Borland C++ 4.0 Example Programs
Listed by File Getting Started

Filenames:

ACLOCK
APPLAUNC
BMPVIEW
BUTTON
CALC
CHELP
CMDLG
COLORDLG
COMBOBOX
COMMDLG
CURSOR
DDEML
DDESRVR
DELIVER
DIAGXPRT
DLLDEMO
DLLHELLO
DOCVIEW
DOSEXAMP
DRAGDROP -- OWL
DRAGDROP -- WINDOWS
DRAW
EDIT
EDITSEAR
FFIND
FILEBROW
FILTER
GAUGE
GDIDEMO
GROUPBOX
HDUMP
HELLOAPP
HELP
HELPEX
INSTANCE
INTLDEMO
LABELS
LAYOUT
LISTBOX
LOOKUP
MAKEALL
MCISOUND
MDI
MDIFILE
MDISTRM
MTHREAD
MULTITRG
NOTIFY
OLDFILEW
OWLBWCC
OWL TUTORIAL

OWLCMD
OWLSCRN
OWNERDRA
PAINT
PALETTE
PEEPER
POPUP
PRINTING
PRNTPREV
PROGMANX
PSTREAM
QUEUETST
REVERSE
SCROLLBA
SCROLLER
SDIFILE
SLIDER
SOUNDER
SRCSPOOL
STATIC
STEPS
STRNGMAX
STYLESHT
SWAT
SYSINFO
TDWDEMO
TESTDIR
TODO
TRANSFER
TRUETYPE -- OWL
TRUETYPE -- WINDOWS
TSTAPP
VALIDATE
VBDIALOG
VBXCTL
WHELLO -- WINDOWS
XREF

Borland C++ 4.0 Example Programs
Getting Started -- How to Run the Samples File List

Running the samples within the IDE:

Use Project / Open to open the project file (*.IDE) associated with the program(s) you want to run. Use Compile / Build All to build the project and Debug / Run to see the program output. See the Borland C++ Users Guide for more information on using the IDE.

Running the samples from the command line:

Go to the directory that contains the example(s) you want to run and type "MAKE". When make has completed building your program, go to the Windows Program Manager, select "RUN", and enter the path and filename of your program.

Problems? Questions?

If you have questions about running the sample programs and cannot find your answer in the documentation and on-line help, contact Borland technical support at (408) 461-9133.

Project Filename: PSTREAM.IDE
Project Directory: EXAMPLES\CLASSLIB\PSTREAM

This project creates three executables, stream0.exe, stream1.exe, and stream2.exe, with the latter two being extensions of the functionality found in the first. Together they illustrate how to work with streamable objects, i.e., objects that can be stored and retrieved from a stream such as a disk file, etc.

Project Filename: TODO.IDE
Project Directory: EXAMPLES\CLASSLIB\TODO

This program implements a simple Todo list. It uses streamable objects for the things-to-do entries and implements a sorted array for storing the entries. Also uses common file dialog boxes.

Project Filename: BUTTON.IDE
Project Directory: EXAMPLES\OWL\OWLAPI\BUTTON

(Note: project file creates executable called buttonx.)

This project shows how to use the OWL interface elements for various kinds of Windows buttons. Standard push style buttons are demonstrated as well as radio buttons and check boxes.

Project Filename: COLORDLG.IDE
Project Directory: EXAMPLES\OWL\OWLAPI\COLORDLG

This example implements a custom control (TColorControl) for displaying and selecting a color. It also shows how to use this custom control inside a dialog box (TColorDialog).

Project Filename: COMBOBOX.IDE
Project Directory: EXAMPLES\OWL\OWLAPI\COMBOBOX

(Note: Project produces an executable called COMBOBXX.EXE).

This comprehensive example shows how to use, manipulate, and query combo-boxes based on the OWL interface class, TComboBox. It illustrates simple, drop down, and drop down list comboxes.

Project Filename: COMMDLG.IDE
Project Directory: EXAMPLES\OWL\OWLAPI\COMMDLG

This program has example code that uses the Common Dialog classes in Owl.

The main window will have menu selections for opening a file, changing the font and changing the color used for the selected font. When a file is selected the name will be displayed on the client area of the window.

For an example of using Common Dialogs in C, see: CMDLG.

Project Filenames: DLLHELLO.IDE
Project Directory: EXAMPLES\OWL\OWLAPI\DLLHELLO

This program is called DLLHELLO because it is a simple example showing you how to create and use DLLs with your programs. It will show you a number of techniques, including how to:

Call into a DLL to create a window.
Load a String resource from a DLL.
Load a Cursor resource from a DLL.
Load an Icon resource from a DLL.
Load a Bitmap resource from a DLL.

Project Filename: DOCVIEW.IDE
Project Directory: EXAMPLES\OWL\OWLAPI\DOCVIEW

This OWL example shows how an application can use documents and views by creating objects derived from TDocument and TView, respectively. The docviewx.exe it produces is a file viewer and editor which can use different Views to handle different types of files (text files, binary files, source code, etc.).

Project Filename: EDIT.IDE
Project Directory: EXAMPLES\OWL\OWLAPI\EDIT

This program shows how to derive a custom edit control from the TEdit class and use such a control in your application.

Project Filename: EDITSEAR.IDE
Project Directory: EXAMPLES\OWL\OWLAPI\EDITSEAR

TEditSearch is a streamable edit control that responds to Find, Replace, and FindNext menu commands. This project builds an application which uses a TEditSearch control as the applications main window.

Project Filename: GAUGE.IDE
Project Directory: EXAMPLES\OWL\OWLAPI\GAUGE

This program shows how to use a gauge control (TGauge) and a slider control (TSlider) in your applications.

See also: SLIDER

Project Filename: GROUPBOX.IDE
Project Directory: EXAMPLES\OWL\OWLAPI\GROUPBOX

Example of a Group Box control , i.e., a collection of radio buttons offering the user a choice between mutually exclusive selections. Uses OWLs TGroupBox class. See also LISTBOX

Project Filename: INSTANCE.IDE
Project Directory: EXAMPLES\OWL\OWLAPI\INSTANCE

Simple program showing how Windows keeps track of a programs Instance Handle, i.e., a handle indicating whether this is the first copy of the application to be opened or not. To see how this works, build this project and then run several copies of instance.exe at once.

Project Filename: LAYOUT.IDE
Project Directory: EXAMPLES\OWL\OWLAPI\LAYOUT

This is an interactive program to demonstrate a TLayoutWindow. It creates a frame window, colored child windows and a client window.

The file LAYOUT.CPP has some initial comments which describe this project in more detail.

Project Filename: LISTBOX.IDE
Project Directory: EXAMPLES\OWL\OWLAPI\LISTBOX

This example illustrates the class TListBox. It shows the different types of listboxes you can create and the methods used to edit, query, and retrieve strings and user input from these controls.

See also: GROUPBOX.

Project Filename: MDI.IDE
Project Directory: EXAMPLES\OWL\OWLAPI\MDI

This is a minimal MDI application showing the relationship between the MDI client window and its children.

For an application showing how this relationship is used in the context of a text editor, see MDIFILE. To see how to save the state of the MDI parent and child windows (i.e., how to save the desktop), see MDISTRM.

Project Filename: NOTIFY.IDE
Project Directory: EXAMPLES\OWL\OWLAPI\NOTIFY

A notification tester program which shows how a simple set of child controls (buttons, in this case), notifies its parent windows of events.

Project Filename: OWNERDRA.IDE
Project Directory: EXAMPLES\OWL\OWLAPI\OWNERDRA

Shows how owner draw buttons -- i.e., buttons whose appearance are controlled by the programmer rather than by Windows -- are created.

Project Filename: PALETTE.IDE
Project Directory: EXAMPLES\OWL\OWLAPI\PALETTE

Shows how to test how colors are mapped to a device, and how to display the color palette.

To see how to use a common dialog box to have users select a color from a palette, see COLORDLG.

Project Filename: POPUP.IDE
Project Directory: EXAMPLES\OWL\OWLAPI\POPUP

This program illustrates the differences among child windows, popup windows with parents, and popup windows without parents in terms of how they behave and are created.

Project Filename: PRINTING.IDE
Project Directory: EXAMPLES\OWL\OWLAPI\PRINTING

This printing program shows how to use a class derived from TPrintout to send information to the printer.

Project Filename: PRNTPREV.IDE
Project Directory: EXAMPLES\OWL\OWLAPI\PRNTPREV

This example shows one way to implement a window showing a print preview.

Project Filename: SCROLLBA.IDE
Project Directory: EXAMPLES\OWL\OWLAPI\SCROLLBA

This program shows how to use a TScrollBar control as a child window control. It uses a scrollbar to enter a temperature value, which the parent window displays.

Project Filename: SCROLLER.IDE
Project Directory: EXAMPLES\OWL\OWLAPI\SCROLLER

This example shows how to derive a class TScrollWindow from TFrameWindow, with horizontal and vertical scrollbars.

Project Filename: SLIDER.IDE
Project Directory: EXAMPLES\OWL\OWLAPI\SLIDER

This program shows how to use a gauge control (TGauge) and a slider control (TSlider) in your applications.

See also: GAUGE.

Project Filename: STATIC.IDE
Project Directory: EXAMPLES\OWL\OWLAPI\STATIC

This example shows the various styles associated with static controls, and what these styles will do.

Project Filename: TRANSFER.IDE
Project Directory: EXAMPLES\OWL\OWLAPI\TRANSFER

This example shows how to use the OWL Transfer mechanism to save and restore information which a user enters into a dialog box.

Project Filename: VALIDATE.IDE
Project Directory: EXAMPLES\OWL\OWLAPI\VALIDATE

This example shows how to use the OWL validator objects to validate user input (i.e., to make sure user input conforms to a given pattern, such as limiting input to either Manufacturing or Sales, or using a picture field like (###) ### - #### for a phone number).

Project Filename: VBXCTL.IDE
Project Directory: EXAMPLES\OWL\OWLAPI\VBXCTL

This sample shows how to use VBX controls in your applications. By using VBX controls, you can take advantage of a host of different kinds of controls provided by Borland and by third-party vendors -- spinners, gauges, spreadsheet-like controls, etc.

This program demonstrates a light-switch like control, a picture control which implements drag and drop, and a VBX implementation of a gauge control (See also GAUGE for an alternate way to implement the gauge control).

This example is also available in non-OWL form; see VBDIALOG.

Project Filename: ACLOCK.IDE
Project Directory: EXAMPLES\OWL\OWLAPPS\ACLOCK

A cuckoo clock program which provides an Animate class to sequentially display bitmaps. Also demonstrates the use of windows timers.

Project Filename: BMPVIEW.IDE
Project Directory: EXAMPLES\OWL\OWLAPPS\BMPVIEW

This example program shows DIBs, Bitmaps and Palettes in a scrolling Window. Also uses diagnostics to trace through some routines. Creates a class TBmpViewWindow, a Bitmap displaying window derived from TClipboardViewer. Shows how bitmaps can be retrieved from disk or used as resources internal to an executable program

Project Filename: CALC.IDE
Project Directory: EXAMPLES\OWL\OWLAPPS\CALC

This calculator program demonstrates how to use a dialog box as the main window for your application, and how to set up an accelerator table.

Project Filename: CURSOR.IDE
Project Directory: EXAMPLES\OWL\OWLAPPS\CURSOR

Program displays the position of the cursor (i.e., mouse coordinates) relative to the entire video display and to a given window. Also displays information about the handle to the window below the cursor.

Project Filename: DRAW.IDE
Project Directory: EXAMPLES\OWL\OWLAPPS\DRAW

A simple program demonstrating how to use the mouse to draw lines; right button clears the image drawn. This is a good program to look at if you need to learn how to trap mouse events using OWL.

Project Filename: FILEBROW.IDE
Project Directory: EXAMPLES\OWL\OWLAPPS\FILEBROW

This sample implements a file browsing window which browses through files and installed drives on your computer. Text files are then displayed in the lower portion of the window.

Project Filename: GDIDEMO.IDE
Project Directory: EXAMPLES\OWL\OWLAPPS\GDIDEMO

This program shows four mini-examples of how to use the ObjectWindows Library classes which encapsulate the functionality of the Windows Graphics Device Interface (GDI). Some of the techniques illustrated include working with fonts, displaying and animating bitmaps, and line drawing.

Project Filename: HELLOAPP.IDE
Project Directory: EXAMPLES\OWL\OWLAPPS\HELLO

This is the ObjectWindows Library Hello World program -- this program simply shows how to set up and run a minimal Windows application using the ObjectWindows Library.

Project Filename: MDIFILE.IDE
Project Directory: EXAMPLES\OWL\OWLAPPS\MDIFILE

This OWL example is a text editing program based on the Windows Multiple Document Interface, or MDI. As such, it allows multiple client windows to be opened and manipulated at the same time. This example also shows how to implement a toolbar and status bar. See also SDIFILE, MDI

.

Project Filename: MTHREAD.IDE
Project Directory: EXAMPLES\OWL\OWLAPPS\MTHREAD

This example is an OWL example of multithreading.

Windows NT is required to run this example.

Project Filename: OWLCMD.IDE
Project Directory: EXAMPLES\OWL\OWLAPPS\OWLCMD

This program demonstrates the use of the TTinyCaption mix-in. It uses a class derived from TComboBox that catches the <RETURN> key in an edit field and executes the command which was typed. If the command executed successfully it is stored in the list box portion of the combo box for future retrieval.

Commands supported:

DOS	Executes COMMAND.COM
DIR	Launches WINFILE
CD x$	Changes default directory (CD\ CD \)
EXIT	Closes OWLCMD
x$:	Changes default drive (D:)
x$	Executes command x$

Project Filename: PAINT.IDE
Project Directory: EXAMPLES\OWL\OWLAPPS\PAINT

This SDI program shows how to implement a bitmap-drawing program like Paintbrush using ObjectWindows. It implements line drawing, color fill, and the ability to draw objects such as ellipses and rectangles.

Project Filename: PEEPER.IDE
Project Directory: EXAMPLES\OWL\OWLAPPS\PEEPER

An OWL program demonstrating how messages are sent between one window and another, and how Windows captions are set.

Project Filename: OWLSCRN.IDE
Project Directory: EXAMPLES\OWL\OWLAPPS\SCRNSAVE

This project shows how to create your own screen savers for use with Windows 3.1.

Project Filename: SDIFILE.IDE
Project Directory: EXAMPLES\OWL\OWLAPPS\SDIFILE

This is an SDI (Single Document Interface) editor, similar to Notepad. One file can be opened at a time. For an example of an MDI (Multiple Document Interface) editor, see MDIFILE.

Project Filename: SWAT.IDE
Project Directory: EXAMPLES\OWL\OWLAPPS\SWAT

A ObjectWindows debugging game. Kill the bugs!

Project Filename: DDEML.IDE
Project Directory: EXAMPLES\OWL\WINAPI\DDEML

This is a sample application using OWL that demonstrates the use of the Windows 3.1 DDEML API in a client application. You should first build the server application, DDESVR.EXE, and run it. Then run the client application, DDECLI.EXE, to start a conversation. Note that because the IDE only allows you to run one target at a time, you will want to run one or more of these applications from Program Manager or file Manager. Detailed information on DDEML can found in the on-line help and is suggested reading for anyone interested in writing DDEML applications. Search on the keyword DDEML.

Project Filename: DRAGDROP.IDE
Project Directory: EXAMPLES\OWL\WINAPI\DRAGDROP

This OWL example shows how to drag and drop files from within the Windows file manager. For a Windows API version, see DRAGDROP -- WINDOWS.

Note: To run this demo successfully, you will want to make sure that the Dragdrop examples window and the File Manager Window are both visible, side by side. Otherwise clicking within file manager may cause the dragdrop example window to disappear.

Project Filename: HELP.IDE
Project Directory: EXAMPLES\OWL\WINAPI\HELP

This example shows how to implement context sensitive help within a Windows program created with OWL. See also: CHELP and HELPEX.

Project Filename: MCISOUND.IDE
Project Directory: EXAMPLES\OWL\WINAPI\MCISOUND

This example demonstrates the use of MCI APIs in Windows 3.1 in an OWL application. It plays sound files (*.WAV).

You must have a sound board/speaker and its device driver installed under Windows 3.1 and be certain that it works -- Use the sound applet in Windows Control Panel to generate a sound. You may copy one of the .WAV files from the WINDOWS subdirectory in your system to the EXAMPLES\OWL\WINAPI\MCISOUND directory.

Project Filename: PROGMAN.IDE
Project Directory: EXAMPLES\OWL\WINAPI\PROGMAN

This program uses a DDE conversation with Program Manager to create a program group and icons based on a users input.

Project Filename: SYSINFO.IDE
Project Directory: EXAMPLES\OWL\WINAPI\SYSINFO

This Windows program displays information about your system such as DOS version, Windows version, CPU type, etc. in a Window.

Project Filename: TRUETYPE.IDE
Project Directory: EXAMPLES\OWL\WINAPI\TRUETYPE

The True Type example is provided in two forms, this OWL form and a traditional, C/SDK version, TRUETYPE -- WINDOWS

Both programs show off two features of Windows 3.1, True Type fonts and common dialogs.

True Type fonts are scalable fonts supplied by the system. The True Type programs demonstrate resizing and rotation of True Type characters within a window, and allow the user to select the True Type font to be used.

The dialog used in selecting the font is a Windows 3.1 "common dialog." These are available to Windows 3.1 OWL programs in the form of objects derived from TCommonDialog (TChooseFont is the one used here). These common dialogs are also available to applications programmed in C, as shown in the C/SDK version.

Project Filename: MAKEALL.IDE
Project Directory: EXAMPLES\WINDOWS

This project file builds all the examples that are located in the subdirectories under EXAMPLES\WINDOWS. To build all of these examples, go to EXAMPLES\WINDOWS and type MAKE.

Project Filename: CHELP.IDE
Project Directory: EXAMPLES\WINDOWS\CHELP

This example shows how to implement context sensitive help within a Windows program created in C.
See also: HELP and HELPEX.

Project Filename: CMDLG.IDE
Project Directory: EXAMPLES\WINDOWS\CMDLG

This program demonstrates the use of the Windows 3.1 common dialog boxes.

The main window will have menu selections for opening a file, changing the font and changing the color used for the selected font. When a file is selected the name will be displayed on the client area of the window.

For an OWL example, see COMMDLG.

Project Filename: DDESRVR.IDE
Project Directory: EXAMPLES\WINDOWS\DDEML

This project builds two files, a DDE Server executable and a DDE Client Executable.

To run the DDE client/server example, launch both ddeclnt.exe and ddesrvr.exe. At this point, all user interaction takes place in the client window. Establish a connection (you can verify the connection in the server window). Send data to and request data from the server. Disconnect from the server when you are finished.

Note that because the IDE only allows you to run one target at a time, youll want to run one or more of these applications from Program Manager or file Manager.

Project Filename: DELIVER.IDE
Project Directory: EXAMPLES\WINDOWS\DELIVER

Deliver is a tool for copying or moving files and creating directories.

Deliver recognizes the commands COPY, MOVE, and MKDIR. It is most useful when attached to a SourcePool node.

For more information about Deliver, see the file Deliver.txt in the projects directory.

Project Filename: DLLDEMO.IDE
Project Directory: EXAMPLES\WINDOWS\DLLDEMO

An example of a program using a DLL, written in C++.

Project Filename: DRAGDROP.IDE
Project Directory: EXAMPLES\WINDOWS\DRAGDROP

This example shows how to drag and drop files from within the Windows file manager. For an OWL version, see DRAGDROP -- OWL .

Note: To run this demo successfully, you will want to make sure that the Dragdrop examples window and the File Manager Window are both visible, side by side. Otherwise clicking within file manager may cause the dragdrop example window to disappear.

Project Filename: FFIND.IDE
Project Directory: EXAMPLES\WINDOWS\FFIND

This project builds a Windows program to find files on system drives.

Project Filename: HDUMP.IDE
Project Directory: EXAMPLES\WINDOWS\HDUMP

This Windows program shows how to do a hex dump of a file in Windows.

Project Filename: HELPEX.IDE
Project Directory: EXAMPLES\WINDOWS\HELPEX

This example shows how to integrate on-line help in your application; see also HELP and CHELP.

Project Filename: INTLDEMO.IDE
Project Directory: EXAMPLES\WINDOWS\INTLDEMO

This is a program which demonstrates much of the functionality afforded with the setlocale function and the locale libraries. It also demonstrates programming a user interface in such a way that it will be language-independent and allow switching of the language at run-time.

Project Filename: SOUNDER.IDE
Project Directory: EXAMPLES\WINDOWS\SOUNDER

This C Windows demonstrates sound APIs in Win 3.1. A sound board / speaker combination is required to run this program.

Project Filename: TDWDEMO.IDE
Project Directory: EXAMPLES\WINDOWS\TDW

This project file builds S_PAINT.EXE, a Windows program used as a Turbo Debugger for Windows demonstration file. See the Borland C++ Users Guide for more information.

Project Filename: TRUETYPE.IDE
Project Directory: EXAMPLES\WINDOWS\TRUETYPE

The True Type example is provided in two forms, this form and an ObjectWindows version, TRUETYPE -- OWL

Both programs show off two features of Windows 3.1: True Type fonts, and common dialogs.

True Type fonts are scalable fonts supplied by the system. The True Type programs demonstrate resizing and rotation of True Type characters within a window, and allows the user to select the True Type font to be used.

The dialog used in selecting the font is a Windows 3.1 "common dialog." These are available to Windows 3.1 applications programmed in C, as shown here, and also in OWL form as TChooseFontDialog (used in this program), TFileOpenDialog, etc.

Project Filename: TSTAPP.IDE
Project Directory: EXAMPLES\WINDOWS\TSTAPP

This project file builds two applications, a DDE client and a DDE server, which demonstrate the use of DDE for sending messages between applications.

Project Filename: VBDIALOG.IDE
Project Directory: EXAMPLES\WINDOWS\VBDIALOG

This sample shows how to use VBX controls in your applications. By using VBX controls, you can take advantage of a host of different kinds of controls provided by Borland and by third-party vendors -- spinners, gauges, spreadsheet-like controls, etc.

This program demonstrates a light-switch like control, a picture control which implements drag and drop, and a VBX implementation of a gauge control (See also GAUGE for an alternate way to implement the gauge control).

This example is also available in OWL form; see VBXCTL.

Project Filename: STEP*.EXE
Project Directory: EXAMPLES\OWL\TUTORIAL

The sample programs in the EXAMPLES\OWL\TUTORIAL directory are designed to be used in conjunction with the tutorial in the ObjectWindows Programmers Guide.

If you would like to build all of the executable files in one step, use the STEPS.IDE project file or the MAKEFILE located in the EXAMPLES\OWL\TUTORIAL directory. For more information, see the instructions on building the sample programs in Getting Started.

Project Filename: WHELLO.IDE

Project Directory: EXAMPLES\WINDOWS\WHELLO

This project shows how you would write a simple Hello World program that uses C++ classes but does not use the ObjectWindows library.

Project Filename: MULTITRG.IDE
Project Directory: EXAMPLES\IDE\MULTITARG\MULTITRG

The following is the text of the on-line document MULTITARG.TXT, located in this projects directory:

This project is an example of multi-targeting in the new project manager.

The two whello.exe targets in the project are mostly similar with two major differences: The local options for the first target specifies output directories to point to .\out16 and is tagged as Win3.x (16 bit) application. The second target has local options specifying output directories to point to .\out32 and is tagged as a Win32 application.

The targets were created by
1) selecting Project|New Target,
2) typing "whello" in the 'Target Name' field,
3) selecting 'Standard' from the 'Target Type' list,
4) selecting 'OK'

This leads the standard 'Add Target' dialog where:
5) selecting "Windows 3.x (16 bit)" from the 'Platforms' list
6) selecting 'OK'

Repeating steps 1-6 for the second target with the exception of step (5) where "Win32" was selected from the 'Platforms' list.

To redirect output on the first target to the 'out16' directory
6) Right click on the first target
7) Select 'Local Options' with brings up the Multi-Page-Dialog
8) In the 'Directories' section entering 'out16' in the 'Intermediate' and 'Final output' fields.
9) Select 'OK'

Repeating steps 6-9 for the second target replacing 'out16' with 'out32' completes the process.

You should note that the source nodes under both executables are copied under each target. If you wanted to add a source module you would have to do so (in this project) in two places to keep you 16bit and 32bit targets in synch. To get relief see the example in the SrcPool directory.

Project Filename: SRCPOOL.IDE
Project Directory: EXAMPLES\IDE\SRCPOOL\SRCPOOL

The following is the text of the on-line document SRCPOOL.TXT, located in this projects directory:

This project demonstrates the use of a simple source pool target and
'reference copying' within the project.

Source pools are abstract container objects that hold dependencies.
There are not buildable and runnable by themselves. However, when moved,
copied or (most useful:) reference copied to under 'real' targets, they
take on the options and target attributes of the context they are in.
When the project make facility is checking dependencies or building
response table for linkers and librarians, the SourcePool becomes
invisible and the node is seen from the target as a direct dependency.

Reference copying allows one node (and all of it's dependencies) to be
referenced in many different places in the project's dependency tree.
In this example we reference copy a SourcePool node to under two
executables (a 16bit target and a 32bit target, both called whello.exe).
This allows you to modify the contents of the SourcePool, adding or
deleting nodes or changing the names of node or their
node-types, and having the change automatically update all of the
references to the SourcePool.

NOTE: The example here assumes that you have looked at the example in the
'MultiTrg' directory and are familiar with how to create several
targets within the project and set local options on the target nodes.

Here are steps that were taken to create this project:
1) Select Project|New Target from the main menu
2) Typing "My whello source" in 'Target name'
3) Selecting 'SourcePool' from the 'Target type' list
4) Hitting 'OK'

This creates a node called 'Source Pool'.
5) Selecting that node in the Project Window
6) Hitting the Insert key to bring up the 'Add Item' dialog
7) Selecting whello.cpp, whello.rc and whello.def
8) Hitting 'OK'

This creates three dependencies under the SourcePool node.

Now create the two targets the same way as in the directory of the MultiTrg
example and delete the .cpp, .rc. and .def nodes from under both targets
by selecting all of them (Ctrl-LeftButton will select additively) and
hitting the Delete key. (Do not forget to modify the Intermediate and
Final output fields of the two WHELLO.EXEs.)

While holding down the Alt key, drag the SourcePool under the first
whello [.exe] node and release the mouse. Do the same for second whello
[.exe] node.

More notes on SourcePools

More advanced uses of SourcePools include nesting them, which allows

you to logically group source files without changing there location on disk. All nested SourcePools are 'flattened' during target dependency checking and creation time.

More notes on Reference Copies

Setting options on a reference copied node *does not* affect options of any other reference copy of original of the node. To see this in effect, see the example in the 'StyleSht' directory.

Project Filename: STYLESHT.IDE
Project Directory: EXAMPLES\IDE\STYLESHT\STYLESHT

The following is the text of the on-line document STYLESHT.TXT, located in this projects directory:

Note:
[This example builds on another example in the 'SrcPool' directory that walks through how build a multi-target project using abstract SourcePools and reference-copies. A general understanding of that example is assumed here. Although SourcePools and reference copying are used in this example, they are by no means necessary for using StyleSheets.]

This example shows how applying StyleSheets can dramatically ease the process of creating multiple targets based on the same source code with completely different results: one target is designed for debugging and pre-production, while the other target, using the exact same pool of source code creates a 'delivery' edition with no debugging symbols and fully optimized.

Anywhere along the way you can inspect the impact of setting options and assigning StyleSheets by using the Options Inspector. You can reach the Inspector by:

1) Bring up the SpeedMenu for node in the Project View by hitting
   the right-button while the cursor is over that node.
2) Select 'Inspect Options'

To see the full effect of this example select:

Options|Environment|Project View|Style Sheet

This will allow you see the Style Sheets assigned throughout the tree while in the Project View.

As you can now see, the Whello16 [.exe] will be created with full debugging on, whereas the Whello32 [.exe] will be created with debugging off and optimized for speed. Please note that even though the "My whello source" SourcePool is reference copied, applying these StyleSheets (or any local overrides) at the reference nodes, has no effect on the other copies and allows maximum flexibility for creating the right targets.

This is accomplished with the following steps:
1) Bring up the SpeedMenu for the SourcePool under Whello16 by hitting
   the right-button while the cursor is over that node.
2) Select 'Node Attributes'
3) Select from the drop-list called StyleSheets one of pre-existing
   StyleSheets to apply to the node.
4) Optionally you can create your own StyleSheet (either based on a
   pre-existing one, or from scratch) by pushing the 'Styles...'
   button in this dialog or from Option|Style Sheets from the main
   menu.

Project Filename: XREF.IDE
Project Directory: EXAMPLES\CLASSLIB\XREF

A BIDS (Borland International Data Structures) example that shows one way to handle text cross-referencing.

Project Filename: TESTDIR.IDE
Project Directory: EXAMPLES\CLASSLIB\TESTDIR

A BIDS (Borland International Data Structures) example that uses a Filedata class to do sorting of files in a directory.

Project Filename: STRNGMAX.IDE
Project Directory: EXAMPLES\CLASSLIB\STRNGMAX

A BIDS (Borland International Data Structures) example that uses a string class.

Project Filename: REVERSE.IDE
Project Directory: EXAMPLES\CLASSLIB\REVERSE

A BIDS (Borland International Data Structures) example that uses string class together with a stack to reverse strings.

Project Filename: QUEUETST.IDE
Project Directory: EXAMPLES\CLASSLIB\QUEUETST

A BIDS (Borland International Data Structures) example that stores a series of time values in a queue container.

Project Filename: LOOKUP.IDE
Project Directory: EXAMPLES\CLASSLIB\LOOKUP

A BIDS (Borland International Data Structures) example that illustrates the use of a dictionary container class.

Project Filename: LABELS.IDE
Project Directory: EXAMPLES\CLASSLIB\LABELS

This project is a BIDS (Borland International Data Structures) example updates and displays the contents of a mailing list. It uses the classes TISListImp, string, TDate, ipstream, and opstream.

For more information about this example, see the projects main source file, LABELS.CPP.

Project Filename: FILTER.IDE
Project Directory: EXAMPLES\IDE\FILTER

This example builds a variety of different DLLs that serve to import the output from command line tools into the IDE. Look at this file if you need to customize filters for the IDE tools or build filters for other command line tools.

Project Filename: DOSEXAMP.IDE
Project Directory: EXAMPLES\DOS

This project file builds the examples located in the EXAMPLES\DOS directory.

Project Filename: APPLAUNC.IDE
Project Directory: EXAMPLES\OWL\OWLAPPS\APPLAUNC

The AppLauncher project creates APPLAUNC.EXE. This application uses a floating button bar, with each button representing an application that can be started by pressing a button. This kind of utility provides an alternative to the program groups / icons structure of the Windows Program Manager.

For more details on how to use APPLAUNC, build the application and select Help from the Applaunc system menu.

Project Filename: DIAGXPRT.IDE
Project Directory: EXAMPLES\OWL\OWLAPPS\DIAGXPRT

The DIAGXPRT executable displays diagnostic messages sent by OutputDebugString() or by the OWL diagnostic macros, TRACE & WARN.

OWL diagnostics can be enabled or disabled by using the configure button. There are two levels of diagnostics used in OWL, 0 and 1. You can configure this for each area.

You can also add your own diagnostic groups. Each new group will be written to the OWL.INI file and the declaration placed on the clipboard for inclusion in your code.

Please see the OWL documentation for a complete description of the diagnostic macros and their use.

Project Filename: MDISTRM.IDE
Project Directory: EXAMPLES\OWL\OWLAPI\MDISTRM

This example shows a persistent desktop using the MDI metaphor. It is identical to the simpler example, MDI , but with streaming code added to implement persistence for the frame, parent (client) window, and children.

Project Filename: OWLBWCC.IDE
Project Directory: EXAMPLES\OWL\OWL_1\BWCC

This project file does not create an executable; instead, it creates a library of OWL 1 interface objects. This project is included to help maintain backward compatibility with OWL 1.

Project Filename: OLDFILEW.IDE
Project Directory: EXAMPLES\OWL\OWL_1\OLDFILEW

This project file does not create an executable; instead, it creates a number of OWL 1 interface objects (EDITWND, FILEWND, and FILEDIAL objects). This project is included to help maintain backward compatibility with OWL 1.