SDWB 5.0

# ODE Build Feature for WebSphere Studio User Guide

**September 30, 2003**

# Contents

# Figures

# Tables

# About this guide

The ODE Build Feature for Websphere Studio User Guide presents information about the Open Development Environment (ODE) Feature for WebSphere Studio. ODE is a development environment that enables common build processes across multiple operating systems. Other advantages of ODE include:

- Provides dependency generation features
- Simplifies makefiles
- Standardizes makefiles for multiple platforms
- Provides NLS support

The ODE Build Feature for WebSphere Studio enables you to perform the following ODE actions in an Eclipse environment:

- Generate makefiles for Java and C/C++ projects
- Build projects with ODE inside the Workbench
- Build a pre-existing sandbox
- View backing chain file listings

## Who should read this guide?

Read this guide if you use ODE to build your products. This guide is written for development builders who want to use ODE in an Eclipse environment. The guide assumes that the reader is familiar with ODE concepts such as *sandbox, backing build*, and *makefiles*.

## What else should you read?

For ODE conceptual and reference information, see the ODE product documentation at http://w3.sdwb.ibm.com/ode/documentation.html.

To access all SDWB documentation, see the SDWB Web site at http://w3.sdwb.ibm.com/documentation.html

## Terminology used in this guide

For more information about SDWB terminology, see the SDWB Glossary at http://w3.sdwb.ibm.com/glossary.htm.

For more information about IBM and industry terminology, search the IBM Terminology Database at http://ibmterm.torolab.ibm.com/.

## How to send your comments

We welcome and appreciate your input on the documentation. Please send any comments or suggestions on this guide to sdwbid@us.ibm.com.

# Chapter 1. Prerequisites

## Supported WebSphere Studio Workbench products

To use the ODE Build Feature, you must first install a supported WebSphere Studio Workbench (WSWB) product.

### What is the WebSphere Studio Workbench?

IBM's WebSphere Studio Workbench (WSWB) is an Integrated Development Environment (IDE) based on the open-source Eclipse Project. Products across IBM are being redesigned and delivered as components that extend the base WSWB platform. These components, known as plug-ins, are packaged together as product offerings, known as features.

In this Guide, `WebSphere Studio Workbench` and `Workbench` refer to the WebSphere Studio products that are based on the WSWB IDE and supported by the ODE Build Feature.

In addition to providing an environment for integrated development, the Workbench enables developers to seamlessly integrate their client applications into a framework that provides a common and consistent interface for users of diverse applications.

Developers can use the prepackaged Workbench presentation and functions, or they can extend and enhance their applications using Workbench extension points. This can include simple enhancements such as menu items and icons, or more complex enhancements such as application-specific perspectives and views. To integrate applications into the Workbench, developers create Eclipse plug-ins (Java and XML code) which are packaged as features, such as the ODE Build Feature. Users can easily install and update these features using the Workbench Install/Update Manager.

**Related information:**

- For more information about WebSphere Studio Workbench, see the basic tutorial in the Getting Started section of your Workbench product documentation. To access your Workbench documentation, click **Help > Help Contents**.
- For more information about Eclipse and the open-source Eclipse Project, see the Eclipse Web site at http://www.eclipse.org/.

### Supported Workbench products and versions

The ODE Build Feature can be used with the following WebSphere Studio products:

- WebSphere Studio Workbench (WSWB), Version 2.1 and 2.1.1.
- WebSphere Studio Application Developer (WSAD), Version 5.1
- WebSphere Studio Application Developer Integration Edition (WSADIE), Version 5.1
- WebSphere Studio Enterprise Developer (WSED), Version 5.1

- WebSphere Studio Site Developer (WSSD), Version 5.1
- Any other WebSphere Studio product based on WSWB, Version 2.1 or 2.1.1

**Note:** If you are a new Workbench user, see the Basic Tutorial in the Getting Started section of your Workbench product documentation. To access Workbench documentation, click **Help > Help Contents**.

### Workbench download sites

*Table 1. Download sites for WebSphere Studio products*

| **Product** | **URL** |
|---|---|
| All WebSphere Studio products except WSWB | http://w3.ibm.com/software/xl/ui2/downloads/downloads.jsp |
| WSWB, Version 2.1.1 | ftp://wswbanon@ott4f.ott.oti.com/inet1/teams/wswb/anon/out/2.1.1-GA/ |
| WSWB, Version 2.1 | ftp://wswbanon@ott4f.ott.oti.com/out/2.1/wswb-2.1-200303280912/ |

## Supported ODE base application versions

The ODE Build Feature supports Version 4.0 or higher of the ODE base application. The required base application files are packaged with the Build Feature. You do not need to perform a separate install to get the base files.

**Note:** The default value of the PATH environment variable is the location of the base files included with the Build Feature. To use your local version of the ODE base application, specify the path to your local version on the PATH environment variable.

# Chapter 2. Installing, updating, or removing the ODE Build Feature

## Installing the ODE Build Feature

Use the steps in this section to install the ODE Build Feature for the first time. Before beginning, you must install a supported WebSphere Studio Workbench product. For more information, see Chapter 1, “Prerequisites,” on page 1.

### Step 1. Open the Workbench Install/Update perspective

Click **Window > Open Perspective > Install/Update**. If **Install/Update** is not in the list, click **Other** to display all available perspectives.

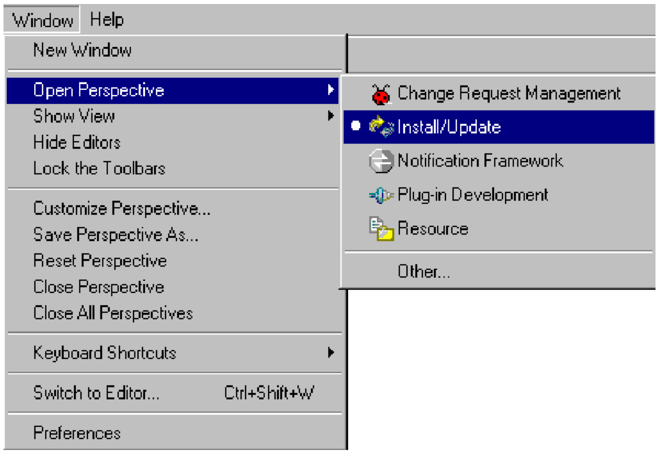


*Figure 1. Opening the Workbench Install/Update Perspective*

### Step 2. Create a site bookmark for the SDWB download site

Before you can download the ODE Build Feature, you must create a site bookmark for the SDWB download site. If you have already created this bookmark for another SDWB feature install, go to “Step 3. Install the ODE Build Feature” on page 5.

To create a site bookmark:

1. Right-click anywhere in the Feature Updates view, then click **New > Site Bookmark**.

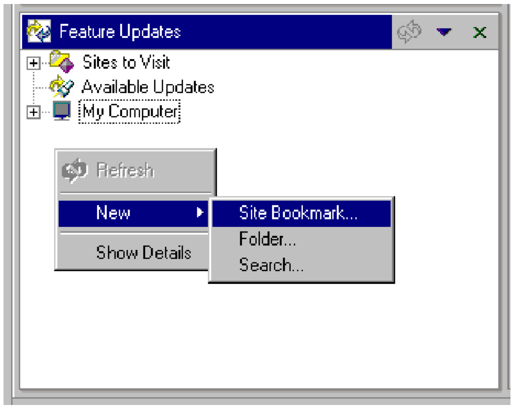
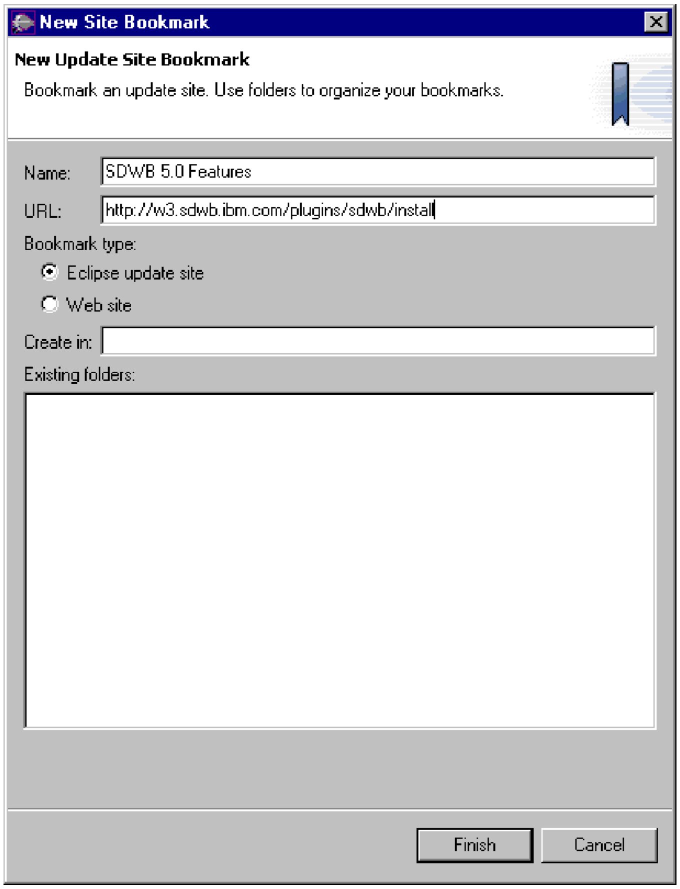


*Figure 2. Creating a new Site Bookmark*

2. In the New Site Bookmark window:
   a. Type a site name of your choice, for example, **SDWB 5.0 Features**.
   b. Type the SDWB download site URL: **http://w3.sdwb.ibm.com/plugins/sdwb/install/**.
   c. Select **Eclipse update site**.
   d. Click **Finish**.



**New Update Site Bookmark**

Bookmark an update site. Use folders to organize your bookmarks.

Name: SDWB 5.0 Features

URL: http://w3.sdwb.ibm.com/plugins/sdwb/install

Bookmark type:

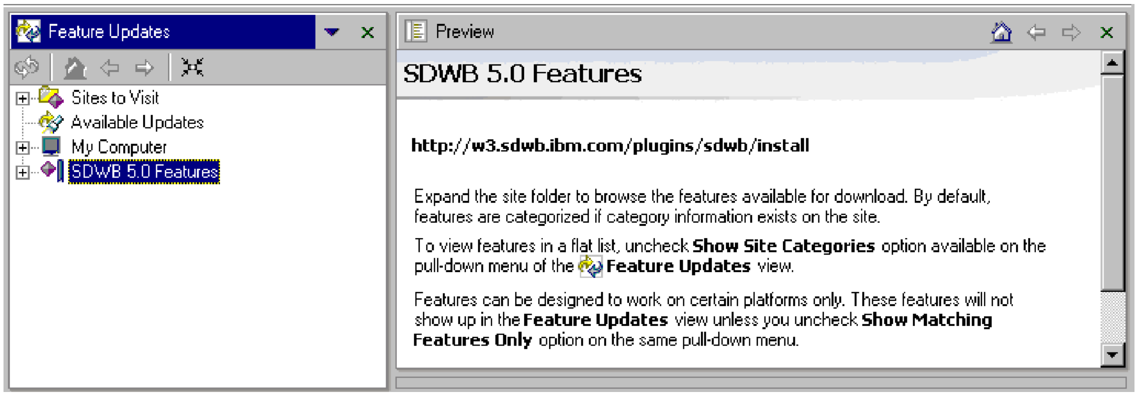
● Eclipse update site

○ Web site

Create in:

Existing folders:

*Figure 3. Settings for the SDWB 5.0 download site bookmark*



SDWB 5.0 Features

**http://w3.sdwb.ibm.com/plugins/sdwb/install**

Expand the site folder to browse the features available for download. By default, features are categorized if category information exists on the site.

To view features in a flat list, uncheck **Show Site Categories** option available on the pull-down menu of the **Feature Updates** view.

Features can be designed to work on certain platforms only. These features will not show up in the **Feature Updates** view unless you uncheck **Show Matching Features Only** option on the same pull-down menu.

*Figure 4. SDWB Features Site Bookmark*

## Step 3. Install the ODE Build Feature

1. In the Feature Updates view, expand the SDWB download site folder.

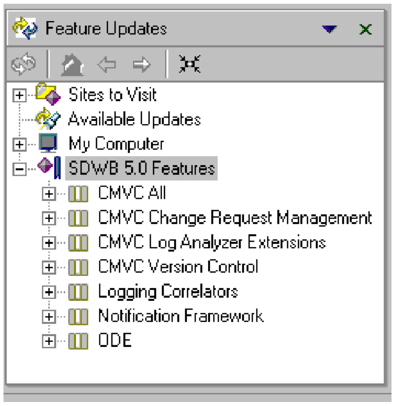
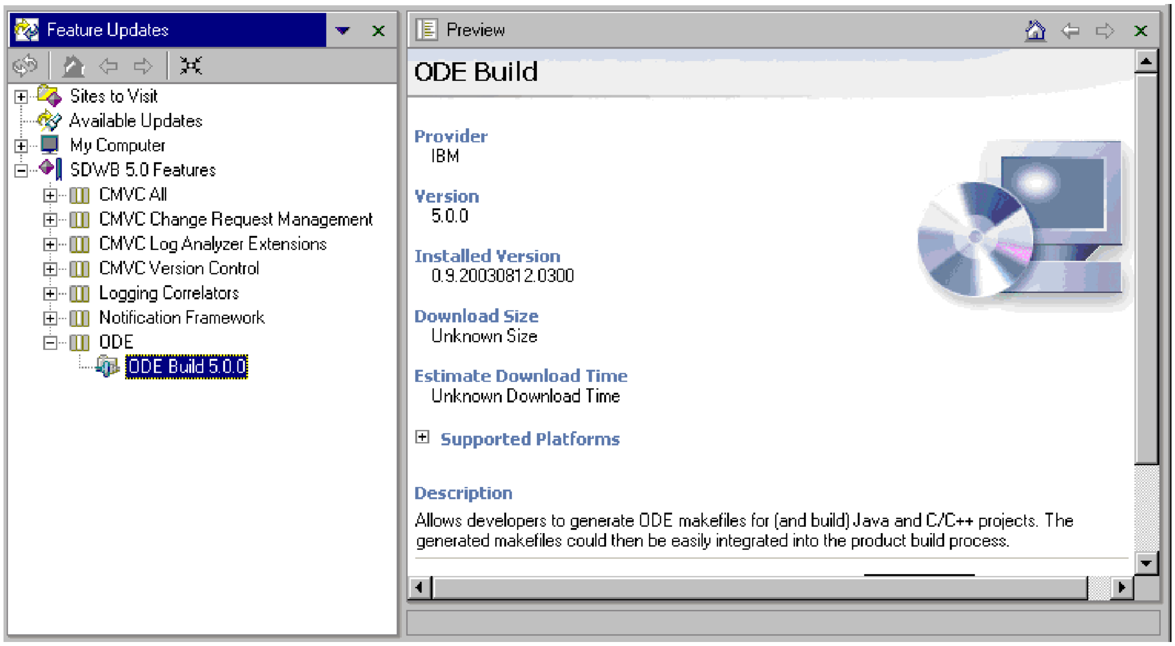


*Figure 5. Available SDWB features*

2. Select the ODE Build Feature. Information about the feature is available in the Preview view.



ODE Build

**Provider**
IBM

**Version**
5.0.0

**Installed Version**
0.9.200308120300

**Download Size**
Unknown Size

**Estimate Download Time**
Unknown Download Time

⊞ **Supported Platforms**

**Description**
Allows developers to generate ODE makefiles for (and build) Java and C/C++ projects. The generated makefiles could then be easily integrated into the product build process.

*Figure 6. Install information for ODE Build Feature*

3. To install, click **Install Now** > **Next**.
4. In the Feature License window, review and accept the license agreement, then click **Next**.
5. In the Install Location window, specify an install directory for the feature, then click **Finish**.

   **Recommendation:**

   For maintenance purposes, it is recommended that you specify a separate install directory for each feature. See "Removing features" on page 8.
6. In the Feature Verification window, click **Install** to complete the installation.
7. When prompted to restart the Workbench:
   - Click **No** if you are adding or updating other features.
   - Click **Yes** if you have completed all feature installs and updates.

## Installing feature updates

### Step 1. Check for available feature updates

To check for available updates, click **Help> Software Updates** > **New Updates**.

### Step 2. If updates are available, open your Workbench Install/Update perspective

Click **Window** > **Open Perspective** > **Install/Update**. If **Install/Update** is not in the list, click **Other** to display all available perspectives.

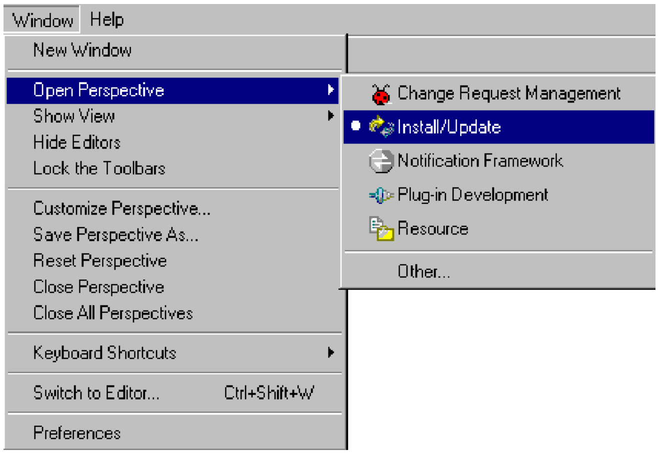


*Figure 7. Opening the Workbench Install/Update Perspective*

## Step 3. Install updates for the ODE Build Feature

1. In the Feature Updates view, expand the SDWB download site folder.

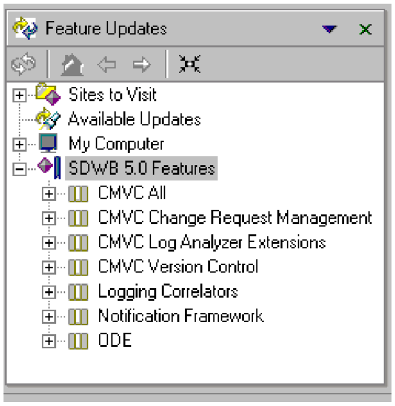
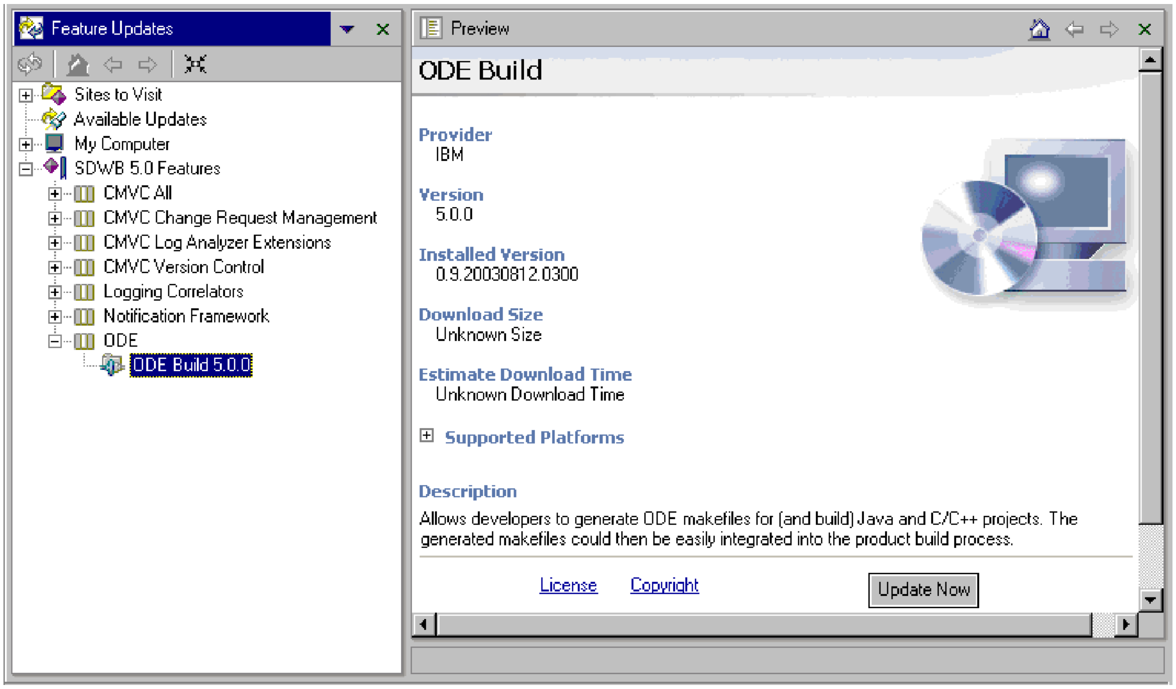


*Figure 8. Available SDWB features*

2. Select the ODE Build Feature:
   - Click **Install Now > Next** to install the feature updates.
   - If there is no **Update Now** push button in the Preview view, you have the latest updates.



ODE Build

Provider
IBM

Version
5.0.0

Installed Version
0.9.20030812.0300

Download Size
Unknown Size

Estimate Download Time
Unknown Download Time

Supported Platforms

Description
Allows developers to generate ODE makefiles for (and build) Java and C/C++ projects. The generated makefiles could then be easily integrated into the product build process.

*Figure 9. Update information for ODE Build Feature*

3. In the Feature License window, review and accept the license agreement, then click **Next**.
4. In the Install Location window, click **Finish**.
5. In the Feature Verification window, click **Install** to complete the installation.
6. When prompted to restart the Workbench:
   - Click **No** if you are adding or updating other features.
   - Click **Yes** if you have completed all feature installs and updates.

## Removing features

The Workbench allows you to disable features. However, disabling a feature does not remove the feature from your file system.

The Workbench does not provide a feature uninstall option. Features must be removed from the file system manually. If you need to remove a Workbench feature, contact SDWB support for assistance.

# Chapter 3. Getting started

To use the ODE Build Feature:

1. Install a supported WebSphere Studio Workbench. See Chapter 1, “Prerequisites,” on page 1.
2. Install the ODE Build Feature using your Workbench Install/Update Manager. See “Installing the ODE Build Feature” on page 3.
3. Review the known limitations of the ODE Build Feature. See “Known limitations of the ODE Build Feature.”
4. Set global preferences for your Workbench and the ODE Build Feature. See “Setting global preferences and properties.”
5. Understand which Workbench views you can use with the ODE Build Feature. See “Using Workbench views to perform ODE Build Feature tasks” on page 17.
6. Create Workbench projects for the ODE Build Feature. See “Creating a new Java project” on page 18.

## Known limitations of the ODE Build Feature

The following limitations should be noted before using the ODE Build Feature on your project:

- The path to the project workspace should not have any spaces in it.
- There can be only one source folder in a project and it should be named *src*.
- The *src* folder must be an immediate subdirectory of the project folder.
- Java source files should be created in a Java package. The source files should not be located directly under the *src* folder.

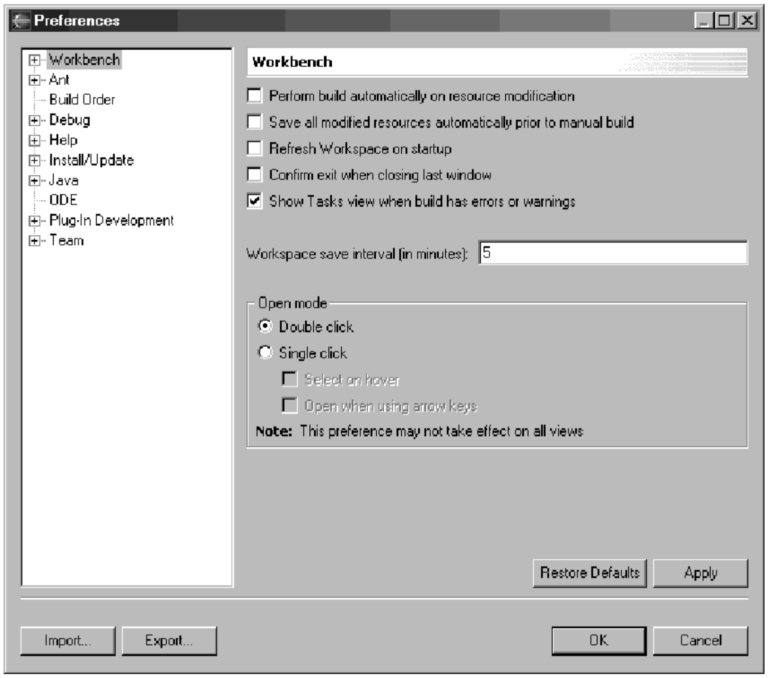
## Setting global preferences and properties

You can set the following global preferences and properties, which affect all ODE Build Feature actions:

- “Workbench preferences”
- “ODE Build Feature preferences” on page 10
- “ODE Build Feature properties” on page 12

### Workbench preferences

To set Workbench preferences, click **Window > Preferences**.



Workbench

☐ Perform build automatically on resource modification

☐ Save all modified resources automatically prior to manual build

☐ Refresh Workspace on startup

☐ Confirm exit when closing last window

☑ Show Tasks view when build has errors or warnings

Workspace save interval (in minutes): 5

Open mode

● Double click

○ Single click

☐ Select on hover

☐ Open when using arrow keys

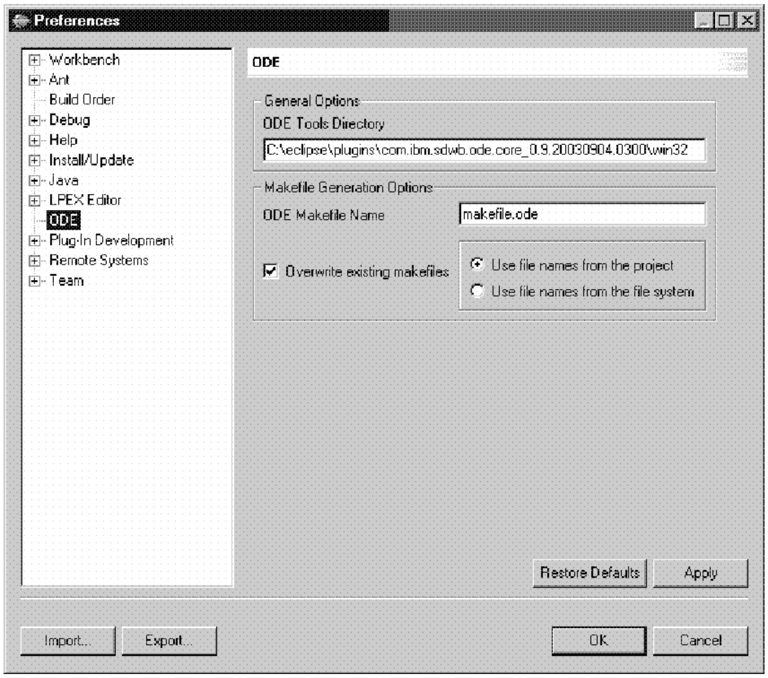
**Note:** This preference may not take effect on all views

*Figure 10. Workbench preferences*

In the Preferences window, clear the **Perform build automatically on resource modification** option. It should be off, or unchecked, while you are using the ODE Build Feature. This prevents the Workbench from automatically building your project using the default build tool (Java Incremental Builder) rather than ODE.

## ODE Build Feature preferences

The ODE feature preferences must be set before using the ODE Build Feature. To view and set ODE feature preferences, choose the ODE option in the **Window > Preferences** window. The options set in the ODE preferences window are applied to all the projects in the Workbench.



ODE

General Options

ODE Tools Directory: C:\eclipse\plugins\com.ibm.sdwb.ode.core_0.9.20030904.0300\win32

Makefile Generation Options

ODE Makefile Name: makefile.ode

☑ Overwrite existing makefiles

● Use file names from the project

○ Use file names from the file system

*Figure 11. Global preferences for the ODE Build Feature*

The following table describes each field in the ODE preferences window:

*Table 2. Global preferences and descriptions*

| Option | Required | Description | Default |
|---|---|---|---|
| ODE Tools Directory | Yes | Specify the path to the directory containing ODE executables (for example, mk, build). The default value is the location of the base files included with the Build Feature. To use your local version of the ODE base application, change the path to the location of the application. | location of build tools supplied by the ODE Build Feature |
| ODE Makefile Name | Yes | Specify a file name for makefiles that are generated by the **ODE > Generate Makefiles** action. Currently, it is preferable to use one of the four ODE recognized names for this value. The preferred names are: *makefile.ode, Makefile.ode, makefile, Makefile*. | makefile.ode |
| Overwrite existing makefiles | No | This check box specifies whether to overwrite existing makefiles. If this check box is cleared (unchecked) and there are existing makefiles, the **ODE > Generate Makefiles** action will not overwrite the existing makefiles. | On |
| Use file names from the project | No | This option specifies that the names of the Java classes and the directories will be hard coded in the makefiles. | On |
| Use file names from the file system | No | This option specifies that the ODE rules will obtain the list of .java files from the file system at run time. This option should not be used if you have .java files in your project that you do not wish to be compiled. | Off |

## ODE Build Feature properties

To set the ODE project properties:

1. Right-click on an ODE project in the Navigator view.
2. Click **Properties**.

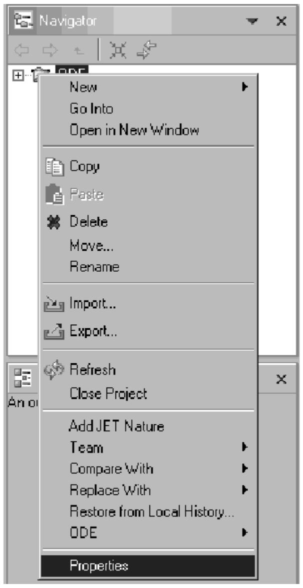
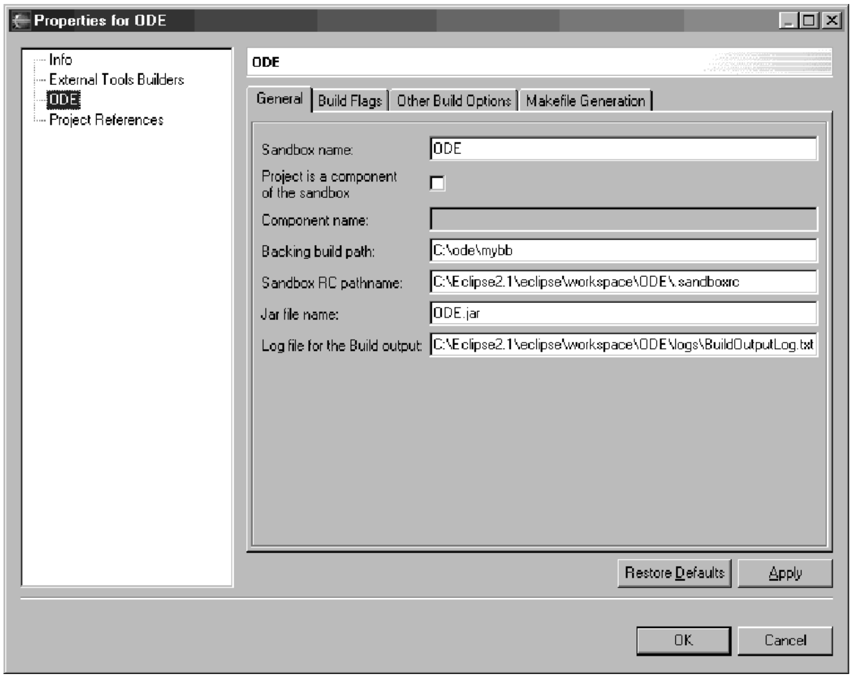


*Figure 12. Setting ODE project properties*

3. Select ODE from the list of possible properties.

Figure 13 on page 13 shows the General section of the ODE properties window and tabbed sections for other properties. To select another section, click on the desired tab.



Sandbox name: ODE

Project is a component of the sandbox □

Component name:

Backing build path: C:\ode\mybb

Sandbox RC pathname: C:\Eclipse2.1\eclipse\workspace\ODE\.sandboxrc

Jar file name: ODE.jar

Log file for the Build output: C:\Eclipse2.1\eclipse\workspace\ODE\logs\BuildOutputLog.txt

*Figure 13. General properties for ODE*

Table 3 describes each field in the General section of the ODE project properties window:

*Table 3. General project properties: ODE Build Feature*

| Option | Required | Description | Default |
|---|---|---|---|
| Sandbox name | Yes | Specifies the name for a sandbox that will be created by the **ODE > Build Project** action. | project name |
| Project is a component of the sandbox | No | This check box is used to specify if the project corresponds to a subdirectory in the source tree of an existing sandbox. This should not be used if you are new to ODE. See “additional information” on page 14. | Off |
| Component name | No | Specifies the path to the subdirectory of the sandbox relative to the source directory of an existing sandbox. This field is enabled only if the **Project is a component of the sandbox check** box is selected. See “additional information” on page 14. | none |
| Backing build path | Yes | Specifies the path to a backing build. | C:\ode\mybb on Windows<br>/ode/mybb on Unix |
| Sandbox RC pathname | No | Specifies the path to the sandbox configuration file. Specifying no value in this field is equivalent to specifying *%HOME%\.sandboxrc*. | <project directory>\.sandboxrc |

*Table 3. General project properties: ODE Build Feature (continued)*

| Option | Required | Description | Default |
|---|---|---|---|
| Jar file name | No | Specifies the name of the jar file created by the **ODE > Build Project** action. This jar file will contain all the class files in the output directory. The value of this field is also used by the **ODE > Generate Makefiles** action to generate **JAR_LIBRARIES** makefile variable in the top level makefile. | none |
| Log file for the Build output | No | Specifies the path to the ODE build output file. This file will contain the output of the last build command. The output of the previous build command is stored in the .prev file at the same location. If this field is empty, then a log file is not generated. | none |

**Additional information on component options:**

Selecting the **Project is a component of the sandbox** option indicates that your project is a single subdirectory tree of an existing sandbox, rather than an entire sandbox. This is typically the case when the directories in *src* are subprojects, and you are working in a single subproject. For example, if your ODE sandbox contains subdirectories under *src* called *project1* and *project2*, and your project only includes the files in *project1*, you can select the checkbox and type *project1* in the **Component name** field. With this option selected, all project-level ODE builds and sandbox file listings apply to *src/project1* rather than *src*. The component name field contains only one directory, but the path to that directory may be of any depth (for example, *com/ibm/myproject*).

The Build Flags section of the ODE project properties window is shown in Figure 14:

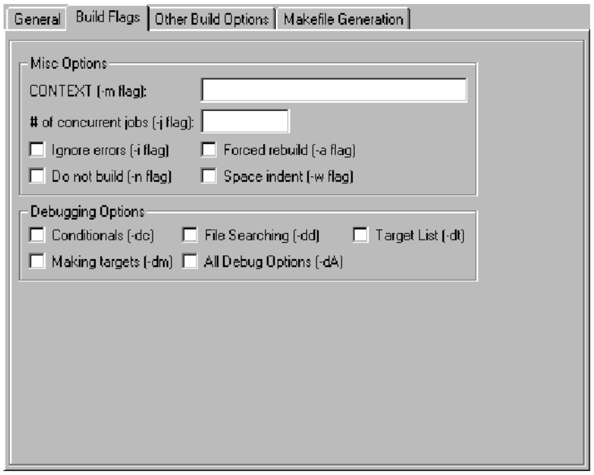


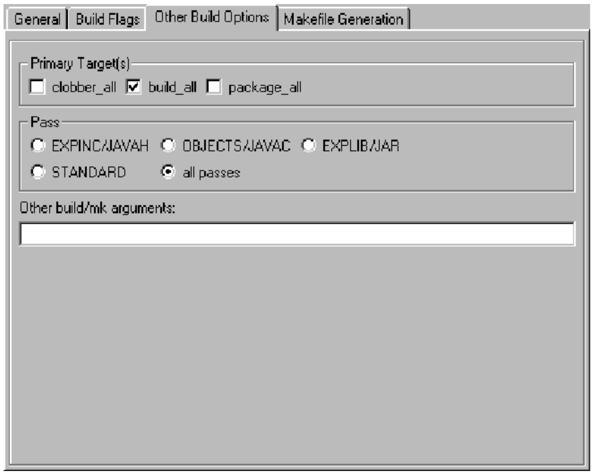
*Figure 14. Build Flags properties for ODE*

Table 4 describes each field in the Build Flags section of the ODE project properties window:

*Table 4. Build Flags project properties: ODE Build Feature*

| Option | Required | Description | Default |
|---|---|---|---|
| CONTEXT | No | Permits the user to specify the -m flag argument on the *build* command. This becomes the value of the environment variable CONTEXT during the build session. | none |
| # of concurrent jobs | No | Contains the argument for the -j flag. This specifies the number of concurrent jobs that can execute for each directory. | none |
| Ignore errors | No | Activates the -i flag, which causes all errors that mk encounters from spawned commands to be ignored. | Off |
| Forced rebuild | No | Turns on the -a flag, which causes all targets to be recreated, even if they are up-to-date. | Off |
| Do not build | No | Enables the -n flag, which causes mk to display the commands on the screen. No build is performed. | Off |
| Space indent | No | Activates the -w flag, which allows ordinary white space to be used in makefiles to indent commands (hard tabs are required by default). | Off |
| Debugging options | No | Enables various debugging options. Refer to the ODE Make Reference for details. | none |

The Other Build Options section of the ODE project properties window is shown in Figure 15:



Primary Target(s)

□ clobber_all ☑ build_all □ package_all

Pass

○ EXPINC/JAVAH ○ OBJECTS/JAVAC ○ EXPLIB/JAR

○ STANDARD ● all passes

Other build/mk arguments:

*Figure 15. Other Build Options properties for ODE*

Table 5 describes each field in the Other Build Options section of the ODE project properties window:

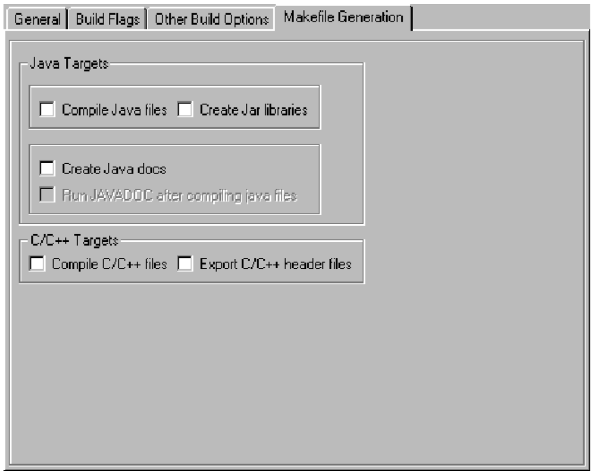
*Table 5. Other Build Options project properties: ODE Build Feature*

| **Option** | **Required** | **Description** | **Default** |
|---|---|---|---|
| Primary Target(s) | No | Allows the user to select one or more main ODE targets to run. build_all is the default target. If more than one target is selected, the order is always as follows: clobber_all, build_all, and package_all. | build_all |
| Pass | No | Allows a particular ODE pass to be run. The default is for all of the individual passes shown to be run. | all passes |
| Other build/mk arguments | No | Specifies any other command line options for the ODE *build* and/or *mk* commands. Refer to the ODE Make Reference for a full list of valid parameters that can be passed to the ODE *mk* command. | none |

To learn more about the ODE build and mk commands, refer to the ODE Documentation Web page at http://w3.sdwb.ibm.com/ode/5.0/doc/pubs/index.htm. Consult the following reference documents:

- build command: ODE Build Reference at http://w3.sdwb.ibm.com/ode/5.0/doc/pubs/ODEBuildReference.htm
- mk command: ODE Make Reference at http://w3.sdwb.ibm.com/ode/5.0/doc/pubs/ODEMakeReference.htm

The Makefile Generation section of the ODE project properties window is shown in Figure 16:



□ Compile Java files □ Create Jar libraries

□ Create Java docs

□ Run JAVADOC after compiling java files

□ Compile C/C++ files □ Export C/C++ header files

*Figure 16. Makefile Generation properties for ODE*

Table 6 describes each field in the Makefile Generation section of the ODE project properties window:

*Table 6. Makefile Generation project properties: ODE Build Feature*

| Option | Required | Description | Default |
|---|---|---|---|
| Compile Java files | No | Causes a JAVA_CLASSES variable to be created in each makefile, which contains a list of all the .java files in the packages corresponding to each directory. | none |
| Create Jar libraries | No | Creates a JAR_LIBRARIES entry in the makefile so the entire Java build output tree will be put into a jar file. | none |
| Create Java docs | No | Causes the JAVADOCS variable to be created in order to run the javadoc command so that Java documentation is generated. | none |
| Run JAVADOC after compiling java files | No | Allows the user to override the default time during which ODE creates Java documentation. Normally, it is run first, before the EXPINC/JAVAH pass. Selecting this checkbox will cause the documentation to be generated after the OBJECTS/JAVAC pass. Many users wish to ensure their source code compiles fully before generating documentation, and this is how to enable that behavior. | none |
| Compile C/C++ files | No | Causes an OBJECTS variable to be created in each directory which contains all C/C++ source files (.c, .cpp, etc.) in that directory. This will trigger the files to be compiled during the build. Users must edit the makefiles they wish in order to take action beyond object file creation, such as creating programs or libraries (with the PROGRAMS, LIBRARIES, and/or SHARED_LIBRARIES variables). | none |
| Export C/C++ header files | No | Creates an INCLUDES variable in each directory that contains a list of all header files (.h, etc.) in that directory, along with a default export location. | none |

## Using Workbench views to perform ODE Build Feature tasks

You can perform ODE Build Feature tasks using the following Workbench views:

- Navigator View
- “Package Explorer view” on page 18
- “C/C++ Project view” on page 18

To show Workbench views in your workspace:

- Click **Window > Show View**.
- Select a view from the list or select **Other** for additional views.

- Click **OK** to show the view in your workspace.

**Note:** If these views are not available in the list of views in your workspace, you may have to open a specialized perspective first. Please see the "Package Explorer view" or "C/C++ Project view" sections for more information.

### Package Explorer view

The Package Explorer view is available when you open the Java perspective in your Workbench. To open the Java perspective:

1. Click **Window > Open Perspective**.
2. Select Java from the list of possible perspectives.

When you select the Java perspective from the shortcut bar at the far left of the Workbench window, you will see the Package Explorer view next to the shortcut bar.

### C/C++ Project view

The C/C++ Project view is available when you open the C/C++ Development perspective. This perspective is added to your Workspace when you install the C/C++ Development Tools (CDT) plugin. This plugin is similar to the Java Development Tools (JDT) plugin and provides a C/C++ development environment for C/C++ programmers using Eclipse or the WebSphere Studio Workbench (WSWB). For more information on the CDT plugin see the Eclipse Project Web site at http://www.eclipse.org/cdt.

To open the C/C++ Development perspective:

1. Click **Window > Open Perspective**.
2. Select **C/C++ Development** from the list of possible perspectives (this may be in the Other section of the list).

## Creating a new Java project

The ODE Build Feature can generate makefiles for either Java or C/C++ projects. Before you can perform any build actions using the ODE Build Feature, you must create a project within the Workbench. Review the known limitations of the ODE Build Feature before creating the new project.

To create a new Java project in your Workbench:

1. In the Navigator or the Package Explorer view, right-click inside the Navigator space (not on any existing project) and click **New > Project**.
2. In the New Project window, select Java from the list of possible project types.
3. Click **Next**.
4. Type a name for your project in the Project Name field.
5. Click **Next**.
6. Add a new *src* (source) folder to the project, as follows:
   a. In the Java Settings panel, make sure that the Source tab is selected.
   b. Expand the project folder.
   c. Click **Add Folder**.
   d. In the Source Folder Selection window, click **Create New Folder**.
   e. In the New Folder window, type **src** as the folder name.

f. When you return to the Source Folder Selection window, a tree view of your new project is shown with the project name at the top level, and src as the only child directory.

g. Click **OK**.

h. When asked, ″Do you want to remove the project as source folder and update build output folder to ″[project_name]/bin?″, click **No**.

i. The Source Folder window appears, indicating that you have successfully added the folder.

7. The standard default output folder is the *\export\classes* folder. Add the name of this folder to the project by typing the folder path in the **Default ouput folder** field at the bottom of the New Java Project window. For example, if your project name is *test*, then the path to the default output folder should be *test\export\classes*.

8. Click **Finish** to create the new Java project.

# Chapter 4. ODE Build Feature tasks

## Generating makefiles

Makefiles are required to build a project with ODE. Review the prerequisities before generating makefiles.

### Prerequisites for generating makefiles

- Make sure the ODE Build Feature is installed and configured properly, as described in “Installing the ODE Build Feature” on page 3.
- Select necessary makefile targets in the Makefile Generation section of the ODE Properties window. See “ODE Build Feature properties” on page 12 for details.
- Create a Java (or C/C++) project in your Workbench. Detailed instructions are provided in “Creating a new Java project” on page 18.

### Generating makefiles for a Java project

The **Generate Makefiles** command can generate makefiles with the necessary makefile variables for compiling Java source files and for creating a jar file, but first you must select the correct options in the Makefile Generate section of the ODE properties window:

1. Right-click on your project name in the Navigator view (or Package Explorer view), then click **Properties**.
2. Select ODE from the list of possible properties.
3. In the ODE window, click on the **Makefile Generation** tab.
4. Select the desired option(s) under Java Targets:
   - **Compile Java files** (generates necessary makefile variables for compiling Java source files)
   - **Create Jar libraries** (generates necessary makefile variables for creating a jar file)
5. Click **OK** to save the change.

To generate makefiles for a Java project:

1. Right-click on the project name in the Navigator view (or the Package Explorer view).
2. Click **ODE > Generate Makefiles**.

### Generating makefiles for a C/C++ project

Before generating makefiles for a C/C++ project, select the makefile targets in the ODE project properties page:

1. Right-click on your project name in the Navigator view (or Package Explorer view), then click **Properties**.
2. Select ODE from the list of possible properties.
3. In the ODE panel, click on the **Makefile Generation** tab.
4. Select the **Compile C/C++ files** and the **Export C/C++ header files** options under C/C++ Targets.
5. Click **OK** to save the change.

Selecting these options instructs the Generate Makefiles command to generate makefiles with all the necessary makefile variables to compile C/C++ source files and to include the header files.

To generate makefiles for a C/C++ project:

1. Right-click on the project name in the Navigator view (or the C/C++ Projects view).
2. Click **ODE > Generate Makefiles**.

## Generating makefiles for a project containing Java and C/C++ files

Makefiles can be generated for a project containing both Java and C/C++ files. For a project to have both Java and C/C++ project properties, it must be associated with both C/C++ and Java natures. The natures settings are stored inside the *.project* XML file in the root directory of the project.

To create the natures elements, shut down the Workbench, open the *.project* file and add three nature elements with the following content: *org.eclipse.cdt.core.cnature*, *org.eclipse.cdt.core.ccnature*, and *org.eclipse.jdt.core.javanature*. The nature elements are added inside the <natures> element. When completed, the XML coding looks like this example:

```
<natures>
   <nature>org.eclipse.cdt.core.cnature</nature>
   <nature>org.eclipse.cdt.core.ccnature</nature>
   <nature>org.eclipse.jdt.core.javanature</nature>
</natures>
```

## Changing the file name of the makefiles

The file name assigned to makefiles that you create using the **Generate Makefiles** command is determined by the value set in the ODE Makefile Name field. The default value for this field is *makefile.ode*, but you can type a different value in the ODE Makefile Name field to change the file name. See “ODE Build Feature preferences” on page 10 for more information. The new value will be used when the **Generate Makefiles** command is issued again.

## Overwriting existing makefiles

You can set the behavior of the **Generate Makefiles** command to overwrite existing makefiles, or not overwrite existing files. By default, the Generate Makefiles command will overwrite existing makefiles in the same path. Clear this option to change the behavior. See “ODE Build Feature preferences” on page 10 for more information.

## Building a project using the ODE Build Feature

The **ODE > Build Project** action runs the ODE *build* command with the build options and debugging options specified in the ODE properties window.

### Prerequisites for building a project

These preconditions should be satisfied in order to successfully build a project with ODE:

- Make sure the ODE Build Feature is installed and configured properly, as described in “Installing the ODE Build Feature” on page 3.
- Generate makefiles for the project, or use existing makefiles (such as a backing build).
- Ensure that all paths and names in the General section of the ODE properties window are correct. See “ODE Build Feature properties” on page 12 for more information.

Before building a project or generating a project’s sandbox file listing for the first time, review and modify, if necessary, the sandbox data in the General section of the ODE Properties window. When you click **OK** or **Apply** in the Properties window, the sandbox and/or backing build is created. A backing build is created at the specified location if you do not already have one. This ensures that the required ODE configuration files and common makefile rules are available. If you clear the **Backing build path** field, your project/sandbox will effectively be the backing build itself, and the files will be extracted to the project directory.

### Building a full project

To build a project:

1. Right-click on the project name in the Navigator view (or the Package Explorer or C/C++ Projects view).
2. Click **ODE > Build Project**.

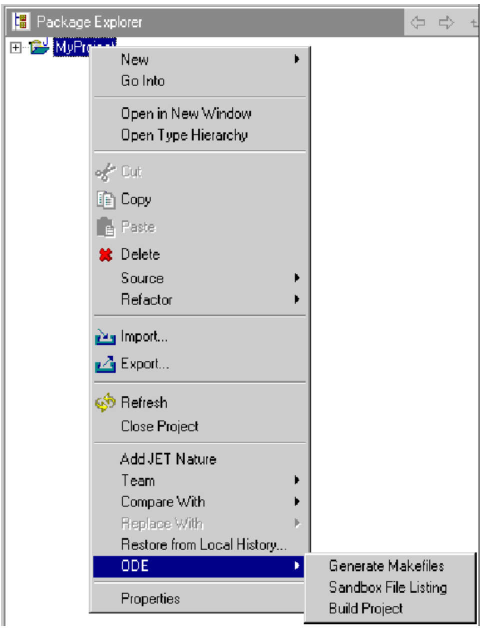


*Figure 17. Building a full project using ODE*

## Building a package or a subdirectory

Each package of a Java project can also be built separately. To build a package:

1. Check the Prerequisites for building a project to make sure you are ready to build.
2. Right-click on the package name (below the main project name) in the Navigator view (or the Package Explorer or C/C++ Projects view).
3. Click **ODE > Build Tree**.

## Specifying the name of the jar file

You can change the name of the jar file that is created from a Java project by the **ODE > Build Project** command. When you select the **Create Jar libraries** option in the Makefile Generation section of the ODE properties window, the Build Project command creates a jar file containing all the .class files that belong to the project. To change the name of this jar file, type the desired jar file name in the **Jar file name** field on the General tab in the ODE properties window. See ″General Project properties″ in Table 3 on page 13 for more information.

## Specifying the log file name

You can change the name of the log file. The output of the **ODE > Build Project** action is stored in the log file specified in the **Log file for the Build output** field. To change the log file name, type a new value in the **Log file for the Build output** field on the General tab in the ODE properties window. See ″General Project properties″ in Table 3 on page 13 for more information.

The specified log file contains the output of the last build action. The output of the next-to-last build action is stored in the *.prev* file.

# Creating a project using an existing sandbox

A project can be created using an existing sandbox. To create a Java (or C/C++) project from an existing sandbox:

1. Click **File > New > Project**.
2. Click **Java > Java Project** in the New Project wizard (or C/C++ Project)
3. Click **Next**.
4. Type a project name.
5. Clear the **Use default** check box.
6. Click **Browse** and navigate to the root directory of the existing sandbox.
7. Click the **Next** button.
8. In the Java Settings panel, click on the name of the existing sandbox project, then click **Add Folder**.
9. In the Source Folder Selection window, open the existing project by expanding the project folder.
10. Click the *src* folder.
11. Click the **OK** button.
12. When asked, ″Do you want to remove the project as source folder and update build output folder to *project_name*/bin?″ click **Yes**.
13. The Source folders on build path is now set to *project_name*/src, and the Default output folder is set to *project_name*/bin.
14. Click **Finish** to create the new project.

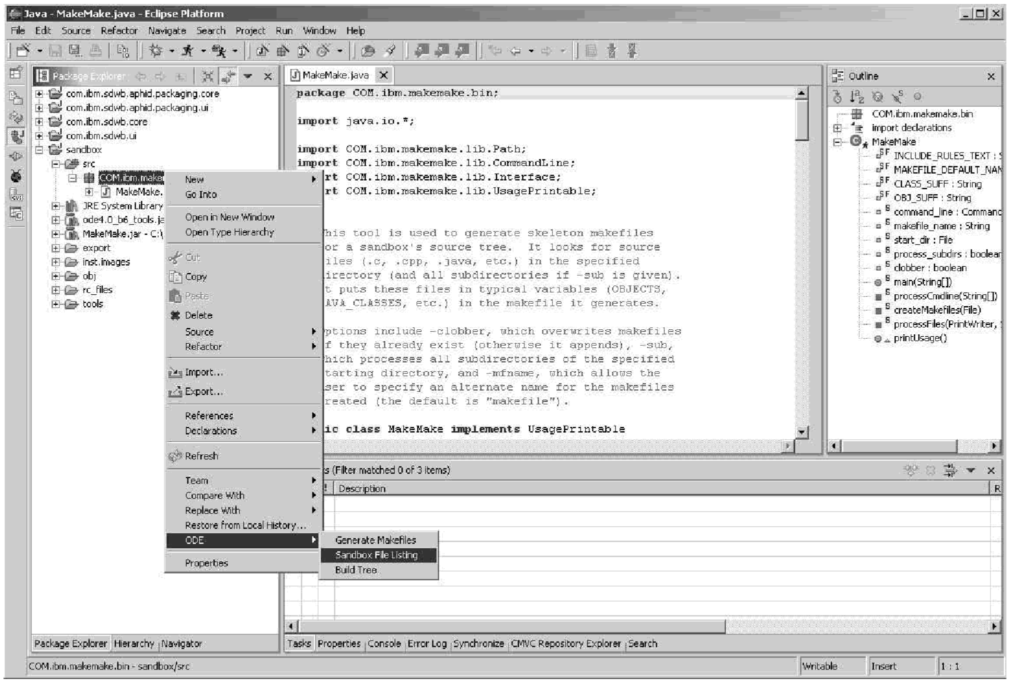
Before continuing to work on the new project, set the path to the backing build and to the sandboxrc file:

1. Right-click on your project name in the Navigator view (or Package Explorer view), then click **Properties**.
2. Select ODE from the list of possible properties.
3. In the ODE panel, click on the **General** tab.
4. Type the path for the backing build in the **Backing build path** field.
5. Type the path for the sandboxrc file in the **Sandbox RC pathname** field.
6. Click **OK** to save the changes.

## Viewing files in the sandbox chain

To view all the files in the sandbox chain, right-click on the project in the Navigator view and then click **ODE > Sandbox File Listing**. This menu is also available in the ″Package Explorer″ or ″C/C++ Projects″ views. The Sandbox File Listing command displays all the files that belong to each sandbox in the sandbox chain.

**Note:** Before generating a project's sandbox file listing for the first time, review and modify, if necessary, the sandbox data in the General section of the ODE Properties window. See "ODE Build Feature properties" on page 12 for more information.



```
package COM.ibm.makemake.bin;

import java.io.*;

import COM.ibm.makemake.lib.Path;
import COM.ibm.makemake.lib.CommandLine;
rt COM.ibm.makemake.lib.Interface;
rt COM.ibm.makemake.lib.UsagePrintable;


his tool is used to generate skeleton makefiles
or a sandbox's source tree.  It looks for source
iles (.c, .cpp, .java, etc.) in the specified
irectory (and all subdirectories if -sub is given).
t puts these files in typical variables (OBJECTS,
AVA_CLASSES, etc.) in the makefile it generates.

ptions include -clobber, which overwrites makefiles
f they already exist (otherwise it appends), -sub,
hich processes all subdirectories of the specified
tarting directory, and -mfname, which allows the
ser to specify an alternate name for the makefiles
reated (the default is "makefile").

ic class MakeMake implements UsagePrintable
```

*Figure 18. Creating a sandbox file listing*

# Index