SourcePro
by Perforce

SourcePro for Open Source Databases

Rogue Wave® Software SourcePro®

Version 2023.1

PERFORCE

www.perforce.com

SOURCEPRO FOR OPEN SOURCE DATABASES

ACKNOWLEDGMENTS

Rogue Wave by Perforce
https://www.perforce.com/products/sourcepro
https://help.perforce.com/sourcepro

Contents

Chapter 1, Introduction

Welcome

SourcePro for Open Source Databases provides prebuilt libraries for the SourcePro products and support within the SourcePro DB product for open source databases PostgreSQL and MySQL.

SourcePro for Open Source Databases creates a powerful foundation for the initial development of secure, stable applications — critical aspects of any application that cannot be easily added later but must be integrated from the ground up.

Upgrading to the full SourcePro is an easy migration that would provide support for full-featured, industry-standard databases, as well as all source code, allowing easy integration into any supported build environment.

SourcePro for Open Source Databases provides:

- ***Cross-platform extensibility.*** SourcePro C++ interfaces are consistent across platforms. In many situations, you can simply recompile and run on a different platform without rewriting code.
- ***Integration and neutrality.*** The components themselves work together across product lines, and work with a variety of industry-leading operating systems, compilers, databases, and threading libraries.
- ***Security and stability.*** SourcePro provides a reliable foundation for your applications, proven in hundreds of thousands of applications for more than two decades. Its code base is extensively tested and verified using industry-standard tools such as Valgrind and Klocwork.
- ***Broad functionality.*** SourcePro products contain the components you need for virtually any programming task in C++, with specialized functionality for networking, database access, and business analysis.

This product contains prebuilt versions of SourcePro libraries for use on supported platforms, as described in the ***Supported_Platforms.pdf***, in the distribution under `<installdir>\`. This document is also on the web at https://help.perforce.com/sourcepro/current/Supported_Platforms.pdf. Both release and debug binaries are included.

SourcePro for Open Source Databases also includes complete documentation for each library, as well as code examples illustrating the key features of each library.

Product Overview

Each SourcePro product provides an integrated, comprehensive set of classes that addresses a specific programming area in C++. SourcePro products simplify development projects and reduce development time by providing intuitive, easy-to-use, C++ interfaces to complex constructs.

SourcePro for Open Source Databases includes:

- *The Essential Tools Module of SourcePro Core*, an internationalized set of C++ classes that provide the basic building blocks for most C++ applications. These include classes for dates/times, strings, virtual streams, collections, internationalization, file systems, and others.
- *SourcePro DB* contains database and data manipulation classes that encapsulate SQL 92 in a C++ interface that is consistent across all supported platforms and databases.

  The included databases access modules provide access support for the open source databases PostgreSQL and MySQL.

Both release and debug binaries are included.

Each SourcePro product is composed of ***modules***; modules are composed of either ***packages*** or ***class groups***; packages and class groups are composed of ***classes***.

For a more in-depth overview of the full SourcePro product, and code examples demonstrating interoperability, see the Introduction to SourcePro. This overview document is available both on the web at https://help.perforce.com/sourcepro/current/HTML/index.html#page/Introduction_to_SourcePro/startug-Title.html and through the main documentation access page at `<buildspace>\docs\index.html`, where `<buildspace>` indicates the location where you installed SourcePro for Open Source Databases.

Software Requirements

The SourcePro for Open Source Databases contains precompiled dynamic or shared libraries, also called “DLLs” on Windows. These dynamic libraries are built with multithreading support and using the native C++ Standard Library. The included DB Access Modules provide support for a limited set of databases, as specified in General Software Requirements.

General Software Requirements

To use SourcePro for Open Source Databases, you need a supported combination of operating system and compiler, as specified in the ***Supported_Platforms.pdf***, in the distribution under `<installdir>\` or on the web at https://help.perforce.com/sourcepro/current/Supported_Platforms.pdf..

The compiler is needed to build the examples.

To connect to a database and create examples using SourcePro DB components, you need to install the database client software and establish connectivity with the database server. The supported databases for each access module are also identified in the ***Supported_Platforms.pdf***.

Documentation

SourcePro for Open Source Databases comes with a full set of SourcePro documentation, covering all products and modules, as listed at `<buildspace>\docs\index.html`, where `<buildspace>` is the directory location where you installed this product. This page provides single-point access to the HTML and PDF documentation. Table 2 lists the individual documents and shows their locations.

Table 1 – Documentation resources for SourcePro products

Name, location, and description
HTML documentation `<buildspace>\docs\html\index.html` All HTML documentation in a single, unified document set. The set includes this SourcePro for Open Source Databases guide, the ***Introduction to SourcePro***, all user guides, and the API reference guide.
SourcePro for Open Source Databases `<buildspace>\docs\pdf\SourcePro_Open_Source_Databases.pdf` An introduction to SourcePro for Open Source Databases. (This document)
Introduction to SourcePro: `<buildspace>\docs\pdf\Introduction_to_SourcePro.pdf` Organization of the SourcePro products and code examples demonstrating interoperability.
User Guide, one per module: `<buildspace>\docs\pdf\<module-name>_<document-name>.pdf` Task-oriented instruction on using a module.
Reference Guide: `<buildspace>\docs\html\sourceproref\index.html` Descriptions of classes and other significant elements of a module's API.
Release Notes `<buildspace>\Release_Notes.html` The release notes contain information on: new features and enhancements, migration from the immediately preceding release, known issues and limitations, and bugs fixed in the release.

NOTE: There are references in the documentation to RCB, the Rogue Wave Component Builder, a wizard-like interface that builds SourcePro components from their source form into libraries. Although the RCB books are included in the documentation set, RCB is not relevant to the SourcePro for Open Source Databases because this product contains prebuilt libraries.

Conventions

SourcePro documentation uses the following conventions.

Pathname Conventions

The SourcePro products use special pathname conventions. These are summarized in Table 3:

Table 2 – Placeholder conventions

Convention	Use	Example
`<buildspace>`	The name you gave the directory where your SourcePro components are installed.	`<buildspace>\docs\index.html`
`<ver>`	SourcePro version number	`c:\RogueWave\SourcePro\<ver>-osd`
`<lib-ver>`	library version number	
`<package-name>`	The name of a SourcePro package. Some modules have just a single package of the same name, but other modules have multiple packages.	`c:\RogueWave\SourcePro\<ver>-osd\examples\<package-name>`

Typographic Conventions

Documentation for the SourcePro products uses special typographic conventions. These conventions are summarized in Table 4:

Table 3 – Typographic conventions

Conventions	Use	Example
`courier`	Function names, code, directories, file names, examples, operating system commands, URLs.	`RWWinSockInfo info`; `cout << "Error Processing:"` Visit us at `https://www.perforce.com/products/sourcepro`.
italic	Conventional uses, such as new terms and titles.	*Functors,* or *function objects* *Essential Tools Module Reference Guide*
bold italic	Class names, emphasis, and some subheads.	***RWCollectable***
bold	Conventional uses, such as emphasis. Commands from an interface.	You should **always** read the manual. Click the **OK** button

ellipsis	Indicates part of the code is missing from an example.	`foo(){`
		`  ... //something happens`
		`}`

pathnames `/` or `\`	When the context for a pathname could be either Windows or UNIX, the path is written with a Windows "`\`" separator.	Windows path: *`buildspace`*`\sproenv_vars.bat` UNIX path: *`buildspace`*`/sproenv_vars.sh` OS-neutral path: `<`*`buildspace`*`>\docs\index.html`

Using This Manual

This manual assumes that you are familiar with the basic features of the C++ programming language. There are many widely-available books on learning C++.

Organization

This manual is divided into the following chapters:

- Chapter 1, Introduction, (this chapter) describes SourcePro for Open Source Databases.
- Chapter 2, Installing the Libraries, describes how to install SourcePro for Open Source Databases.
- Chapter 3, Building and Running the Examples, discusses how to build and run the code examples included with each SourcePro library. Studying these examples is one of the best ways to learn how to use a library effectively.
- Chapter 4, The SourcePro DB Examples, describes how to build and run the examples included with SourcePro DB. These examples require some special attention due to the complexity of establishing a database connection.
- Chapter 5, Writing Your Own Applications, describes how to get started building your own applications using SourcePro components.

Support Services

Submit support requests at your customer support portal (https://techsupport.roguewave.com/servicedesk/customer/user/login.

Chapter 2, Installing the Libraries

Overview

SourcePro for Open Sources Databases contains prebuilt versions of all the provided SourcePro libraries. This chapter describes:

- How to install the libraries
- The resulting directory structure
- Environment variables set by the installation program
- How to uninstall the libraries

For information on how to get started using the libraries, see Chapter 3.

Installing the Libraries

To download and install the SourcePro for Open Sources Databases:

1. **Download the product via the link provided in your email after purchase**.
2. **The email will also have the license.key file.** Save the file to a known location on your system. You will be asked for this location during installation.
3. **Install the product**:
   - On Windows, double-click the installer executable `sourcepro_<ver>_osd*.exe`.
   - On Linux, invoke `sourcepro_<ver>_osd*.run` at the command prompt.

   If you need a new `license.key` file, please contact license@perforce.com.

Directory Structure

The installation program places the Rogue Wave components in a *buildspace* of your choosing. For example, your buildspace might be `c:\RogueWave\SourcePro\<ver>-osd`. Files are installed in the following buildspace directory structure:

```
<buildspace>
  docs        // release and user documentation
  examples
     ...      // examples for each module and package
  lib         // prebuilt libraries (.lib, .so, and/or DLL)
  rw          // header files for installed modules
     ...      // header files for each module and package
```

Environment Variables

SourcePro for Open Sources Databases is a binary version of the SourcePro product. The installation program places the SourcePro link libraries (`.lib` on Win32, `.so` on UNIX) in the `<buildspace>\lib` directory.

To run applications using SourcePro components, your system must be able to find the dynamic libraries for all dependent libraries. To simplify your use of SourcePro for Open Sources Databases, this version uses the environment variable `RW_SOURCEPRO_HOME`, which specifies your installation location. The Windows `PATH` and UNIX `LD_LIBRARY_PATH` or `LD_LIBRARY_PATH_64` must be modified to point to the Rogue Wave and 3rd-party dynamic libraries.

Windows: Batch file `sproenv_vars.bat` defines the environment variable `RW_SOURCEPRO_HOME` and adds the SourcePro libraries to your `PATH`. Simply execute `<buildspace>/sproenv_vars.bat` at the command prompt.

UNIX: On UNIX, `sproenv_vars.sh` defines the environment variable `RW_SOURCEPRO_HOME` and adds SourcePro libraries to your `LD_LIBRARY_PATH` or `LD_LIBRARY_PATH_64`. If you are using bash or ksh (or a compatible shell), simply "dot" or source `<buildspace>/sproenv_vars.sh` as follows:

```
. <buildspace>/sproenv_vars.sh
```

NOTE: If you are using a shell compatible with csh, you will have to modify sproenv_vars.sh to accommodate csh syntax.

Uninstalling the Libraries

The installer includes an uninstall executable.

Uninstalling on Windows

On Windows you can access the uninstaller from the Start menu. The executable can also be found in the root of the installation directory.

Uninstalling on UNIX

On UNIX you must run the uninstaller from the root of the installation directory.

If you have altered your startup scripts to set `RW_SOURCEPRO_HOME` or `LD_LIBRARY_PATH`, you may wish to undo those changes.

Chapter 3, Building and Running the Examples

Overview

Each SourcePro module or package includes code examples illustrating the key features of the library. Studying these examples is one of the best ways to learn how to use a library effectively. This chapter describes how to build and run the examples.

Building and running the examples for SourcePro DB requires that you establish database connectivity. See Chapter 4 for more information on these examples. For information on how to develop your own applications using SourcePro components, see Chapter 5.

Building the Examples

The examples for each library are located in the `<buildspace>\examples\<mnemonic>` directory:

```
<buildspace>
    examples
       dbcore     // examples for DB Core Module
       mysql      // examples for the MySQL Access Module
       pgs        // examples for the PostgreSQL Access Module
       tools      // examples for Essential Tools Module
```

Building the Examples from the Command Line

Each example is built using a makefile in a `12d` or `15d` (debug) directory within the examples directory for each library. For example:

```
<buildspace>
    examples
       mysql
```

```
12d
15d
```

The provided makefiles contain the necessary include paths, link libraries, and compiler flags for each example.

Note that the Essential Tools Module's examples are categorized into the directories "manual" and "tutorial." A single makefile builds examples in both directories.

Windows

1. Open a command prompt window.
2. Ensure that the environment variables `lib` and `include` are set for the Microsoft Visual C++ compiler. If not, run `vcvarsall.bat`, usually located in the *<MSVC_installdir>*`\`*<installer_edition>*`\VC\Auxiliary\Build` directory of your compiler installation.

   You must pass the architecture argument `amd64`, for example:

   ```
   "C:\Program Files (x86)\Microsoft Visual Studio\2019\Community\VC\Auxil-
   iary\Build\vcvarsall.bat" amd64
   ```

   For more information, see the MSVC documentation.
3. Change to the `12d` subdirectory below the examples you want to build.
4. Enter `nmake`. The program looks in the current directory and executes the commands found in the `makefile`.

For instance, to build the examples for the Essential Tools Module on Windows:

```
cd <buildspace>\examples\tools\12d
nmake
```

UNIX

1. Verify that the supported compiler for your platform is working as your compiler. If not, set your environment for the supported compiler.
2. Within your SourcePro for Open Sources Databases installation, change to the `12d` directory below the examples you want to build.
3. Type `make`. The program looks in the current directory and executes the commands found in the `makefile`.

For instance, to build the examples for the Essential Tools Module on UNIX:

```
cd <buildspace>/examples/tools/12d
nmake
```

Building from Microsoft Visual Studio

Below are instructions for setting up a project in Microsoft Visual Studio to build and run the examples. You can also use this for guidance in setting up a project for your own application that depends on SourcePro libraries.

1. Open Visual Studio and select File | New | Project.
2. Select Win32 Console Application, give the project a name, and click OK.

   The Win32 Application Wizard launches. Click Next.
3. In Application Settings, select the Empty Project option and click Finish.
4. In the Solution Explorer, right-click on Source and select Add | Existing Item.
5. Navigate to and add the source code for the examples you are interested in.
6. Again, in the Solution Explorer, right click on Source and select Add | Existing Item.
7. Navigate to and add the library files the examples depend on.

   If you are uncertain what these are, look in the makefiles for the examples you want to build. For example, to build the Essential Tools Module tutorial examples, look in `<buildspace>\examples\tools\tutorial\12d\makefile`. The `REQLIBS` entry specifies all required libraries.
8. In the Solution Explorer, right click on the project and choose Properties.
9. Under VC++ Directories, edit the Executable Directories to contain the path to the DLL files.

   If the buildspace is located at `c:\RogueWave\SourcePro\<ver>-osd`, the additional executable path needed is `c:\RogueWave\SourcePro\<ver>-osd\lib`.
10. Edit the Include Directories to contain the path to the `rw` directory.

    If the buildspace is located at `c:\RogueWave\SourcePro\<ver>-osd`, the additional include path needed is `c:\RogueWave\SourcePro\<ver>-osd`.
11. Expand the C/C++ section.
12. Select Preprocessor and edit the Preprocessor Definitions to add `_RWCONFIG=12d`.
13. Select Precompiled Headers | Precompiled Header and from the dropdown list for this item, select Not Using Precompiled Headers.
14. Click OK.

Your project is now set up to compile and run the examples you have selected.

Running the Examples

To run the examples in SourcePro for Open Sources Databases, your system must be able to find the dynamic libraries (`.dll` on Win32, `.so` on UNIX). Windows users can execute the batch file `<buildspace>\sproenv_vars.bat` to set the `PATH`, while UNIX users can execute the shell script `<buildspace>/sproenv_vars.sh` to set the `LD_LIBRARY_PATH` or `LD_LIBRARY_PATH_64`, as described in Environment Variables.

Running from the Command Line

To run an example from the command line, run the executable in the `12d` subdirectory below the example source file. For instance, to run the `datetime` example for the Essential Tools Module from a command prompt, type `.\datetime` in the `<buildspace>\examples\tools\tutorial\12d` directory. This directory also contains any in put files required by the example; any generated output files are sent to the same directory.

Running from Microsoft Visual Studio

If you have set up a Microsoft Visual Studio solution for the example you want to run, you can do the following:

1. Open the appropriate solution in Visual Studio. (See Building from Microsoft Visual Studio.)
2. Right-click the example project file in the solution explorer and select **Set as Startup Project**.
3. Select **Debug | Start Without Debugging** from the main menu bar.

Input files required by the example and any generated output files are located in the **Other Files** folder.

Chapter 4, The SourcePro DB Examples

Overview

The SourcePro DB examples require special attention due to the complexity of establishing a connection to a database.

SourcePro DB provides a set of examples that illustrate how to transfer data from a relational database into a client program and back again. The examples are:

- SourcePro DB video rental store tutorials
- Bulk operation example
- Data callback example
- Cache manager example

You can also use these short, simple examples to cut-and-paste into your own database-enabled applications. See the ***DB Interface Module User's Guide*** for more information about the examples.

SourcePro Core includes two access modules. Since SourcePro DB provides a consistent API across all supported databases, both access modules use the same examples, located in the `<buildspace>\examples\dbcore` directory.

Table 4 – DB Access Modules and corresponding mnemonic

Module	Mnemonic (module)
DB Access Module for MySQL Server	`mysql`
DB Access Module for PostgreSQL Server	`pgs`

Makefiles for the examples are located in the directories named as follows:

`<buildspace>\examples\<module>\12d`

where `<module>` corresponds to your database (`mysql`, or `pgs`).

To build and run the SourcePro DB examples:

1. Ensure that your database client software has been installed and that you can connect to your database, as specified in Software Requirements.
2. Modify the tutorial configuration file `tutdefs.h` to reference your database server.
3. Build the tutorials and the `bulkcopy`, `datacb`, and `memcache` examples.
4. To run the SourcePro DB video store tutorials:
   - Run `tutinit.exe` to create all dependent database objects.
   - Run the individual tutorial programs.
   - Run `tutclean.exe` to clean up all the tables the tutorials create and use.
5. To run the `bulkcopy`, `datacb`, and `memcache` examples, run `bulkcopy.exe`, `datacb.exe`, and `memcache.exe` respectively.

The following sections describe these steps in more detail.

Modifying Files for Database Connection

The examples use include files with hard-coded values for database name, database server, database user name and database user password. You must change these values to reflect your own organizational settings.

The file `<buildspace>\examples\<package-name>\12d\tutdefs.h` contains configuration variables used by the examples. Modify this file with values for your own database connection.

Modify this file as follows:

- Do *not* change **DBSERVER_TYPE**.
- Change **DBSERVER_NAME** to the name of your database server.
- Change **DBUSER_NAME** and **DBPASSWORD** to a valid user name and password for your database; the user account must have permission to create tables on the database in order to run the tutorials.
- If your database requires it, specify the database name using **DBDATABASE_NAME**.
- If your database requires it, specify the property string using **DBDATABASE_PSTRING**. Please refer to the Access Module Reference Guide of the access module you are using for the contents of the property string.

For example, your configuration file might look like this:

```
#define DBSERVER_TYPE "msq<lib-ver>12d.dll"
#define DBSERVER_NAME "DataSource"
#define DBUSER_NAME "username"
#define DBPASSWORD "password"
#define DBDATABASE_NAME "database"
#define DBDATABASE_PSTRING ""
```

NOTE: These values are compiled into the tutorials when you build them. If you need to change the values at a later time, you must clean the example object files with an `nmake clean`, then rebuild the tutorials. (See Building the DB Examples.)

Building the DB Examples

Building from the Command Line

On Windows

To build the examples for a given module `<module-name>`, open a command prompt window set with the Visual Studio environment, as described in Building the Examples from the Command Line, and type `nmake` in the `<buildspace>\examples\<package-name>\12d` directory.

On UNIX

To build the examples for a given module `<module-name>`, type `make` in the `<buildspace>/examples/<package-name>/12d` directory.

Running the DB Examples

This section describes running the video store tutorials in detail, including the purpose of each tutorial, its inputs and outputs. For the `bulkcopy` and `datacb` examples, simply follow the instructions in Running the Examples.

Populate Your Database

From the command line, run `tutinit.exe` to create all dependent database objects for the tutorials. File `setuperr.txt` reports any errors that occur during setup.

Run the Tutorials

To run the tutorials, enter the executable name in a command prompt window.

Table 6 contains a list of the video store tutorials, their related files, and a description of their content. Filenames containing the word `out` give the output of the tutorial program. Filenames containing the word `err` contain error messages.

Input files required by the tutorials and generated output files are placed in the `<buildspace>\examples\<package-name>\12d` directory.

Table 5 – The SourcePro DB video store tutorials

Tutorial Name	Related Files	Tutorial Description

Retrieving Data from a Table	`t1.exe`
	`t1out.txt`
	`t1err.txt`
		Demonstrates how to retrieve data from a table in a database and how to use an arbitrary struct or class

Retrieving Data Selectively	`t2.exe`
	`t2out.txt`
	`t2err.txt`
		Demonstrates selective retrieval of data from a table in a database

Inserting Data into a Table	`t3.exe`
	`t3out.txt`
	`t3err.txt`
		Demonstrates how to insert data into a table in a database

Inserting and Updating Data	`t4.exe`
	`t4out.txt`
	`t4err.txt`
		Demonstrates how to insert and update data in a database

Deleting Rows from a Table	`t5.exe`
	`t5out.txt`
	`t5err.txt`
		Demonstrates how to delete rows from a table in a database

A Complicated Query	`t6.exe`
	`t6out.txt`
	`t6err.txt`
		Demonstrates a complicated query with the entire result set brought directly into memory. Uses a template-based collection from the Essential Tools Module to store query data

Updating Data in Multiple Tables	`t7.exe`
	`t7out.txt`
	`t7err.txt`
		Demonstrates how to enter new data and update existing data in multiple tables within a database

In-Memory Caching	`memcache.exe`
	`memcacheout.txt`
	`memcacheerr.txt`
		Demonstrates how to create and install an in-memory cache and the performance benefit of using one

Clean Up Your Database

From the command line, run `tutclean.exe` to clean up all the tables the tutorials create and use. File `tutclean.txt` reports errors that occur during cleanup.

Chapter 5, Writing Your Own Applications

Overview

Once you have succeeded in building and running the examples, you can start developing your own applications with SourcePro components. This chapter describes:

- How to use SourcePro components in your applications by including the appropriate header files
- How to build your applications, including how to specify proper include paths, linked libraries, and compiler flags
- How to run your applications

Using SourcePro Components

To use a Rogue Wave class, simply include its header file in your application. SourcePro header files are organized in the buildspace as shown below:

```
buildspace
   ...
   rw              // global Essential Tools Module header files
       db          // DB Core header files
         mysrc     // MySQL header files
         pgssrc    // PostgreSQL header files
       tools       // Essential Tools Module header files
```

The `#include` directives for all SourcePro components must contain the path to each header file from the `rw` directory. For example:

```
#include <rw/cstring.h>
#include <rw/db/table.h>
```

Building Your Applications

When you build your applications, you may find it useful to copy and modify the makefiles used to build the SourcePro examples. Another way of displaying command lines is to build an example with `nmake` (Win32) or `make` (UNIX), and capture the command line from the console.

Building in Release or Debug Mode

SourcePro for Open Sources Databases provides both release and debug binaries. The debug versions are powerful tools for uncovering and correcting internal errors in your code. To take advantage of this capability, build your applications in debug mode by:

- Defining the buildtype as debug. See Defining the _RWCONFIG Macro.
- Setting the debug compiler flag. See Specifying Compiler Flags and System Libraries.
- Linking in the debug versions of the libraries. See Linking to the Libraries.

About Compiling and Linking

On Windows, a compiler command line to build an application should contain these elements:

```
<compiler_invocation> -D_RWCONFIG=<buildtype> <include_paths>
  <system-flags-and-macros> -c <cpp-file-name>
```

A link line should contain these elements:

```
<compiler_invocation> /Fe <executable_name> <object_file_name>
  <import_libraries> /link /LINKPATH <import_library_paths> -logo
```

For example, to compile and link a dynamic (DLL) release application on Windows that depends on the Essential Tools Module and MySQL:

```
cl -D_RWCONFIG=12d -Ic:\RogueWave\SourcePro\<ver>-osd
-nologo -EHsc -MD -W3 -O2 -arch:SSE2 -GR -D_CRT_SECURE_NO_DEPRECATE
-D_SCL_SECURE_NO_DEPRECATE -D_CRT_NONSTDC_NO_DEPRECATE -c myapp.cpp

cl /Fe myapp.exe myapp.obj dbt12d.lib tls12d.lib user32.lib
/link /LIBPATH c:\RogueWave\SourcePro\<ver>-osd\lib -nologo
```

where `<ver>` means the current SourcePro version number.

Below are the equivalent lines for Unix. Unix specifies link paths with `-L` and import libraries with `-l`, and the executable name that follows `-o` has no extension.

```
g++ -D_RWCONFIG=12d -I/usr/local/RogueWave/SourcePro/<ver>-osd -m64 -pthread  \
-std=gnu++17 -I. -c myapp.cpp

g++ -m64 -pthread -L/usr/local/RogueWave/SourcePro/<ver>-osd/lib  \
-o myapp myapp.o -lm -ldl -ldbt<lib-ver>12d -ltls<lib-ver>12d
```

where `<ver>` means the current SourcePro version number and `<lib-ver>` means the library version number, which differs from the SourcePro version number.

Setting the Include Paths

For all SourcePro components, the include path must point to the buildspace root. Do not point to the `rw` directory because the `#include` directive in SourcePro code already includes this directory. For example:

```
#include <rw/rstream.h>
```

Thus, if the buildspace is located at `c:\RogueWave\SourcePro\<ver>-osd`, the command line include path is `-Ic:\RogueWave\SourcePro\<ver>-osd`.

Defining the _RWCONFIG Macro

The `_RWCONFIG=<buildtype>` command line macro defines for you all the preprocessor macros required by a particular build type of SourcePro. The prebuilt SourcePro libraries in SourcePro for Open Sources Databases are all shared libraries, also called dynamically linked libraries (DLLs) on Windows.

On Windows, the shared libraries are built with multithreading support and using the Microsoft Visual C++ or Microsoft Visual Studio .NET implementation of the C++ Standard Library. For this build type, set the command line macro as `-D_RWCONFIG=12d` for a release build, or `-D_RWCONFIG=15d` for a debug build.

On UNIX, the shared libraries are built with multithreading support and using the compiler's implementation of the C++ Standard Library. Set the following command line macro for this build type `-D_RWCONFIG=12d` for a release build, and `-D_RWCONFIG=15d` for a debug build.

Specifying Compiler Flags and System Libraries

You must use the same compiler flags to build your application that were used to build the SourcePro libraries you are linking to. The macro `_RWCONFIG=12d` (release build) or `_RWCONFIG=15d` (debug build) handles all SourcePro-specific command line requirements, but not system-specific requirements.

To determine the system requirements, consider SourcePro example makefiles. For instance, since the MySQL, PostgreSQL, and the DB packages depend on the Essential Tools Module, the makefiles will include the requirements for all libraries.

Here are a few lines from an example makefile, including the critical line with the `COMPILEFLAGS` variable:

```
#C++ macros
CPPINVOKE= clCOMPILEFLAGS= -nologo -EHsc /std:c++17 -MD -O2 -GR -D_SCL_SECURE_NO_-
DEPRECATE -I.
CONFIGDEF=-D_RWCONFIG=$(TAG)
LINKLIBS= dbt12d.lib tls12d.lib user32.lib
```

The `COMPILEFLAGS` variable tells you the flags you need.

Linking to the Libraries

For the link libraries, use the path to the SourcePro libraries and DB access modules. Both the SourcePro libraries and the database client files are located in the `lib` subdirectory of the buildspace; that is, if the Rogue Wave buildspace is `c:\RogueWave\SourcePro\<ver>-osd`, then the libraries are in `c:\RogueWave\SourcePro\<ver>-osd\lib`.

Rogue Wave gives a specific name to SourcePro for Open Sources Databases link libraries. For example, if you link the Essential Tools Module and the MySQL access module, your command line would include the link line that pertains to your platform as shown in Table 6.

Table 6 – Link libraries

Platform	Release Link Libraries Command Line	Debug Link Libraries Command Line
Win32	`/LIBPATH: <buildspace>\lib` `dbt12d.lib tls12d.lib`	`/LIBPATH: <buildspace>\lib` `dbt15d.lib tls15d.lib`
UNIX	`-L<buildspace>/lib` `-ldbt<libver>12d -ltls<libver>12d`	`-L<buildspace>/lib` `-ldbt<libver>15d -ltls<libver>15d`

Additional Parameters

Your application may, of course, require additional include paths, system flags and macros, and link libraries. Note that any additional link libraries should be built in the same configuration as the SourcePro libraries, and preferably with the same system flags and macros. If you specify system flags and macros not used in building the SourcePro libraries, there could be conflicts.

Running Your Applications

To run your application, your system must be able to find the DLL or shared library versions of any dependent libraries.

- Windows users may simply execute the `<buildspace>\sproenv_vars.bat` batch file to set the Windows `PATH`.
- UNIX users may simply source the `<buildspace>/sproenv_vars.sh` script to set the UNIX `LD_LIBRARY_PATH` or `LD_LIBRARY_PATH_64`.

For more information about setting your environment on Windows or UNIX, see Environment Variables.

Index