oracle Developer Studio release and contains the following sections:

- "About the Code Analysis Tools" on page 19
- "New discover Features" on page 19
- "New uncover Features" on page 20

About the Code Analysis Tools

The code analysis tools help you make your application more reliable by using static, dynamic, and code coverage analysis to detect many common coding errors, including memory leaks and memory access violations. Previse performs static analysis at compilation and `discover` performs dynamic analysis at application runtime to identify code quality issues. The `uncover` tool analyzes code coverage data to provide information about functions that are not covered by your test suite and tells how you can benefit by covering those functions.

Use the Code Analyzer graphics tool or the `codean` command-line utility to view the three types of analysis. The analysis provides a comprehensive view of your application's vulnerabilities enabling you to improve its correctness and reliability.

New `discover` Features

The following features were added in the `discover` memory analysis tool in this release.

- **Enhanced usage of** `discover` **on previously instrumented binaries** — The `discover` utility automatically saves the binary before overwriting it with the instrumented version of the binary. This utility can now be run directly without having to save the original binary. In addition, `discover` can be run on the same binary as `uncover` without explicitly saving it. For binaries on both Linux and Oracle Solaris, annotations are now generated by default.

For more information, see the *Oracle Developer Studio 12.6: Discover and Uncover User's Guide* and `discover`(1) man page.

New `uncover` Features

The following features were added to the `uncover` code coverage tool in this release.

- **Enhanced usage of** `uncover` **on previously instrumented binaries** — The `uncover` utility automatically saves the binary before overwriting it with the instrumented version of the binary. This utility can now be run directly on instrumented binaries. In addition, `uncover` can be run on the same binary as `discover` without saving it. For binaries on both Linux and Oracle Solaris, annotations are now generated by default.

For more information, see the `uncover`(1) man page and the *Oracle Developer Studio 12.6: Discover and Uncover User's Guide*.

♦ ♦ ♦ CHAPTER 5

Performance Analysis Tools

The performance analysis tools work together to enable you to analyze your application's behavior and find trouble spots that impact performance.

This chapter describes the new and changed features in the performance analysis tools in this Oracle Developer Studio release.

About Performance Analyzer

Performance Analyzer provides insight into the behavior of your application to enable you to find problem areas in your code. It identifies which functions, code segments, and source lines that use the most system resources. Performance Analyzer can profile single-threaded, multithreaded, and multi-process applications, and then present the profiling data to help you identify where you can improve your application's performance.

The Performance Analyzer consists of a set of commands and tools such as:

collect utility	Gathers profiling data on user-level programs.
`er_kernel` utility	Gathers profiling data on the Oracle Solaris kernel.
`er_print` utility	Presents profiling information in text form.
Performance Analyzer GUI	Presents profiling information graphically.

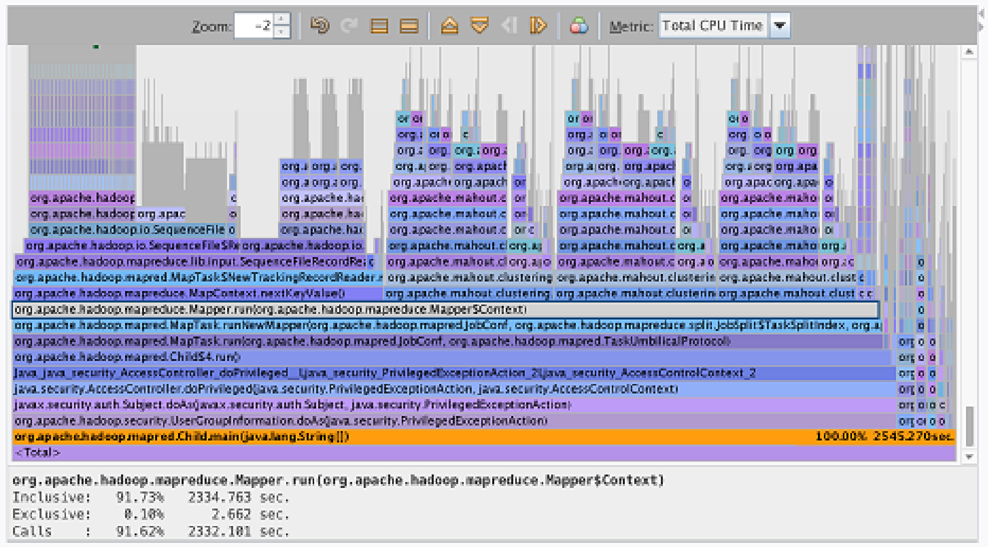
Thread Analyzer is a related tool that enables you to focus on multithreading problems.

For more information, please refer to *Oracle Developer Studio 12.6: Performance Analyzer* and *Oracle Developer Studio 12.6: Performance Analyzer Tutorials*.

Performance Analyzer New Features

This section summarizes the new features of the Performance Analyzer and related tools in this release. For more information, see the Help in Performance Analyzer.

- **New Visual Call Tree View** — A new view, Visual Call Tree, shows the hot paths of execution in a graphical format called a flame graph.

  
  Zoom:
-2
Metric:
Total CPU Time
org.apache.hadoop.mapreduce.MapContext.nextKeyValue()
org.apache.hadoop.mapreduce.Mapper.run(org.apache.hadoop.mapreduce.Mapper$Context)
org.apache.hadoop.mapred.MapTask.runNewMapper(org.apache.hadoop.mapred.JobConf, org.apache.hadoop.mapreduce.split.JobSplit$TaskSplitIndex, org.a
org.apache.hadoop.mapred.MapTask.run(org.apache.hadoop.mapred.JobConf, org.apache.hadoop.mapred.TaskUmbilicalProtocol)
org.apache.hadoop.mapred.Child$4.run()
Java_java_security_AccessController_doPrivileged__Ljava_security_PrivilegedExceptionAction_2Ljava_security_AccessControlContext_2
java.security.AccessController.doPrivileged(java.security.PrivilegedExceptionAction, java.security.AccessControlContext)
javax.security.auth.Subject.doAs(javax.security.auth.Subject, java.security.PrivilegedExceptionAction)
org.apache.hadoop.security.UserGroupInformation.doAs(java.security.PrivilegedExceptionAction)
org.apache.hadoop.mapred.Child.main(java.lang.String[])
100.00% 2545.270sec.
<Total>

  ```
  org.apache.hadoop.mapreduce.Mapper.run(org.apache.hadoop.mapreduce.Mapper$Context)
  Inclusive:   91.73%   2334.763 sec.
  Exclusive:    0.10%      2.662 sec.
  Calls    :   91.62%   2332.101 sec.
  ```

  For more information, see “Flame Graph View” in *Oracle Developer Studio 12.6: Performance Analyzer* manual.

- **New Support for Scala Applications** — The initial support for the analysis of Scala applications has been added.
- **Enhancements to Analyze GNU Inlining** — For binaries compiled with GNU compilers, the PCs and Disassembly views will now identify inlined instructions and the source associated with the leaf-most inlined code.
- **Memoryspace Enhancements** — The Memoryspace view names have been updated to provide user-level view name and descriptions.
- **Improved Analysis of <no Java callstack recorded>** — When a JVM cannot capture the current Java callstack for a given sample, Analyzer will report the sample as *<no Java callstack recorded>*. In this release, the PCs view will describe the error codes associated with missing Java callstacks. In addition, you can use a filter to isolate *<no Java*

callstack recorded> and switch to Machine Mode to analyse the missing Java callstacks and functions.

- **Large Jar File Support** — Java and Scala applications that use large jar files are now supported.
- **Intel Compiler `-ipo` Support** — Performance Analyzer now supports code compiled with the Intel `-ipo` flag.
- **Navigation Features in View History** — New controls support navigation to the previously opened views.

Changes to Command-Line Tools

This section describes changes made to various command-line performance analysis tools. For more information, see the corresponding man pages for each command-line tool.

Changes to Data Collection Tools

Data collection tools include the `collect` command, the `dbx collector` command, and the `er_kernel` command. Each of these tools is used to profile programs to collect data and create experiments that can be read by Performance Analyzer or `er_print`.

`collect` Utility Changes

The `collect` utility is a tool you use to profile your application as it runs to collect data and create an experiment that can be read by Performance Analyzer or `er_print`.

In addition to the changes common to all data collection tools, the `collect` utility is changed in this release as follows:

It is now possible to disable the capture of callstacks during profiling. When profiling Java on x86, disabling the capture of callstacks can reduce the risk of fatal errors related to stack unwind. For more details, see `SP_COLLECTOR_NATIVE_MAX_STACKDEPTH` information in `collect`(1).

`er_kernel` Utility Changes

The `er_kernel` command profiles the Oracle Solaris kernel and generates an experiment that you can examine in Performance Analyzer or `er_print`.

The `er_kernel` utility is changed as follows:

- Kernel profiling is now supported on Oracle Solaris VMs running under OVM (xen).

For more information, see the `er_kernel(1)` man page.

`er_print` Utility Changes

The `er_print` utility generates a plain-text version of the data views presented by the Performance Analyzer. The output is displayed on the standard output window.

The `er_print` utility is changed in this release as follows:

- Command-line editing and history is available on Oracle Solaris.
- Java and Scala applications that use large jar files are now supported.
- Performance Analyzer now supports code compiled with the Intel `-ipo` flag.
- `dumpgc`, a utility that dumps out Java Garbage Collector Events, is available.
- The `-pcs` option will now show a description of various error codes associated with missing Java callstacks.
- For binaries compiled with GNU compilers, the PCs and Disassembly views will now identify inlined instructions and the source associated with the leaf-most inlined code.
- Support for the analysis of Scala applications is now available.

For more information, see the `er_print(1)` man page.

Changes to Other Commands and APIs

The `er_archive` command has the following updates:

- New command option `-d` *path* — This option archives to the common directory specified by the absolute path, *path*.

- New command option `-r` *path* — This option archives to the common directory specified by the relative path, *path*.

The commands `bw`, `ripc`, `spot`, `spot_cmds`, `spot_cmds_timing`, `spot_diff`, and `traps` are no longer supported.

♦ ♦ ♦ CHAPTER 6

Debugging Tools

Oracle Developer Studio provides the command-line `dbx` debugger, and the `dbxtool` graphical tool for using `dbx`. The debugger is also integrated into the IDE. For more information about debugging with the IDE, see Chapter 7, "Oracle Developer Studio IDE".

This chapter contains the following topics:

- "About the `dbx` Debugger" on page 27
- "New and Changed `dbx` Features" on page 27

About the `dbx` Debugger

The `dbx` debugger is an interactive, source-level, postmortem and real-time debugging tool. You can use it at the command line, through the `dbxtool` graphical interface, and in the Oracle Developer Studio IDE. The `dbx` debugger is scriptable and multithread-aware.

New and Changed `dbx` Features

The following features are added or changed in `dbx`:

- The `discover` engine is now supported by Runtime Checking (RTC).
- A vector register can be displayed in different formats. For more information, see `help examine` and `help registers`.
- An embedded python interpreter and a nascent plugin API, primarily intended for the writing of pretty-printing filters is now supported on Linux as well.

For more information, see *Oracle Developer Studio 12.6: Debugging a Program with dbx*, the `dbx(1)` man page, and the `dbx` help file. To access the `dbx` help file, type the following:

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

For more information, issue the `help changes` command under `dbx`, to access the `dbx` help file.

♦ ♦ ♦ CHAPTER 7

Oracle Developer Studio IDE

Oracle Developer Studio IDE integrates many of the components of Oracle Developer Studio for users who prefer a graphical programming environment.

The following topics are covered in this chapter:

About Oracle Developer Studio IDE

Oracle Developer Studio offers a graphical integrated development environment (IDE) that is built on the NetBeans platform, and is configured to use the Oracle Developer Studio C, C++, and Fortran compilers, the `dmake` distributed make command, and `dbx` debugger. The IDE also integrates with some of the Analyzer tools of the analysis suite enabling you to analyze your code without leaving the IDE.

The command to start the IDE is `devstudio`. For more information, see the `devstudio`(1) man page.

For complete documentation of the IDE, see the Help in the IDE. For step-by-step instructions to use the basic features of the IDE, see the *Oracle Developer Studio 12.6: IDE Quick Start Tutorial*.

New and Changed IDE Features

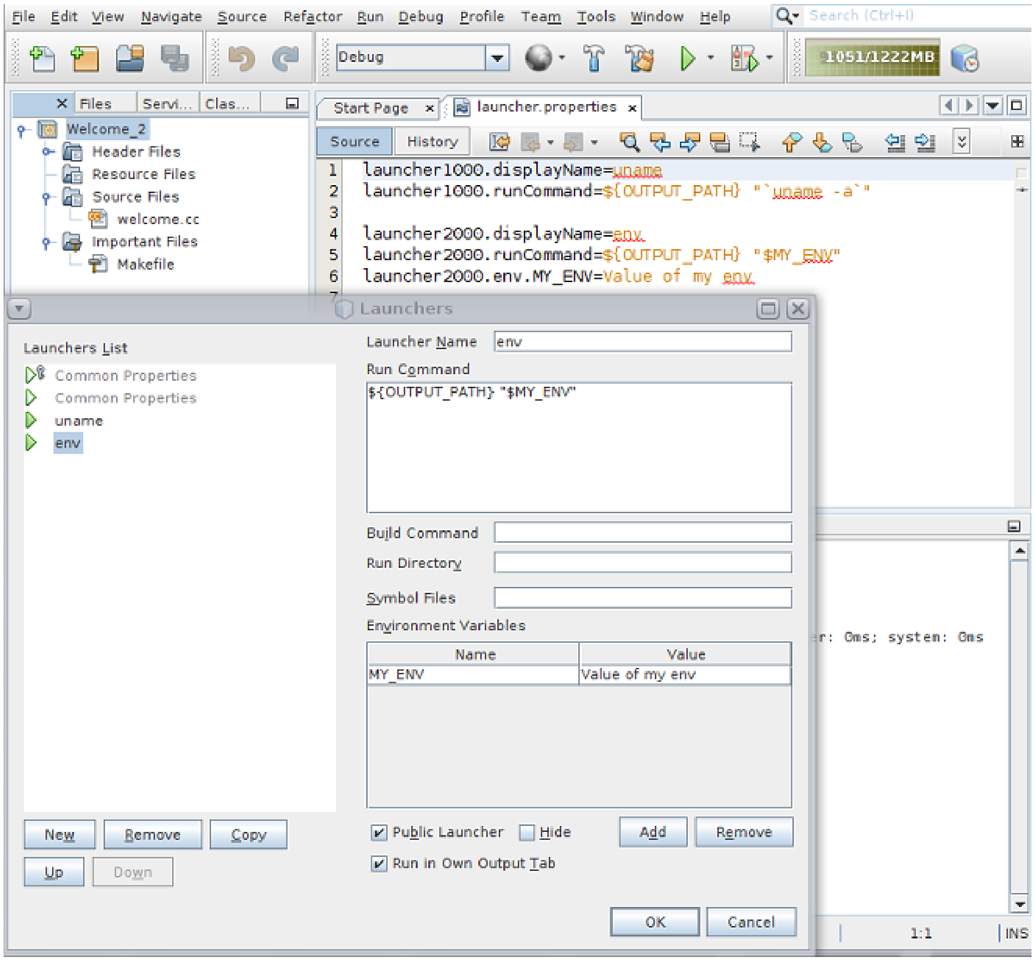
The following features are added or changed in Oracle Developer Studio IDE:

- **Managing Run/Debug Configurations for Launchers** - For more information, see "UI for Run/Debug Launchers" on page 30.
- **Modifying a PATH Variable** - For more information, see "Prepending or Appending a Toolchain Path" on page 31.
- **Environment Variable in Text Fields** - For more information, see "Adding Environment Variables" on page 33.
- **Build Analyzer Based on Tool Collection Wrapper** - For more information, see "Build Analyzer" on page 34
- **Clean C/C++ Cache** - For more information, see "Clean C/C++ Cache" on page 36
- **Distinguish Read/Write Access in Find Usages** - For more information, see "Distinguish Read/Write Access" on page 36
- **Terminal Extensions** - For more information, see "Terminal Feature Extensions" on page 37.
- **New Pinnable Watches** - For more information, see "Pinnable Watches" on page 39

UI for Run/Debug Launchers

The Run/Debug command configurations can be managed using the new UI. To open the configuration window, click *Manage Launchers* from a project's context menu.

FIGURE 2 UI for Run/Debug Launchers



```
launcher1000.displayName=uname
launcher1000.runCommand=${OUTPUT_PATH} "`uname -a`"

launcher2000.displayName=env
launcher2000.runCommand=${OUTPUT_PATH} "$MY_ENV"
launcher2000.env.MY_ENV=Value of my env
```

Launcher Name: env

Run Command: ${OUTPUT_PATH} "$MY_ENV"

Build Command:

Run Directory:

Symbol Files:

Environment Variables

Name	Value
MY_ENV	Value of my env

☑ Public Launcher ☐ Hide

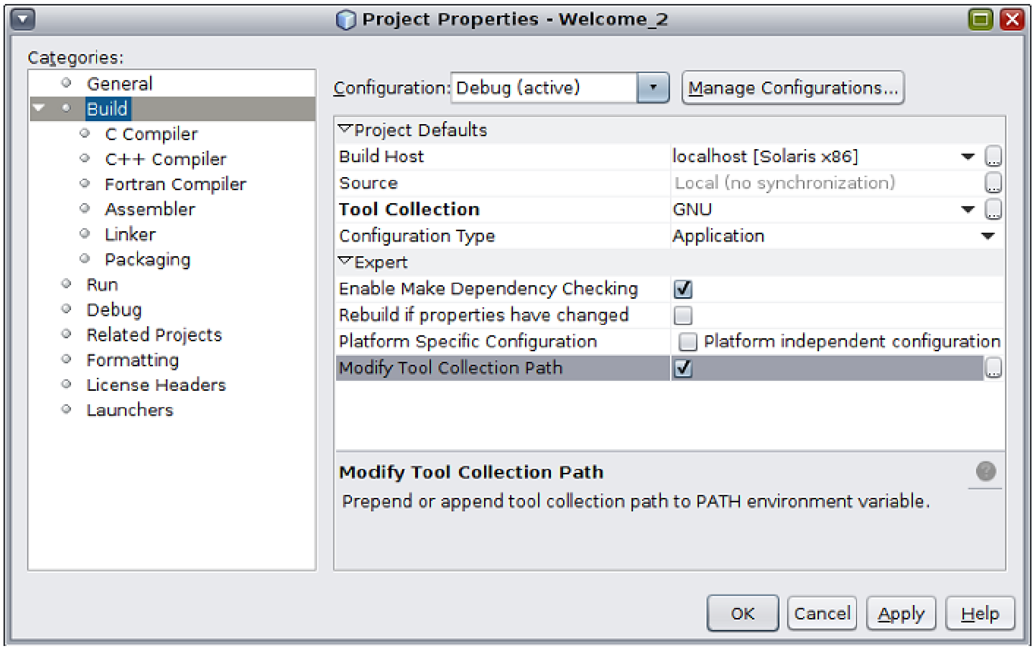
☑ Run in Own Output Tab

```
r: 0ms; system: 0ms
```

Prepending or Appending a Toolchain Path

A New option under *Project Properties* allows you to specify if the toolchain directory should be appended to the PATH variable.

FIGURE 3 Modify Tool Collection Path option



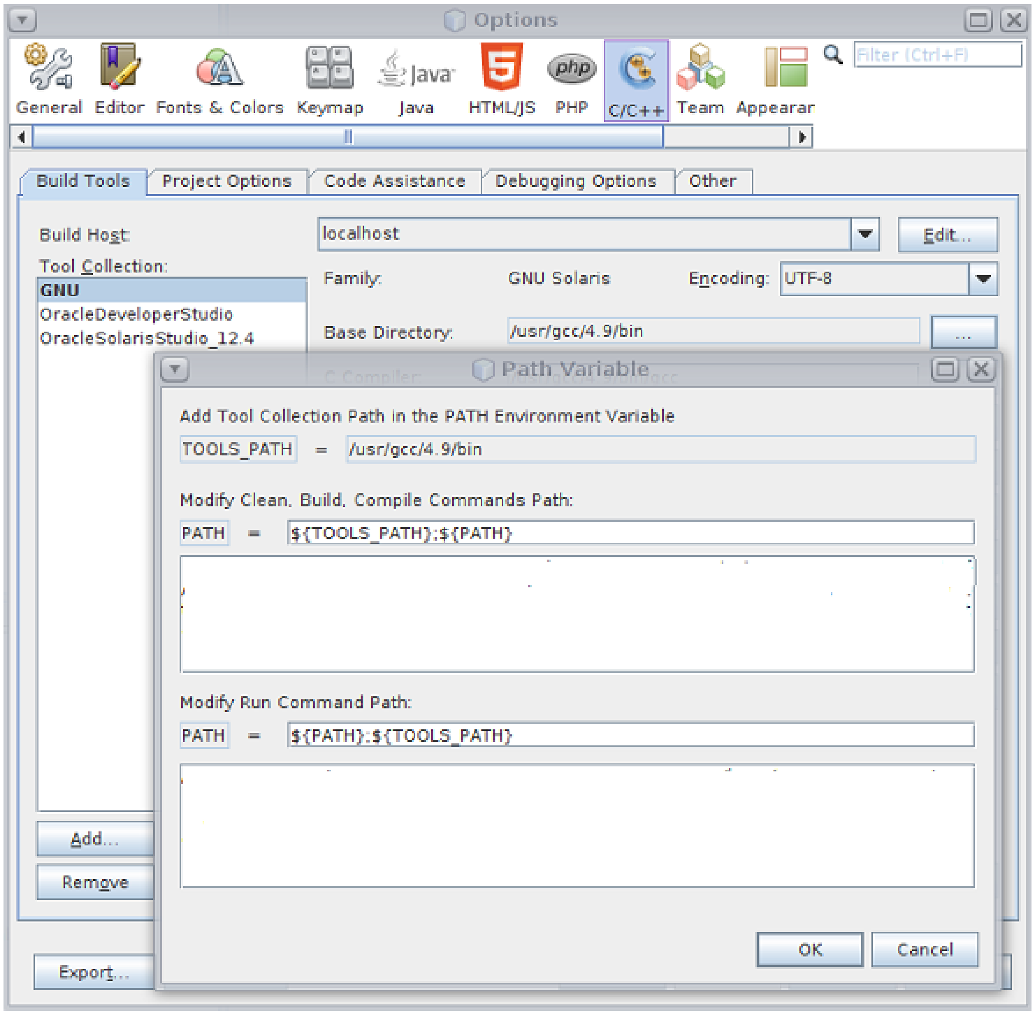
Configuration: Debug (active)

Project Defaults	
Build Host	localhost [Solaris x86]
Source	Local (no synchronization)
Tool Collection	GNU
Configuration Type	Application
Expert	
Enable Make Dependency Checking	☑
Rebuild if properties have changed	□
Platform Specific Configuration	□ Platform independent configuration
Modify Tool Collection Path	☑

Modify Tool Collection Path
Prepend or append tool collection path to PATH environment variable.

By default, the `$TOOLS_PATH` will be prepended. However, you can modify it in the `Options` window.

FIGURE 4 Modifying the Tools Collection Path



Build Host: localhost

Tool Collection:

- GNU
- OracleDeveloperStudio
- OracleSolarisStudio_12.4

Family: GNU Solaris

Encoding: UTF-8

Base Directory: /usr/gcc/4.9/bin

Add Tool Collection Path in the PATH Environment Variable

TOOLS_PATH = /usr/gcc/4.9/bin

Modify Clean, Build, Compile Commands Path:

PATH = ${TOOLS_PATH}:${PATH}

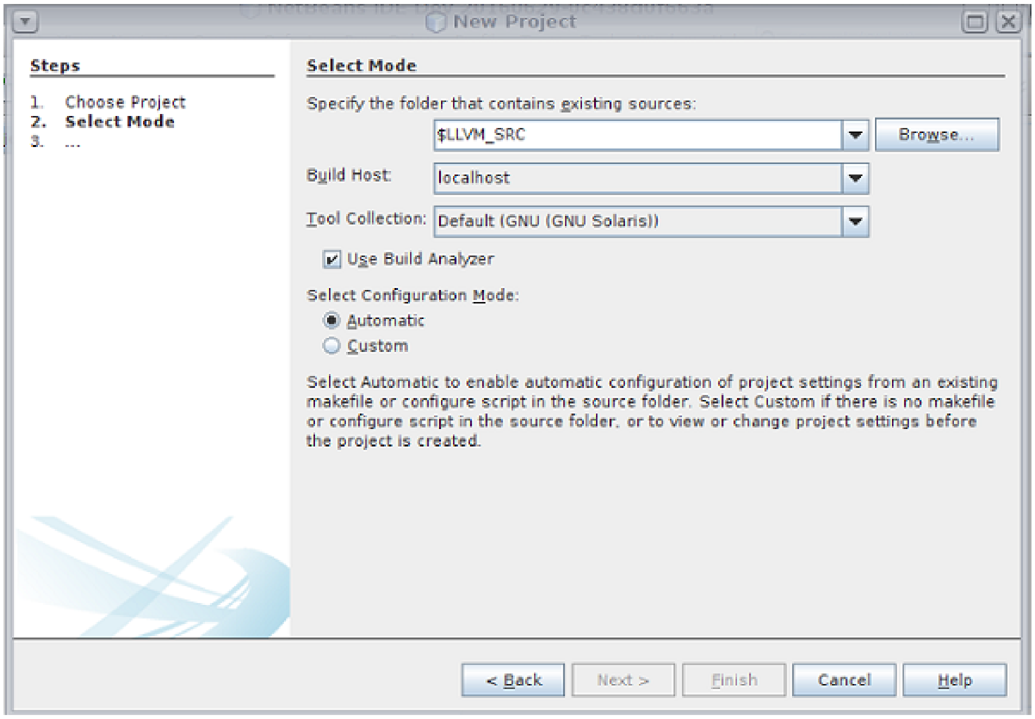
Modify Run Command Path:

PATH = ${PATH}:${TOOLS_PATH}

Adding Environment Variables

You can use environment variables when you create a project with existing sources or from binary.

FIGURE 5 Add Environment Variables



New Project

Steps

1. Choose Project
2. **Select Mode**
3. ...

Select Mode

Specify the folder that contains existing sources: $LLVM_SRC

Build Host: localhost

Tool Collection: Default (GNU (GNU Solaris))

☑ Use Build Analyzer

Select Configuration Mode:

● Automatic

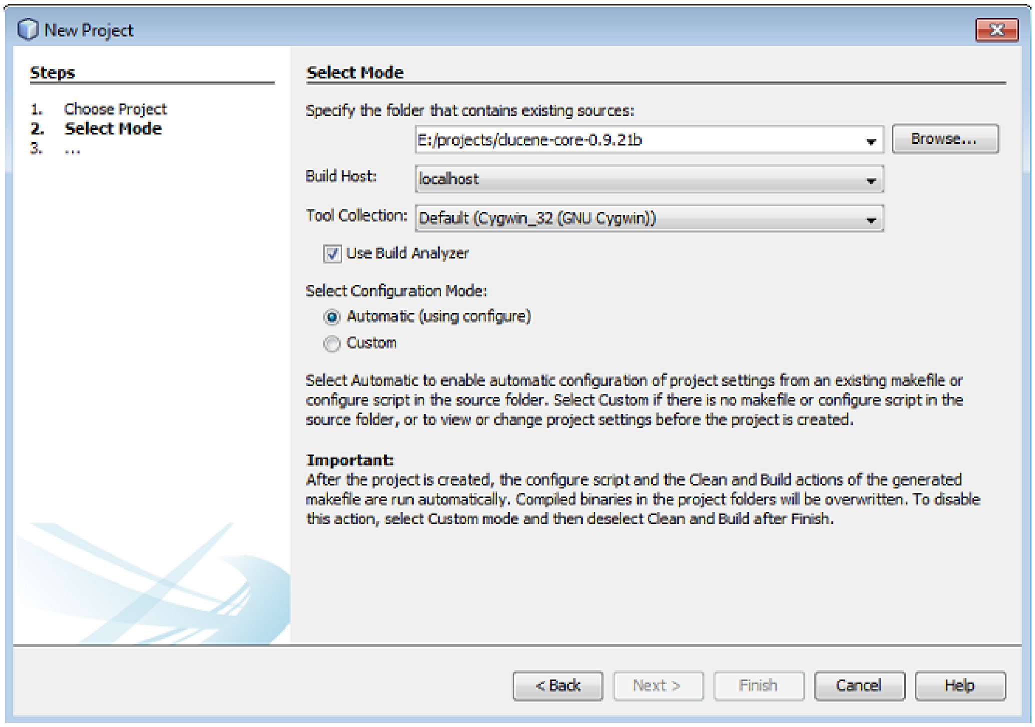
○ Custom

Select Automatic to enable automatic configuration of project settings from an existing makefile or configure script in the source folder. Select Custom if there is no makefile or configure script in the source folder, or to view or change project settings before the project is created.

Build Analyzer

IDE creates tool collection wrapper on Windows and Mac. This wrapper is used to build the Project and as a result, all the compile command lines are stored and used for configuring code assistance.

FIGURE 6 Build Analyzer



Steps

1. Choose Project
2. **Select Mode**
3. ...

Select Mode

Specify the folder that contains existing sources: E:/projects/clucene-core-0.9.21b

Build Host: localhost

Tool Collection: Default (Cygwin_32 (GNU Cygwin))

☑ Use Build Analyzer

Select Configuration Mode:

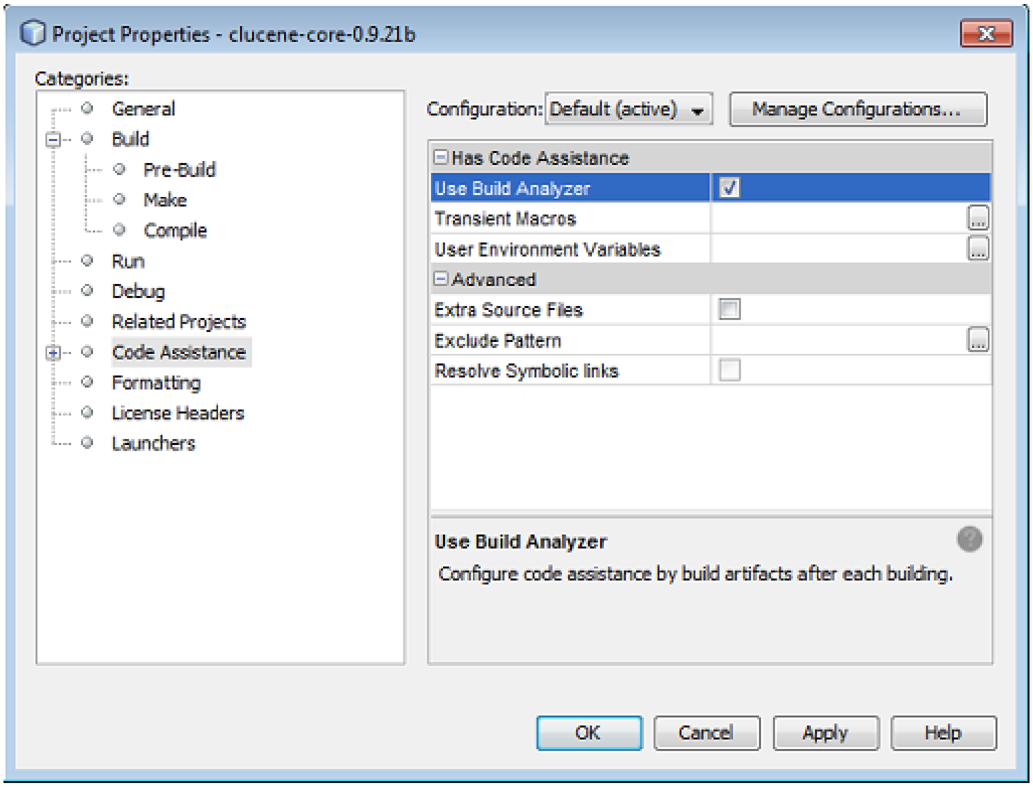
● Automatic (using configure)

○ Custom

Select Automatic to enable automatic configuration of project settings from an existing makefile or configure script in the source folder. Select Custom if there is no makefile or configure script in the source folder, or to view or change project settings before the project is created.

Important:
After the project is created, the configure script and the Clean and Build actions of the generated makefile are run automatically. Compiled binaries in the project folders will be overwritten. To disable this action, select Custom mode and then deselect Clean and Build after Finish.

FIGURE 7 Configuring Code Assistance



Configuration: Default (active)

⊟ Has Code Assistance	
Use Build Analyzer	☑
Transient Macros	...
User Environment Variables	...
⊟ Advanced	
Extra Source Files	☐
Exclude Pattern	...
Resolve Symbolic links	☐

Use Build Analyzer

Configure code assistance by build artifacts after each building.

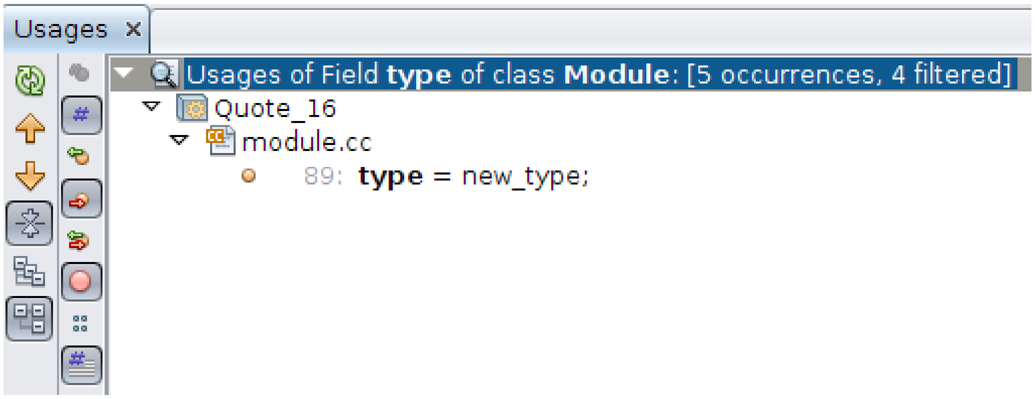
Clean C/C++ Cache

A new option, *Clean C/C++ cache and restart IDE* is added under *Code Assistance*. You can use this option to clean the cache directories and restart the IDE.

Distinguish Read/Write Access

The variables can be marked based on the access provided to them. The variables with read access, write access, or read/write access appear differently.

FIGURE 8 Distinguish Read/Write Access


Usages ×

Usages of Field **type** of class **Module**: [5 occurrences, 4 filtered]

Quote_16

module.cc

89: **type** = new_type;

Terminal Feature Extensions

The following terminal feature extensions are added.

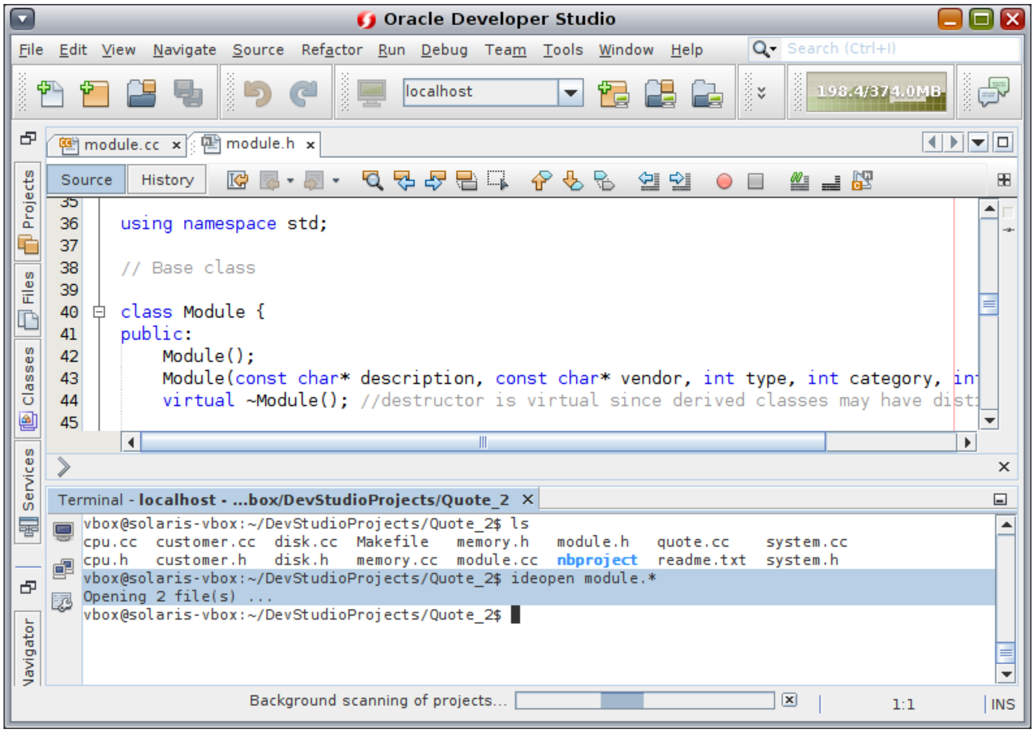
- **Open Files in Editor from terminal** - Choose *Window* → *IDE Tools* → *Terminal* and navigate to the Project folder. You can type the following commands to open the specified files in the Editor.
  - To open a single file, type:

    ```
    ideopen module.cc
    ```
  - To open multiple files, type:

    ```
    ideopen module.cc module.h
    ```
  - To open multiple files with wildcards (supported by your current shell), type:

    ```
    ideopen module.*
    ```

FIGURE 9 Open Files in the Editor


Oracle Developer Studio
File Edit View Navigate Source Refactor Run Debug Team Tools Window Help
Search (Ctrl+I)
localhost
198.4/374.0MB
module.cc
module.h
Source
History
Projects
Files
Classes
Services
Navigator
Terminal - localhost - ...box/DevStudioProjects/Quote_2
Background scanning of projects...
1:1
INS

```
using namespace std;

// Base class

class Module {
public:
    Module();
    Module(const char* description, const char* vendor, int type, int category, in
    virtual ~Module(); //destructor is virtual since derived classes may have dist
```

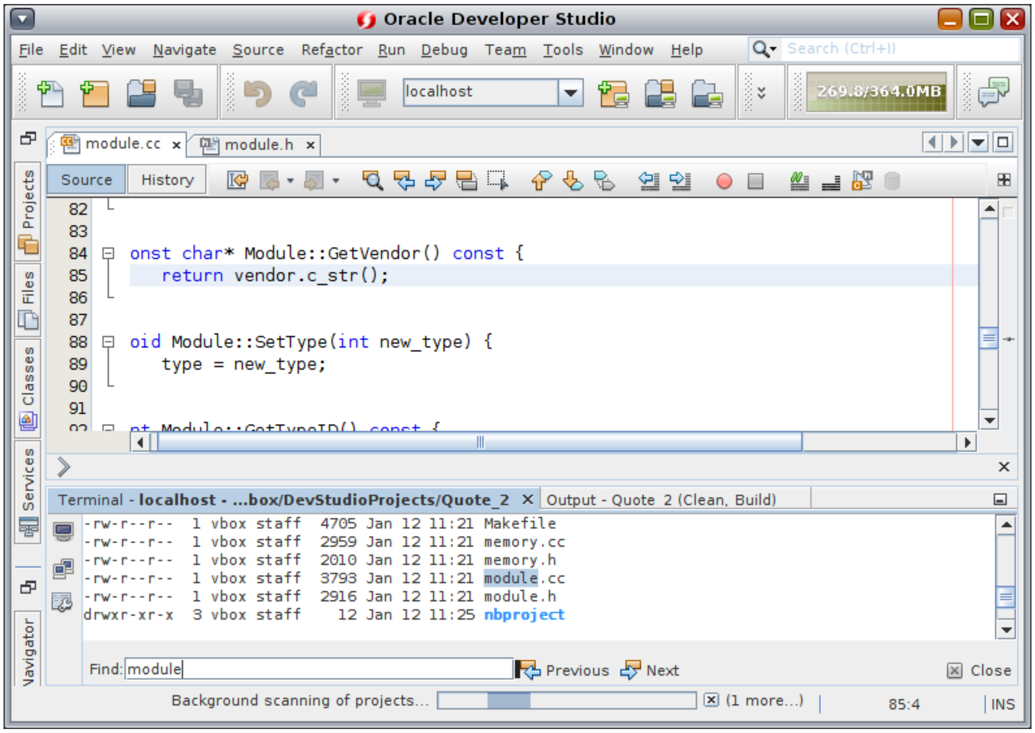
```
vbox@solaris-vbox:~/DevStudioProjects/Quote_2$ ls
cpu.cc   customer.cc  disk.cc  Makefile   memory.h   module.h   quote.cc    system.cc
cpu.h    customer.h   disk.h   memory.cc  module.cc  nbproject  readme.txt  system.h
vbox@solaris-vbox:~/DevStudioProjects/Quote_2$ ideopen module.*
Opening 2 file(s) ...
vbox@solaris-vbox:~/DevStudioProjects/Quote_2$ 
```

- **Support for Keyboard Shortcuts** - This feature enables the user to use the keyboard to open a new tab and to switch between the opened tabs. Use *Ctrl+Alt+Shift+T* to open a new tab and *Alt+1*, *Alt+2* ... to switch between the tabs.
- **Accessing Hyperlinks from the Terminal** - If a program's output prints a link to a file in the Terminal, and if the user clicks on this link in Terminal, the IDE will open this file in the Editor. You can print a hyperlink in the terminal with an escape sequence. For example:

  ```
  fprintf(stdout, "(\033]10;%s;%s\007)]\n", "/home/ilia/NetBeansProjects/
  CppApplication_48/main.cpp:20", "main.cpp:20")
  ```

- **Search in Terminal** - Search feature is now implemented in the terminal. To search, use *Ctrl+Shift+F* or *[Right click]* and choose *Find in Terminal*. You can also navigate between the found occurrences. Use *Shift + F3* to navigate to the previous occurrence and *(F3, Enter)* to navigate to the next occurrence.

FIGURE 10 Search in Terminal


Oracle Developer Studio
File Edit View Navigate Source Refactor Run Debug Team Tools Window Help
Search (Ctrl+I)
localhost
269.8/364.0MB
module.cc
module.h
Source
History
Projects
Files
Classes
Services
Navigator
Terminal - localhost - ...box/DevStudioProjects/Quote_2
Output - Quote_2 (Clean, Build)
Previous
Next
Close
Background scanning of projects...
(1 more...)
85:4
INS

```
onst char* Module::GetVendor() const {
    return vendor.c_str();
}

oid Module::SetType(int new_type) {
    type = new_type;
}

nt Module::GetTypeID() const {
```

```
-rw-r--r--  1 vbox staff  4705 Jan 12 11:21 Makefile
-rw-r--r--  1 vbox staff  2959 Jan 12 11:21 memory.cc
-rw-r--r--  1 vbox staff  2010 Jan 12 11:21 memory.h
-rw-r--r--  1 vbox staff  3793 Jan 12 11:21 module.cc
-rw-r--r--  1 vbox staff  2916 Jan 12 11:21 module.h
drwxr-xr-x  3 vbox staff    12 Jan 12 11:25 nbproject
```

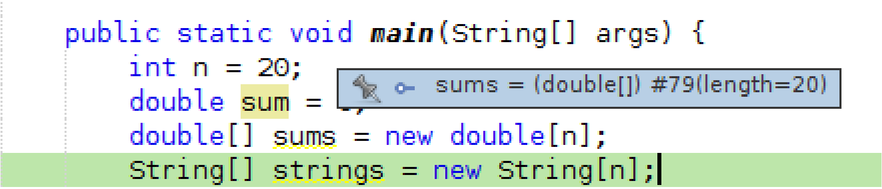
Find: module

Pinnable Watches

Watches can now be pinned into the editor area, When you hover your mouse over a variable or a selection, a tooltip with its value is shown.

The tooltip now contains a pin icon, which when clicked, creates a watch, pinned into the editor.

FIGURE 11 Pin Icon in the Tooltip

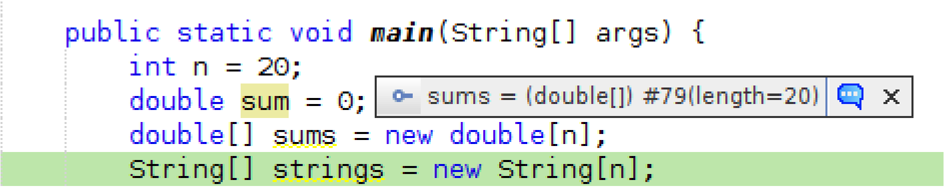


```
public static void main(String[] args) {
    int n = 20;
    double sum =
    double[] sums = new double[n];
    String[] strings = new String[n];
```

sums = (double[]) #79(length=20)

The pin watch window is displayed instead of the tooltip. The location of this window can be adjusted by dragging with a mouse.

FIGURE 12 Pin Watch Window

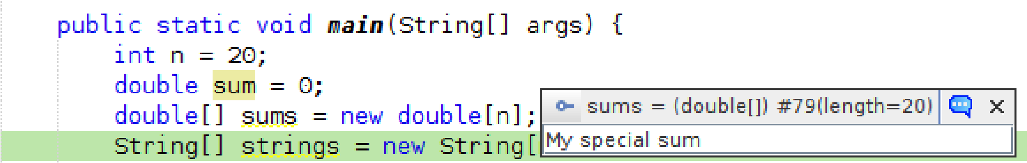


```
public static void main(String[] args) {
    int n = 20;
    double sum = 0;
    double[] sums = new double[n];
    String[] strings = new String[n];
```

sums = (double[]) #79(length=20)

The pin watch window contains two icons on the right: *comment* icon and *close* icon. Clicking on the comment icon shows a text field where you can add comments about the watch.

FIGURE 13 Comment and Close Icons



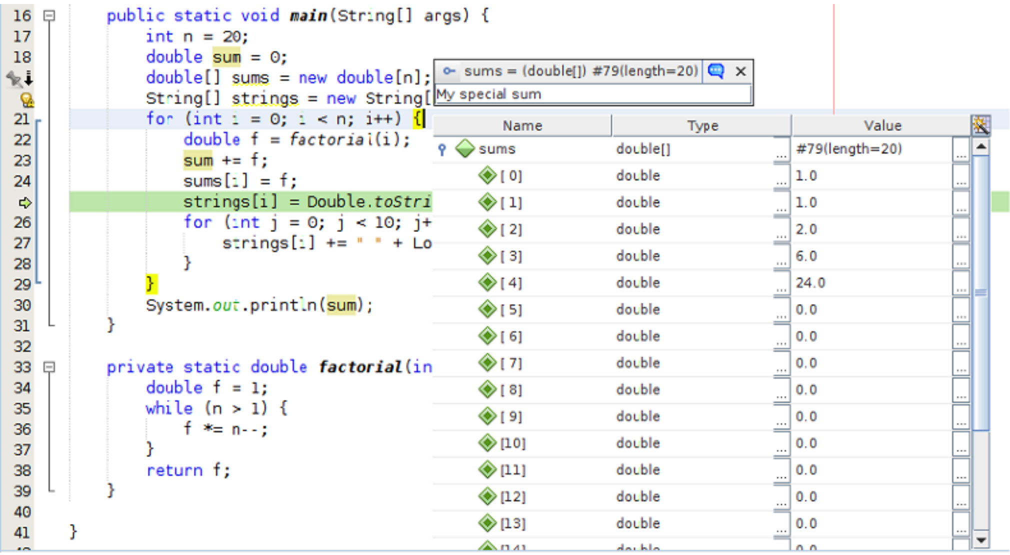
```
public static void main(String[] args) {
    int n = 20;
    double sum = 0;
    double[] sums = new double[n];
    String[] strings = new String[
```

sums = (double[]) #79(length=20)

My special sum

When the pin watch displays structured values, it shows an expansion icon on the left. When expanded, a view with its children is shown:

FIGURE 14 Expansion Icon



```
    public static void main(String[] args) {
        int n = 20;
        double sum = 0;
        double[] sums = new double[n];
        String[] strings = new String[
        for (int i = 0; i < n; i++) {
            double f = factorial(i);
            sum += f;
            sums[i] = f;
            strings[i] = Double.toStri
            for (int j = 0; j < 10; j+
                strings[i] += " " + Lo
            }
        }
        System.out.println(sum);
    }

    private static double factorial(in
        double f = 1;
        while (n > 1) {
            f *= n--;
        }
        return f;
    }

}
```

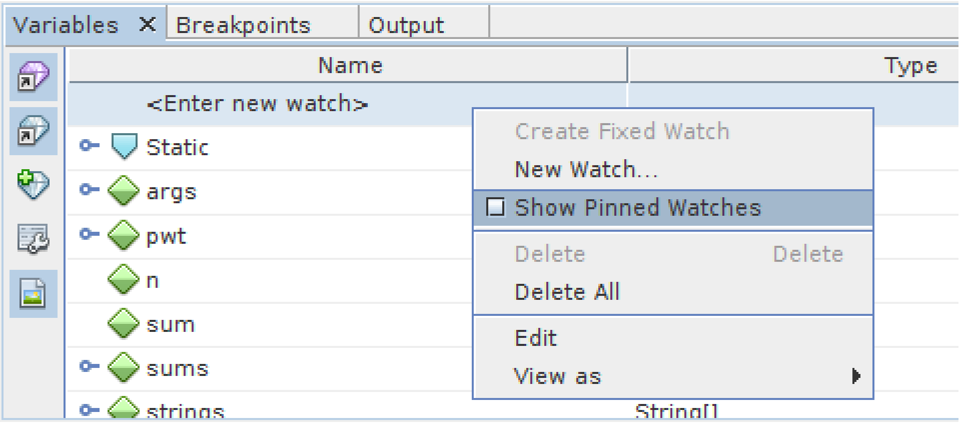
sums = (double[]) #79(length=20)

My special sum

Name	Type	Value
sums	double[]	#79(length=20)
[0]	double	1.0
[1]	double	1.0
[2]	double	2.0
[3]	double	6.0
[4]	double	24.0
[5]	double	0.0
[6]	double	0.0
[7]	double	0.0
[8]	double	0.0
[9]	double	0.0
[10]	double	0.0
[11]	double	0.0
[12]	double	0.0
[13]	double	0.0
[14]	double	0.0

Variables/Watches window does not show the pinned watches by default. However, it is possible to view the pinned watches by using *Show Pinned Watches* context action.

FIGURE 15 Show Pinned Watches Option



Docker

Oracle Developer Studio provides Docker support for the users. The Oracle Developer Studio IDE provides highlighting and auto completion for Docker files, that enables the user to interact with Docker easily. The users can connect to Docker daemon running on their machine or on a remote machine. The users can use the IDE to download images from Docker Hub, build their own images, and run and manage Docker containers.

About Docker

Docker is an open source container engine which provides for the creation, execution, and management of lightweight application containers. Docker allows users to run instances from published images, which are available through a public repository. A domain-specific language, called Dockerfile, may also be used to create new images. Once authored, images can be run in any Docker environment.

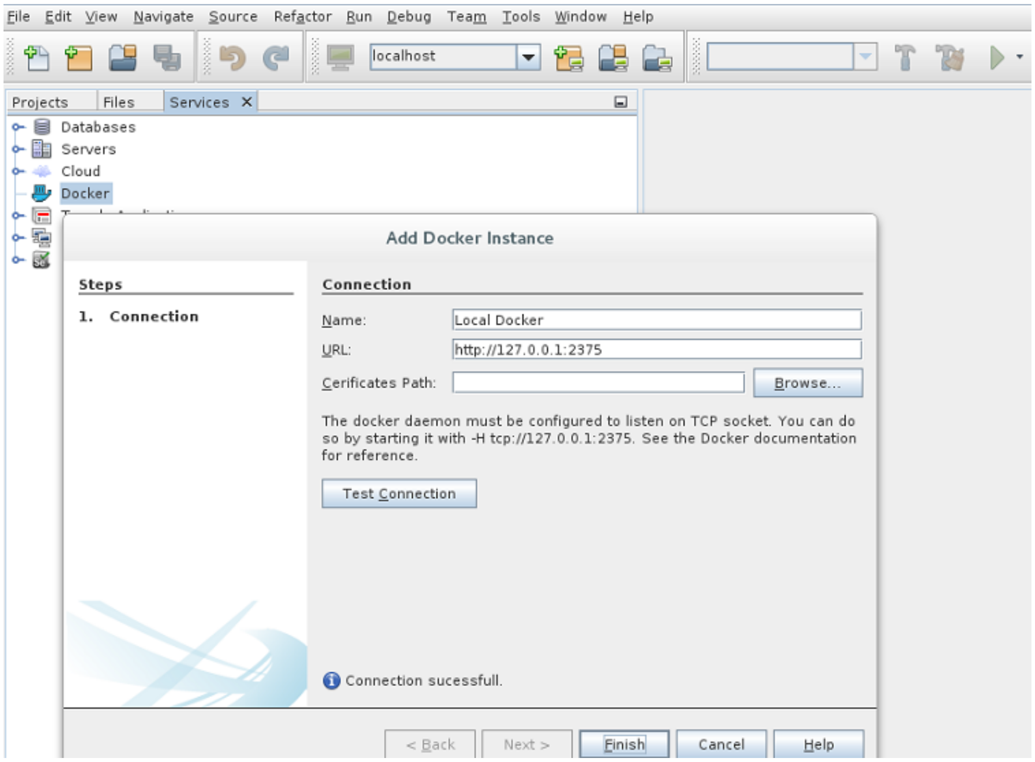
Interacting with Docker

This section describes the steps to interact with Docker using Oracle Developer Studio IDE.

▼ How to Interact with Docker using Oracle Developer Studio IDE.

The users can perform the below tasks on a Docker daemon running on their local machine or on a remote machine. However, interacting with Docker daemon running on a remote machine provides additional opportunities to use Oracle Developer Studio IDE with cloud environment.

1. **Connect to Docker daemon using `http`.**



Name: Local Docker

URL: http://127.0.0.1:2375

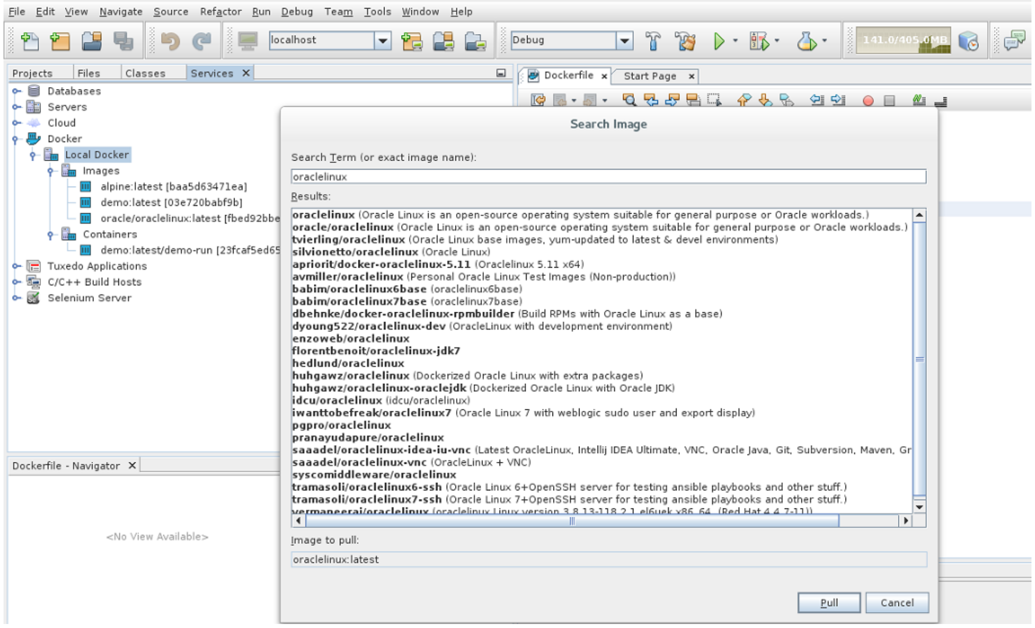
Cerificates Path:

The docker daemon must be configured to listen on TCP socket. You can do so by starting it with -H tcp://127.0.0.1:2375. See the Docker documentation for reference.

Connection sucessfull.

Specify the name of the Docker and the URL under `Name` and `URL` fields respectively and click *Finish*.

2. **In the *Services View*, the new node will appear using which can be used to search and download any image from Docker Hub.**



Search Image

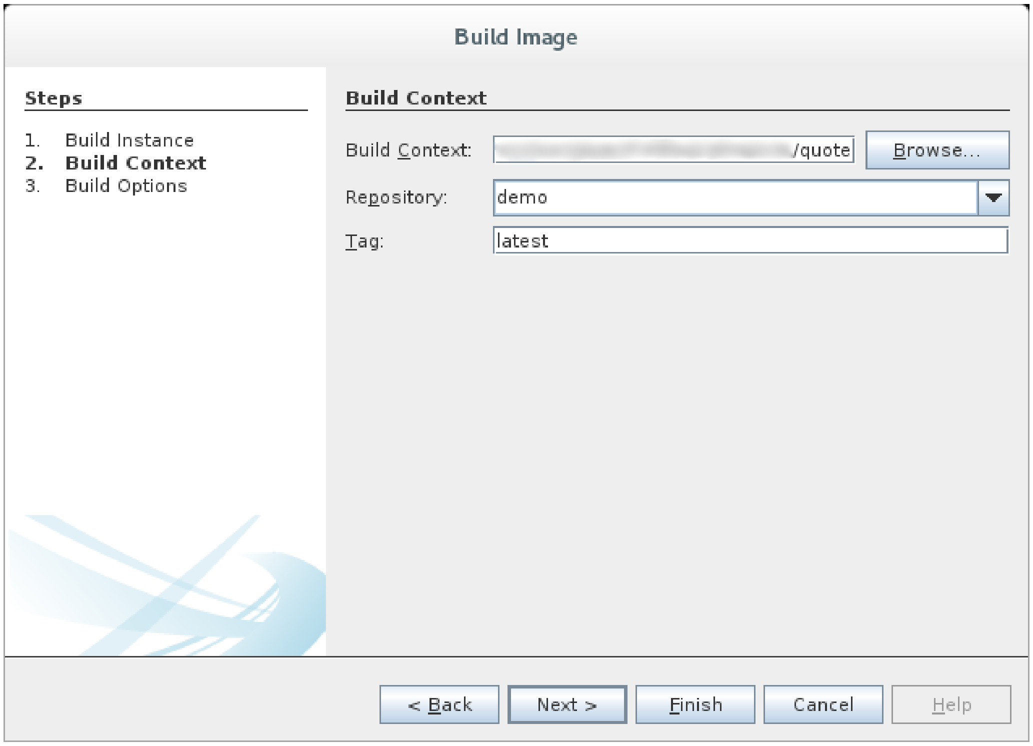
Search Term (or exact image name): oraclelinux

Results:

```
oraclelinux (Oracle Linux is an open-source operating system suitable for general purpose or Oracle workloads.)
oracle/oraclelinux (Oracle Linux is an open-source operating system suitable for general purpose or Oracle workloads.)
tvierling/oraclelinux (Oracle Linux base images, yum-updated to latest & devel environments)
silvionetto/oraclelinux (Oracle Linux)
apriorit/docker-oraclelinux-5.11 (Oraclelinux 5.11 x64)
avmiller/oraclelinux (Personal Oracle Linux Test Images (Non-production))
babim/oraclelinux6base (oraclelinux6base)
babim/oraclelinux7base (oraclelinux7base)
dbehnke/docker-oraclelinux-rpmbuilder (Build RPMs with Oracle Linux as a base)
dyoung522/oraclelinux-dev (OracleLinux with development environment)
enzoweb/oraclelinux
florentbenoit/oraclelinux-jdk7
hedlund/oraclelinux
huhgawz/oraclelinux (Dockerized Oracle Linux with extra packages)
huhgawz/oraclelinux-oraclejdk (Dockerized Oracle Linux with Oracle JDK)
idcu/oraclelinux (idcu/oraclelinux)
iwanttobefreak/oraclelinux7 (Oracle Linux 7 with weblogic sudo user and export display)
pgpro/oraclelinux
pranayudapure/oraclelinux
saaadel/oraclelinux-idea-iu-vnc (Latest OracleLinux, Intellij IDEA Ultimate, VNC, Oracle Java, Git, Subversion, Maven, Gr
saaadel/oraclelinux-vnc (OracleLinux + VNC)
syscomiddleware/oraclelinux
tramasoli/oraclelinux6-ssh (Oracle Linux 6+OpenSSH server for testing ansible playbooks and other stuff.)
tramasoli/oraclelinux7-ssh (Oracle Linux 7+OpenSSH server for testing ansible playbooks and other stuff.)
vermaneerai/oraclelinux (oraclelinux Linux version 3.8.13-118.2.1.el6uek.x86_64 (Red Hat 4.4.7-11))
```

Image to pull: oraclelinux:latest

3. **In the** *Projects view*, **right click on Dockerfile and click** *Build* **to build images. In the** *Build Image window*, **select** *Build Context* **and enter the details.**



Build Image

Steps

1. Build Instance
2. **Build Context**
3. Build Options

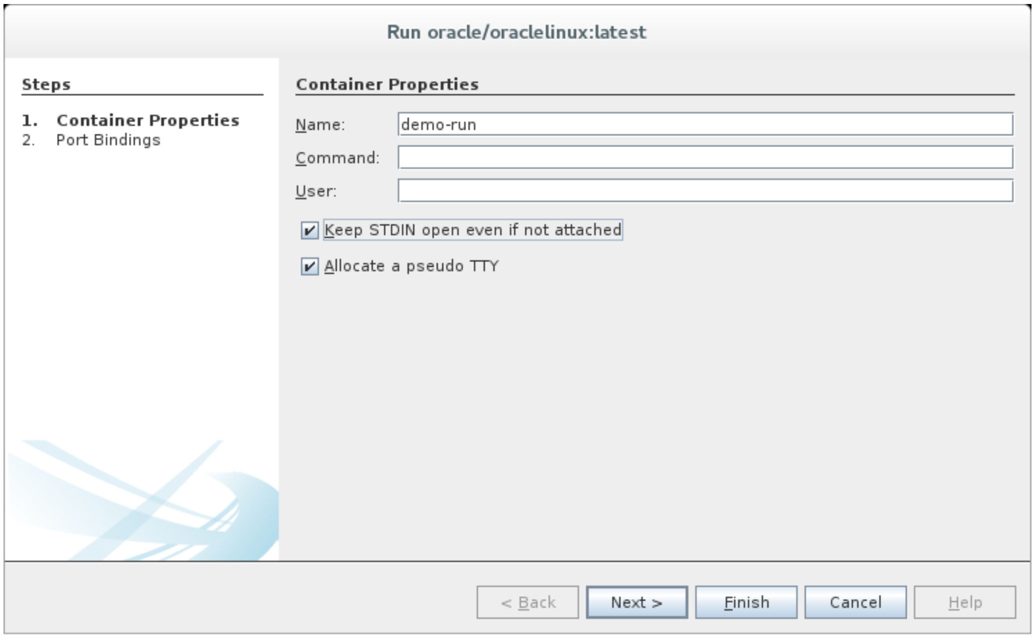
Build Context

Build Context: [redacted]/quote

Repository: demo

Tag: latest

4. **Run Docker container. Use the IDE terminal to interact with the containers.**



Run oracle/oraclelinux:latest

Steps

1. **Container Properties**
2. Port Bindings

Container Properties

Name: demo-run

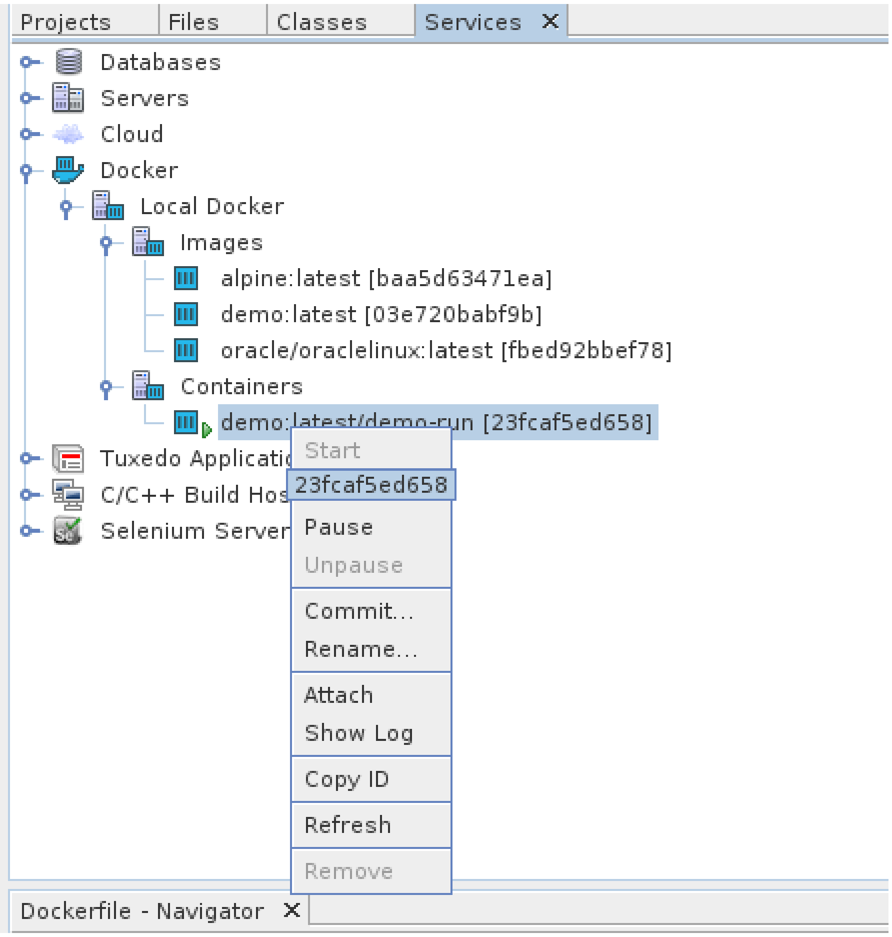
Command:

User:

☑ Keep STDIN open even if not attached

☑ Allocate a pseudo TTY

5. **In the** *Services view*, **right click on the container to manage the containers.**



♦ ♦ ♦ CHAPTER 8

OpenMP API

This chapter describes the changes for OpenMP API support in this release of Oracle Developer Studio.

OpenMP

This section discusses new features and updates to the OpenMP API.

- Improved Dynamic Loop Schedule — The OpenMP 4.5 specification clarifies how the dynamic loop schedule assigns chunks to threads. The chunks of loop iterations are not guaranteed to be assigned to the threads in monotonic order. For example, in the following OpenMP dynamic loop, the 50th iteration is executed before the 0th iteration.

  ```
  #pragma omp parallel for num_threads(2) schedule(dynamic)
  for (i=0;i<100;i++) A[i] = B[i];
  ```

  This enables more efficient dynamic loop scheduling implementation.
- Priority Clause for OpenMP Task Pragma — The priority clause feature includes the following:
  - `OMP_MAX_TASK_PRIORITY` environment variable
  - `omp_get_max_task_priority()` API routine

  For more information, see **OpenMP 4.5 specification** `section 2.9.1` for priority clause for task construct, `section 4.14` for `OMP_MAX_TASK_PRIORITY` environment variable, and `section 3.2.36` for `omp_get_max_task_priority` API routine.
- Query Functions for Thread Affinity (Processor Binding) — The following query API routines are added:
  - `omp_get_num_places()`
  - `omp_get_place_num_procs()`
  - `omp_get_place_proc_ids()`

- `omp_get_place_num()`
- `omp_get_partition_num_places()`
- `omp_get_partition_place_nums()`

  For more information, see `Section 3.2.23` on page 256 to `Section 3.2.28` on page 261 in **OpenMP 4.5 specification**.

For more information about OpenMP, see *Oracle Developer Studio 12.6: OpenMP API User's Guide*.

CHAPTER 9

Other Changes

This chapter describes new and changed features for other components of the Oracle Developer Studio software.

- "Changes to Compilers" on page 51
- "Performance Library Changes" on page 54

Changes to Compilers

The following section describes changes made to the compilers and includes the following topics:

- "New and Changed Features Common to the Compilers" on page 51
- "Fortran Compiler" on page 53

New and Changed Features Common to the Compilers

The following changes were made to the C, C++, and Fortran compilers since the previous release. Details can be found in the compiler man pages. The changes specific to the C++ compiler are detailed in Chapter 2, "C++ Compiler". The changes specific to the C compiler are detailed in Chapter 3, "C Compiler".

GCC Compatible Option `-fvisibility`

The `-fvisibility=v` option is equivalent to the `-xldscope` option as follows:

-fvisibility Options	Equivalent -xldscope Options
default	global
internal	hidden
protected	symbolic
hidden	hidden

GCC Compatible Option -shared

In Oracle Developer Studio 12.6, the -shared option is compatible with GCC. The -shared option creates a shared library, similar to the -G option. By default, the -shared option links the same libraries that would be linked when creating an executable program, whereas the -G option does not automatically link default libraries. In previous releases, the -shared option was equivalent to the -G option.

New Command Options

The following list describes the new command options, common to all the compilers.

- **New options for SPARC M8/T8** — The -xchip and -xtarget values are now available for SPARC M8/T8 systems.
- **New options for x86 processor Skylake** — The -xchip=skylake, -xtarget=skylake, and -xarch=avx512 values are now available for Intel Skylake processor.
- **Default option on Linux** — The default option on Linux is -xannotate.
- The command options -fstrct-aliasing and -fno-strict-aliasing are now available.

New Macro for Oracle Solaris

A new macro, __SUNOS_Release is now available for Oracle Solaris only.

__SunOS_RELEASE on Oracle Solaris

A hex value, 0xRRrrmm, representing the Oracle Solaris release, where RR.rr is the output of the sysinfo (SI_RELEASE) system call, or the uname -r command, with leading zeros added when needed. The mm digits are reserved for possible future micro releases. All the digits are in decimal.

For example, Oracle Solaris 11, which is SunOS 5.11: __SunOS_RELEASE has the value 0x051100.

The value of __SunOS_RELEASE for an older Oracle Solaris release is always less than the value for a later release. For example,

```
#if __SunOS_RELEASE >= 0x051100 // Solaris 11 or later
```

Inlining Behaviour of the Compilers

The Studio compilers automatically inline routines whose body is smaller than the calling overhead at -O3 level. To control the functions that are automatically inlined, use -xinline=list option.

Compress Debug Sections

The -xcompress=debug option can be used to compress debug sections using the format specified by the -xcompress_format option. Support for the equivalent gcc option -gz has also been added.

-xdebugformat=stabs Option

The -xdebugformat=stabs option is removed.

Fortran Compiler

The Fortran compiler supports technical and scientific application development with record-setting runtime performance and compatibility options for the Fortran77, Fortran90, and Fortran95 standards. The majority of Fortran 2003 features and OpenMP 4.5 support is included. The Fortran compiler uses the same high-performance code generation technology as the C and C++ compilers, ensuring that the resulting application generates the highest-performance parallel code for the newest SPARC and x86-based Oracle systems.

The Fortran compiler changes include the changes that are described in "New and Changed Features Common to the Compilers" on page 51 and the following:

- Support for Fortran global program checking (GPC) is removed.
- Parameterized Derived Types, introduced in the Fortran 2003 standard, is now supported.

For more information, see the f95(1) man page and the *Oracle Developer Studio 12.6: Fortran User's Guide*.

Performance Library Changes

Oracle Developer Studio Performance Library is a set of optimized, high-speed mathematical subroutines for solving linear algebra and other numerically intensive problems. Oracle Developer Studio Performance Library is based on a collection of public domain subroutines available mostly from Netlib at `http://www.netlib.org`. Oracle has enhanced these public domain subroutines and bundled them as the Oracle Developer Studio Performance Library.

For this release, LAPACK in Oracle Developer Studio Performance Library is upgraded to version 3.6.1. All of the new features in LAPACK 3.6.1 are implemented in Oracle Developer Studio Performance Library. For more information about the changes, see `http://www.netlib.org/lapack/lapack-3.6.1.html`.

For more information, see *Oracle Developer Studio 12.6: Performance Library User's Guide*.

Index