Java Browser User Guide

Release 6.1

February 2003

Java Browser User Guide

ObjectStore Release 6.1 for all platforms, February 2003

Contents

Preface

Purpose

The *Java Browser User Guide* introduces the ObjectStore and describes how to use it to browse, edit, query, and report on data in ObjectStore® for Java interface (OSJI) databases and PSE Pro for Java databases. You will learn about

- The graphical interface and using the ObjectStore for analyzing databases
- Creating custom views of databases
- Reading and updating objects in the database

Audience and Scope

This guide is for OSJI and PSE Pro for Java database application developers. It assumes some level of familiarity with the concepts and procedures for OSJI and PSE Pro for Java databases.

The Way This Book Is Organized

This book contains the following chapters:

- Chapter 1, Java Browser Overview, on page 1, describes the main components of the ObjectStore, how to start and stop the browser, and how to open and close databases.
- Chapter 2, Working with Database Schema and Roots, on page 11, describes how to examine the various classes that make up the database schema and explains how to work with database roots.
- Chapter 3, Working with Objects, on page 21, describes the way to use the ObjectStore to help you with tasks that you might perform while developing a database application.

Notation Conventions

This document uses the following conventions:

Convention	*Meaning*
`Courier`	`Courier font` indicates code, syntax, file names, API names, system output, and the like.
`Bold Courier`	**`Bold Courier font`** is used to emphasize particular code.
`Italic Courier`	*`Italic Courier font`* indicates the name of an argument or variable for which you must supply a value.
Sans serif	Sans serif typeface indicates the names of user interface elements such as dialog boxes, buttons, and fields.
Italic serif	In text, *italic serif typeface* indicates the first use of an important term.
[ ]	Brackets enclose optional arguments.
{ *a* | *b* | *c* }	Braces enclose two or more items. You can specify only one of the enclosed items. Vertical bars represent OR separators. For example, you can specify *a* or *b* or *c*.
...	Three consecutive periods indicate that you can repeat the immediately previous item. In examples, they also indicate omissions.

ObjectStore on the World Wide Web

ObjectStore has its own Web site (www.objectstore.net) that provides a variety of useful information about products, news and events, special programs, support, and training opportunities.

Technical Support

When you purchase technical support, the following services are available to you:

- You can send questions to support@objectstore.net. Remember to include your site ID in the body of the electronic mail message.
- You can call the Technical Support organization to get help resolving problems.
- You can access the Technical Support Web site, which includes

- A template for submitting a support request. This helps you provide the necessary details, which speeds response time.
- Frequently asked questions (FAQs) that you can browse and query.
- Online documentation for all ObjectStore products.
- White papers and short articles about using ObjectStore products.
- Sample code and examples.
- The latest versions of ObjectStore products, service packs, and publicly available patches that you can download.
- Access to an ObjectStore product matrix.
- Support policies.
- Local phone numbers and hours when support personnel can be reached.

Education Services

Use the ObjectStore education services site (www.objectstore.net/services/education) to learn about the standard course offerings and custom workshops.

If you are in North America, you can call 1-800-477-6473 x4452 to register for classes. For information on current course offerings or pricing, send e-mail to classes@progress.com.

Searchable Documents

In addition to the online documentation that is included with your software distribution, the full set of product documentation is available on the ObjectStore Support Web server. The documentation is found at www.objectstore.net/documentation, and is listed by product. The site supports the most recent release and the previous supported release of ObjectStore documentation. Service Pack README files are also included to provide historical context for specific issues. Be sure to check this site for new information or documentation clarifications posted between releases.

Your Comments

ObjectStore product development welcomes your comments about its documentation. Send any product feedback to support@objectstore.net. To expedite your documentation feedback, begin the subject with `Doc:`. For example:

```
Subject: Doc: Incorrect message on page 76 of reference manual
```

Chapter 1
Java Browser Overview

This chapter describes the benefits of using the Java Browser. It describes the components of the Browser's user interface and how to start the Browser and open a database.

The Java Browser is a graphical tool that lets you browse, edit, and query data in an ObjectStore Java Interface (OSJI) or PSE Pro for Java database.

Contents

This chapter covers the following topics:

Benefits of Using the Java Browser

The Java Browser is a development tool that can help you develop, debug, test, and tune the performance of your database applications.

The Java Browser helps you to

- Understand the schema of your database so you can structure your database applications accordingly.
  - The schema is displayed graphically so you can visually examine the methods and fields of the classes stored in the database.
  - You can create a tabular view, known as a *DataView*, of any persistent *container* in the database. A container is a framework used by the Java Browser to display collections of objects.

    Any classes that implement the `java.util.Collection`, `com.odi.coll.Collection`, `java.util.Map` interfaces, or a Java array can be displayed in a Java Browser container.
  - You can customize DataViews by choosing which fields and methods will be displayed in the view.
  - You can customize DataViews by executing queries against the container.
  - You can save customized DataViews so you can conveniently use them later.
- Debug and test your application.
  - You can change a value of a root or other object in the database to see the way your application behaves — does the application change the objects in the database as you expected?
  - You can invoke class methods to see what happens to the objects in the database.
- Create queries against containers that are stored in the database.
- Create indexes on containers that support the `com.odi.util.IndexedCollection` interface. These indexes enhance the performance of your queries.

Database Metaknowledge

Database metaknowledge is information about the database that is used by the Java Browser. Metaknowledge includes

- DataViews
- Filters applied to DataViews
- String formats specified for DataViews

If you save the database metaknowledge, the various DataViews, filters, and string formats are available the next time you load the database. You can save the metaknowledge in one of two ways: as part of the database or in a separate file in the user's home directory. If you open the database with read-only access, you can only save the metaknowledge to a separate file.

Database Schema and Roots

The database schema consists of all classes stored in the database. The Java Browser depicts the schema in a hierarchical structure that shows all the classes and the fields that belong to each class.

A database root is a persistent entity that is stored in your database. It is associated with a user-defined object that is also stored in your database. You use a database root to navigate to other objects in the database. Databases can have more than one root. For more information about database schema and roots, see Database Panel on page 7 and Chapter 2, Working with Database Schema and Roots, on page 11. You should also refer to the *ObjectStore Java API User Guide*.

Schema and root information is stored with the database.

Starting the Java Browser

To start the Java Browser, enter the following command at the system prompt:

```
java com.odi.browser.Browser
```

An empty Java Browser appears. The Java Browser is a Multiple Document Interface (MDI) application that consists of two main panels: database and

workspace. To activate these panels, you first must open an OSJI or PSE Pro database. Opening a Database on page 4 explains how to do this.

Stopping the Java Browser

To stop the Java Browser, select File | Exit on the menu bar.

Opening a Database

You can open one database at a time in the Java Browser. If you want, you can start multiple Java Browsers and use those sessions to either open the same database or to open several different databases.

To open an OSJI or PSE Pro database in the Java Browser:

1 Click  (Open Database tool) on the main Java Browser toolbar.

Alternative: Select File | Open on the menu bar.

The Open dialog box appears.

2 Select the database that you want to open.

The database appears in the left-hand database panel of the Java Browser.

Alternative: In Windows NT you can drag an OSJI or PSE Pro database and drop it on the Java Browser.

Opening a database as read-only

You can open a database for read-only access by clicking the Open as read-only check box in the Open dialog box. If you open a database for read-only access, you can

- Permit concurrent access to the database
- Prevent unintended changes to the database

When you open a database for read-only access, if you want to be able to save DataViews and other metaknowledge permanently, make sure you enable the Save metaknowledge on the file system option. If the Save metaknowledge on the file system option is not enabled, you can create and use DataViews during a session, but you cannot save the DataViews or other metaknowledge when you exit the session. See Database Metaknowledge on page 3 for more information about metaknowledge.

To set the Save metaknowledge on the file system option:

1 Select Tools | Options from the menu bar.

The Option dialog box is displayed.

2 Click the Global tab.

3 Click the Save metaknowledge on the file system option to add a check mark.

4 Click OK.

Closing a Database

To close an OSJI or PSE Pro database in the Java Browser, select File | Close on the menu bar. You can also select File | Exit on the menu bar if you want to close the database and exit the Java Browser.

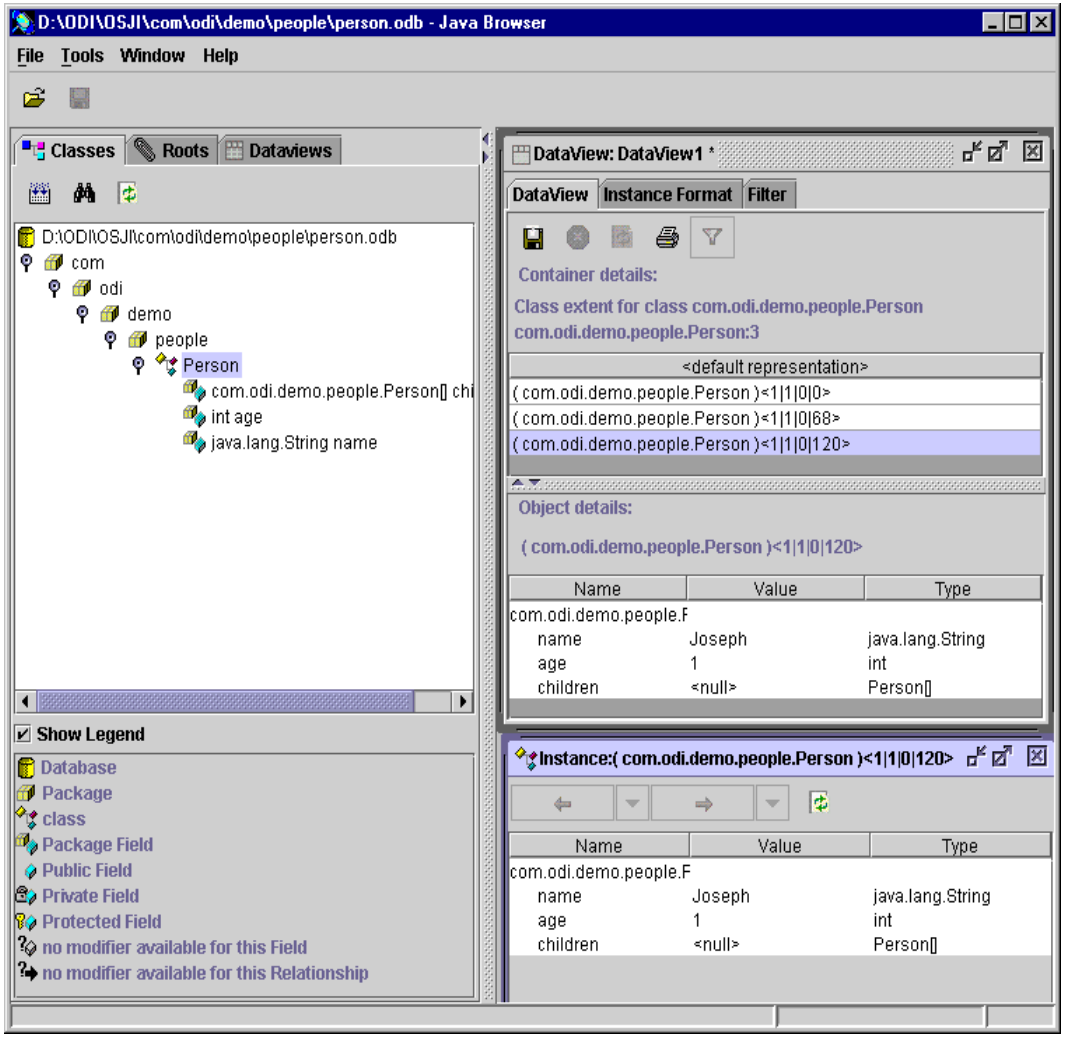
Regardless of the method you choose for closing the database, if you made any changes, the Java Browser prompts you about whether you want to save them as well as the metaknowledge associated with the database.

Java Browser Interface

This section introduces the Java Browser interface components you use to view and modify information stored in OSJI and PSE Pro for Java databases. It provides a brief description of the windows, menus, and tools found in the Java Browser. You use these components to navigate the database and to perform tasks such as modifying the views of the data and querying the database.

The default display of the Java Browser shows two main panels. You use the left-hand panel, the *database panel*, to navigate to the data of interest. You use the right-hand panel, the *workspace panel*, to manipulate the data that is reached through your navigation.

The following figure shows the main panels of the Java Browser:



Database Panel

Workspace Panel

Database Panel

The left-hand panel of the Java Browser is called the database panel. It displays information about an OSJI or PSE Pro database. The panel consists of three tabs, each of which displays different information about the database:

- Classes
- Roots
- DataViews

The tabs in the database panel are docking windows so you can move them around the desktop if you want to view the Schema, Roots, and Dataview tabs simultaneously. Move a tab by clicking the top of the textured band at the left of the tab and dragging it to a new location.

The database panel is your starting point when you first open a database for inspection. From the database panel, you can reach the roots, objects, and the DataViews that are stored in the database.

Classes tab

The Classes tab uses a tree hierarchy to represent the database schema. The database schema consists of all classes stored in the database.

You can navigate this collapsible and expandable hierarchy using your mouse. An optional legend below the tree hierarchy displays the access specifiers of the fields for each Java class stored in the database.

The Classes tab contains a toolbar with icons representing three tools. You use these tools to calculate a class extent; to find a particular object by using its segment, cluster, and offset in the database; or to refresh the database schema display.

Roots tab

The Roots tab displays all roots that exist for the database. The name, value, and type of each root is displayed in tabular form.

The Roots tab contains a toolbar with icons representing four tools. You use these tools to create a root, destroy a root, browse the contents of a particular root, or refresh the display.

Dataviews tab

The Dataviews tab lists all the DataViews that are stored in the database.

You use the Dataviews tab to quickly display a stored DataView. For more information about DataViews, see DataView Window on page 8 and Chapter 3, Working with Objects, on page 21.

Workspace Panel

The right-hand panel of the Java Browser is called the workspace panel. It is the area in the Java Browser where you work with the data in the database. Depending on what you want to do, the workspace panel displays various DataView and Instance windows.

- The DataView window displays a collection of instances of a class. In the DataView window, you create customized views of containers, perform queries on containers, and examine the contents of objects in a container.
- The Instance window displays a single object. In the Instance window, you can inspect and modify the contents of the object.

DataView Window

A DataView window displays a representation of all the instances of a particular class. From the database panel, you display a DataView by double-clicking a class in the Classes tab or by double-clicking a root on the Roots tab.

A DataView window usually consists of the following three tabs:

- DataView
- Instance Format
- Filter

In addition, if the class displayed in your DataView tab implements the `com.odi.util.Indexed.Collection` interface, the DataView window automatically includes an Indexes tab.

You can have more than one DataView window open at a time.

DataView tab

The DataView tab contains a grid that displays objects in any container that supports the `java.util.Collection`, `java.util.Map`, or `com.odi.coll.Collection` interface, or a Java array.

The DataView tab consists of the following embedded panes:

- Container details
- Object details

The Container details pane contains a grid that displays the *default representation* of objects in a container. When you first load a database, the default representation consists of the name of the class to which the objects belong and the physical locations of the objects in the database in the ObjectStore *DSCO* format. The DSCO format specifies an object's database,

segment, cluster, and offset to identify its physical location in an ObjectStore or PSE Pro database. You can change the default representation by right-clicking the name of the class in the Classes tab of the database panel and selecting Set Class String Format from the shortcut menu. See Specifying the Class String Format on page 14 for more information.

The Object details pane displays the contents of the object that is currently highlighted in the Container details pane. The Object details pane has three columns that display the following information for each of the object's fields:

- Name
- Value
- Type

Instance Format tab

The Instance Format tab lets you customize the class information displayed in the grid on the Container details pane. You can select class fields and methods from the Instance Format tab. The names of the fields and methods are used for the column headings, and the values in the grid cells are the values of the fields or the values returned by the methods. See Customizing Your DataView on page 24 for more information about applying instance formats.

Filter tab

The Filter tab lets you create a query you can use to query the container that is displayed in the DataView tab. See Querying Containers on page 30 for more information about using the Filter tab.

Indexes tab

The Indexes tab appears whenever classes that support the `com.odi.util.IndexedCollection` interface are displayed in the DataView tab. It lets you add or remove indexes. See Creating an Index for a Container on page 31 for more information about how to add and remove an Index.

Instance Window

An Instance window displays the contents of a particular instance of a class. You display an Instance window by double-clicking an instance in the DataView tab of the DataView window or by double-clicking an object reference in another Instance window. You can also right-click an object reference and select Navigate from the shortcut menu to open another Instance window. The Java Browser mouse pointer changes to a hand pointer whenever it passes over an object reference.

If a database root points to a single user-defined object instead of a collection of objects, double-clicking the root in the Roots tab of the database panel will display the object in an Instance window instead of displaying it in a DataView window.

The Instance window displays the same information as the Object details pane does. However, you can open one Instance window for every instance in the DataView window and you can have any number of Instance windows open at a time.

You can make permanent changes to an object's values in an Instance window. See Changing an Object's Values on page 29 for more information.

Chapter 2
Working with Database Schema and Roots

This chapter describes the way the ObjectStore displays the schema of OSJI and PSE Pro databases. It also explains the different ways you can navigate through the classes stored in a database.

Contents

This chapter covers the following topics:

Working with Classes

The Classes tab of the database panel graphically depicts the classes that are stored in OSJI and PSE Pro databases. Collectively, these classes make up the database schema. Being able to visually inspect the contents of a database can help you with your application development.

You can quickly examine the contents of a database before and after you run an application, instead of writing code that reads the contents of the database out to a file.

Understanding the Tree Hierarchy

The tree hierarchy representing the database schema displays the classes that are stored in the database. This expandable and collapsible display starts at the top of the tree with the database and ends at the bottom with the class fields.

You expand each node in the tree hierarchy by clicking the key icon to the left of the node. A node is fully expanded when the key points down; a node is collapsed when the key points to the right.

From the Classes tab, you can display DataViews in the DataView window that show collections of all the objects of a given class that are contained in the database. These collections are called *class extents*. See Displaying a Class Extent on page 13 for a brief look at how to display class extents belonging to the class and Chapter 3, Working with Objects, on page 21 for a more complete description.

Understanding the Legend

As you navigate through the tree hierarchy, you can use the ObjectStore legend to help you understand the different parts of the database that you are traversing. The legend items are described in the table below. The legend is displayed at the bottom of the database panel.

Displaying this legend is optional. By default, the legend is turned on. You can turn off the legend by clicking the Show Legend check box directly below the database schema display.

Icon	*Meaning*
	Database.
	Package.
	Class.
	Package field — visible to all classes in the package.
	Public field — visible to any class.
	Private field — visible only within the class.
	Protected field — visible only within the class and subclasses.
	Field (a primitive type) is unknown because the class is not in your `CLASSPATH`.
	Relationship (a reference to an object) is unknown because the class is not in your `CLASSPATH`.

The following icon is displayed if the ObjectStore detects a problem with a database class:

	There is not a field attribute in the class, or the class is not in your `CLASSPATH`.

Displaying a Class Extent

A *class extent* is a collection of all the objects of a given class contained in a database. The extent is displayed in the Container details pane of a DataView.

To display the extent of a class shown in the tree hierarchy on the Classes tab:

1 Highlight a class on the Classes tab of the database panel.

2 Click  (Class Extent tool) on the Classes tab toolbar.

A DataView showing the class extent is displayed in the Workspace panel.

From a DataView you can examine the various class instances stored in the database, navigate to other objects, and modify data. For more information about DataViews and objects, see Chapter 3, Working with Objects, on page 21. See Specifying the Class String Format on page 14 for information on how to change the class information displayed in the class extent.

Specifying the Class String Format

When you display a class extent in a DataView, each object in the extent is listed in the Container details pane using the class's *default representation*. The first time you show a class extent, the default representation shows the name of the class to which the object belongs and the object's physical location in the database. You can change the representation of objects so that it lists the data contained in the class fields or returned by class methods. You do this by setting a *class string format* that specifies the class fields and methods for which you want to display data.

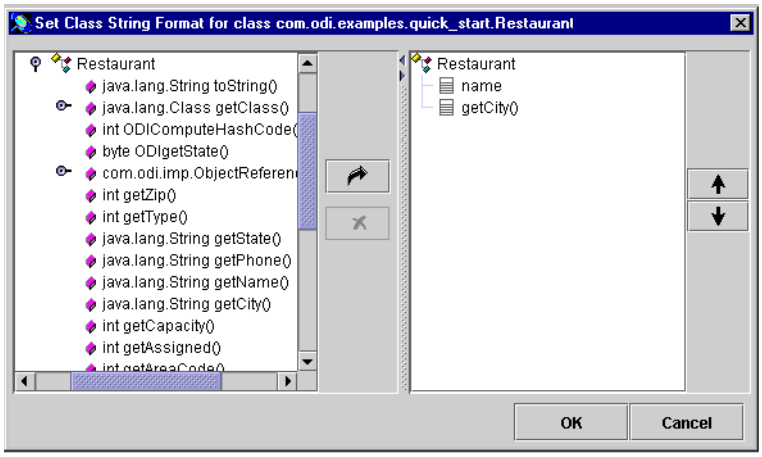
You can apply a class string format to any of the classes appearing in the Classes tab. The class string format also determines the information displayed in the title bars of Instance windows for the class.

Applying string formats

To create a string format for a class:

1 On the Classes tab of the database panel, right-click the name of a class and select Set Class String Format from the shortcut menu.

2 The Set Class String Format dialog box is displayed.



3 From the left pane of the dialog box, select a field or method for which you want to display data.

4 Click  .

The selected field or method is displayed in a hierarchical structure in the right pane.

5 Repeat steps 2 and 3 to select other fields or methods.

6 If you want, click  or  to rearrange the selected fields and methods in the right pane.

7 Click OK.

When you display the class extent of the class, the DataView will show data from the fields and methods you specified.

You do not need to calculate a class extent before applying the string formats; however, the string formats will not appear in the DataView until you calculate a class extent. See Displaying DataViews on page 22 for more information about calculating and displaying class extents.

For each class you can specify which fields and methods you want displayed with a string format. If you save the database metaknowledge, your choices for each class will be reflected the next time you load the database with the ObjectStore and display a DataView.

Modifying string formats

To remove fields or methods from string formats for a class:

1 On the Classes tab of the Database panel, right-click the name of a class and select Set Class String Format from the shortcut menu.

The Set Class String Format dialog box is displayed.

2 Select the class fields or methods from the right pane. You can press Shift+Click to select multiple class string formats.

3 Click  to remove the selected fields or methods.

4 Click OK.

When you remove fields and methods from the string format, data from the fields and methods will no longer be displayed in the Container details pane of the DataView window. However, the data in the object's fields will still appear in the Object details pane of the DataView window and in Instance windows.

Applying Class Filters to the Schema

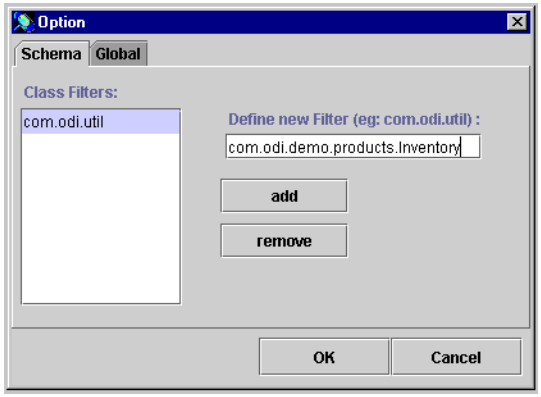
You can use class filters to specify classes you do not want to display in the database schema shown on the Classes tab. Class filters are helpful if you have a complex schema consisting of many classes. Displaying fewer classes lets you focus on the subset of classes in which you are most interested.

Applying a class filter

To apply a class filter:

1 Select Tools | Options from the menu bar.

The Option dialog box appears.



2 On the Schema tab of the Option dialog box, type the name of the class to be *excluded* from the graphical tree hierarchy in the Define new Filter text box.

3 Click Add.

4 Click OK.

The ObjectStore removes the class from the graphical tree hierarchy.

Removing a class filter

Removing a class filter *adds* the class to the graphical tree hierarchy. To remove a class filter:

1 Select Tools | Option from the menu bar.

The Option dialog box appears.

2 From the Class Filters list box, select the filter to be removed. .

3 Click Remove.

4 Click OK.

The ObjectStore adds the class to the graphical tree hierarchy.

Working with Database Roots

A *root* (also referred to as a *database root*) is a persistent object that serves as a starting point when you are navigating through objects in a database. A root is an ObjectStore object; the value of the root is a reference to some other object in the database. The object associated with the root can be a user-defined object or it can be a collection of objects.

This section describes how to use the Java Browser to browse, create, and destroy roots in the database. It also describes how to change a root's value.

After you open a database, you can begin working with roots. You do not need to calculate a class extent.

Note

If you save metaknowledge in your database, ObjectStore adds a new root, `_ODI_JAVA_BROWSER_`, in the Roots tab. This root references a persistent object that contains the database metaknowledge. If you save the database metaknowledge as a separate file, this root will not appear on the Roots tab. See Database Metaknowledge on page 3 for more information about metaknowledge.

Browsing a Root

You can browse a root and examine its contents.

To browse a root:

1 Click the Roots tab in the database panel.

2 Click the root that you want to browse.

3 Click  (Browse Root tool) on the Roots tab.

Alternative: On the Roots tab, right-click the name of the root and select Browse Root from the shortcut menu.

If the root is an object, an Instance window appears to display the root's contents. If the root is a container, then a DataView window appears.

Note You can have only one DataView of a root displayed at any time.

Creating a Root

You can create a new root for a database.

To add a root to a database:

1 Click the Roots tab in the database window.

2 Click  (Create Root tool) on the Roots toolbar.

When you first create a root, it has a default name and a value of null. You can edit the name of the root by selecting it and typing a new name.

You can assign a reference to another object in the database as the value of the root:

1 Display a DataView containing the object you want to assign to the root.

2 Right-click the object and select Copy from the shortcut menu.

3 On the Roots tab right-click the value field of the selected root and select Paste from the shortcut menu.

The new value for the root is displayed.

Destroying a Root

You can delete (destroy) a database root.

To destroy a database root:

1 Click the Roots tab in the database window.

2 Select the root you want to delete.

3 Click  (Destroy Root tool).

Alternative: Right-click the name of the root you want to delete and select Destroy Root from the shortcut menu.

The root and its value are deleted from the database.

When you delete a root, you make persistent data unreachable — applications that attempt to access the database by using that root will fail. Generally, you should never delete a root unless you are certain that you have no use for the values identified by it.

Warning

Deleting a root and deleting a root's values are operations with a high degree of risk. You should perform these operations only when you are certain of the following:

- Existing applications will not be affected negatively.
- The persistent data represented by the root values is no longer needed.

Changing a Root's Value

Since the value of a root is a reference to another object in the database, changing the root's value means the root will become associated with a different object.

You change the value of a root by copying and pasting an object reference to the new object:

1. Display a DataView that shows a reference to the new object you want to associate with the root.
2. Right-click the object in the Container details pane and select Copy from the shortcut menu.
3. On the Roots tab of the database panel, right-click the value of the root and select Paste from the shortcut menu.

See Changing an Object's Values on page 29 for more information about changing object references.

Consequences

When you redefine a root's value, applications can still access the database by using that root. However, the object previously associated with the root may no longer be accessible to those applications.

Chapter 3
Working with Objects

This chapter describes some of the typical tasks for which you can use the ObjectStore to help you in debugging, testing, and tuning your Java database application.

Contents

This chapter covers the following topics:

Containers and Objects

In the ObjectStore you can examine both collections of objects and individual objects. Collections of objects are displayed in DataView windows; individual objects are displayed in Instance windows.

You display a collection of objects of a particular class by calculating the class extent or by navigating from a root that contains a reference to a container object. You display an individual object by navigating to the object from a reference in a DataView, from a reference in another object, or from a root that contains a reference to a single object.

Displaying DataViews

A *DataView* is a window that displays the instances associated with a container. In the ObjectStore you can

- Create DataViews
- Customize DataView
- Print DataViews
- Refresh DataViews
- Save DataViews
- Reuse Dataviews
- Delete DataViews

Creating a DataView

There are two ways you can create a DataView:

- Navigate from a root.
- Calculate a class extent.

After you have created a DataView, you can make its default representation more readable by applying your own customizations. See Customizing Your DataView on page 24 for more information.

Navigating from a root

If a root in an OSJI or PSE Pro for Java database contains a reference to a container, you can display a DataView by navigating from the root:

1 Click the Roots tab in the database panel.

2 Click the root that you want to navigate from.

3 Click  (Browse Root tool) on the Roots tab.

Alternative: On the Roots tab, right-click the name of the root and select Browse Root from the shortcut menu.

A DataView window appears in the workspace panel.

Calculating a class extent

Before you can display and examine objects in an OSJI or PSE Pro for Java database, you must first calculate the class extent. A *class extent* is a container of all objects in the database that are instances of a particular class.

Once you calculate a class extent, you can then

- Create customized views of the objects in a database by applying string formats and filters.
- Execute a query against the class extent.
- Examine the contents of a particular object.
- Change the values of fields in objects.
- Find a particular object in the database.
- Invoke class methods.

To calculate a class extent:

1 Select File | Open from the menu bar.

The Open dialog box is displayed.

2 Select the database to open and click Open.

The database schema is displayed on the Classes tab of the database panel.

3 Select the class for which you want a class extent and click  (Class Extent tool).

Alternative: Right-click the class name and select Class Extent from the shortcut menu.

A DataView window appears in the workspace panel. The Container details pane on the DataView tab displays the default representation of the class extent that you just calculated.

While you are calculating a class extent that contains a large number of objects, the Stop tool () in the DataView window is enabled. You can click the Stop tool to stop the calculation.

Default DataView

A DataView displays the objects that make up a class extent; they are displayed in a grid in the Container details pane. The *default representation* of the DataView consists of the name of the class to which the objects belong and the physical locations of the objects in the database, for example, `(com.odi.demo.products.Product)<1|1|0|420>`. Physical locations are specified using the ObjectStore DSCO notation and the numbers represent the database, segment, cluster, and offset of an object.

You can override the default DataView by customizing it. The next section explains the way to do this.

A DataView also contains the Object details pane, which displays the contents of whichever object is highlighted in the Container details pane.

Customizing Your DataView

To make your default DataView more readable, you use the Instance Format tab of the DataView window to select which fields and methods you want to display. The names of the fields and methods are used for the column headings in the Container details grid and the values in the cells are the values of the fields or the values returned by the methods.

The customized display is called an *instance format* and applies only to the currently selected DataView.

Specifying a customized display

To apply an instance format to a DataView:

1. Click the Instance Format tab.
2. Select a field or method from the left pane.
3. Click  .

   Alternative: Double-click the element to add it to the instance format.
4. Repeat steps 2 and 3 to select additional fields and methods.
5. Click  or  if you want to rearrange the selected fields and methods.
6. Click Apply.

   The instance format is applied to the current DataView.

Removing fields or methods from the display

To remove fields or methods from the instance format for a DataView:

1. Click the Instance Format tab.
2. Select a field or method from the right pane.

3 Click  after each selection.

Alternative: Double-click the element to remove it from the instance format.

4 Repeat steps 2 and 3 for each field or method you want to delete.

5 Click Apply.

All selected fields and methods are removed from the currently selected DataView. If you remove all fields and methods, the instances will be displayed using the default representation for the class.

Printing a DataView

With the ObjectStore, you can print the data displayed in the Container details pane of a DataView.

To print a DataView:

1 Using the Instance Format tab of the DataView, specify the fields and methods you want to print and click Apply. See Customizing Your DataView on page 24 for more information on selecting fields and methods.

2 Display the DataView tab.

3 Click  (Print tool) on the DataView tab.

Refreshing Your DataView

As you work with a database over time, the contents of objects and the objects themselves change. Therefore, you might want to periodically refresh your DataView to reflect those changes.

To refresh a DataView:

1 Click the DataView tab.

2 Click  (Refresh tool).

Alternative: Press F5.

Saving a DataView

You can give names to your DataViews and save them so that you can quickly redisplay them. This is useful if you customize the instance format of the container or if the DataView shows the results of executing a query

against the database. When you save a DataView, you do not save the instances; rather, you save the customized display of fields and methods and the query.

You can save DataViews transiently or persistently. If you save them transiently, you can continue to use them while the database is open, but they disappear when you close the database. If you save DataViews persistently, they are saved as part of the database metaknowledge and you can use them when you open the database during another session

Saving a DataView transiently

To save a DataView transiently:

1 Click  (Save tool) on the DataView tab.

If the contents in a DataView have changed during the session, an asterisk (*) appears next to the name of the DataView.

If the Save tool is disabled, the DataView has not changed since it was last saved.

The ObjectStore prompts you with a dialog box that contains a suggested (default) name for the DataView.

2 Click OK to accept the default name, or type a different name for the DataView and then click OK.

The default names for DataViews are sequentially numbered within a session. This means that if you saved two DataViews in one database, and then you opened a different database to save another DataView, its default name would be DataView3.

Saving a DataView persistently

To save a DataView persistently:

1 Save the DataView transiently, as described above.

2 Select File | Save or File | Save All from the menu bar.

Alternative: Select File | Close or File | Exit, and click Yes to save the database metaknowledge.

If you have enabled the Save metaknowledge on file system option, the database metaknowledge is saved to a metaknowledge properties file in the user's home directory. If the Save metaknowledge on file system option is not enabled, this information is saved with the database. In the latter case, if the file has been opened for read-only, the metaknowledge is not saved when the database is closed.

To set the Save metaknowledge on the file system option:

1 Select Tools | Options from the menu bar.

The Option dialog box is displayed.

2 Click the Global tab.

3 Click the Save metaknowledge on the file system option to add a check mark.

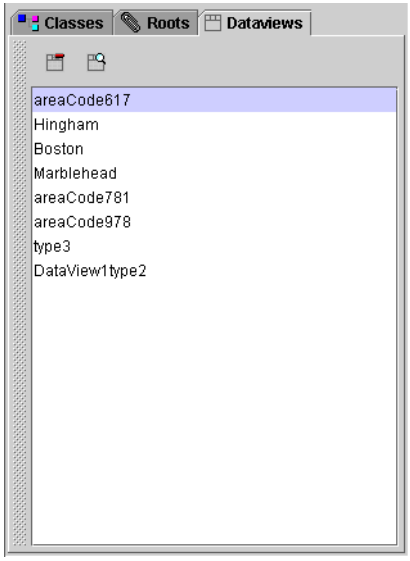
4 Click OK.

Reusing a DataView

You can reuse DataViews that have been saved. When you reuse a DataView, the Container details pane displays the objects using any customized display you may have applied to the DataView. If the DataView displays the results of a query, reusing it re-executes the query and displays the objects returned by the query.

To open a DataView and the metaknowledge that is associated with a database:

1 Click the Dataviews tab in the database panel.



2 Select the DataView you want to open from the list of DataViews.

3 Click  (Open DataView tool) to open the DataView.

Alternative: Double-click the DataView you want.

The DataView opens in the DataView tab of the DataView window.

Deleting a DataView

To delete a DataView from the metaknowledge that is associated with a database:

1 Click the DataViews tab in the database panel.

2 Select the DataView you want to delete from the list of DataViews in the DataViews tab.

3 Click  (Delete DataView tool) to delete the DataView.

A dialog box opens asking for confirmation before deleting the DataView.

4 Click Yes.

Displaying Objects

Once you create a DataView by calculating a class extent or by navigating from a root, you are ready to work with the objects in the database. To examine the contents of a particular object in the class extent, you can use either the Object details pane or the Instance window.

Viewing an Object in the Object Details Pane

The Object details pane is embedded in the DataView tab, below the Container details pane. To see it, drag up the dotted separator bar below the Container details pane or resize the DataView window.

The Object Details pane automatically displays the contents of the object that is currently selected in the DataView. This allows you to traverse the objects in a DataView and examine the contents of any object in the Container details pane.

Viewing an Object in the Instance Window

Another way to view the contents of an object is to use the Instance window. To access the Instance window, you can do one of the following:

- Double-click the object in the DataView.
- Right-click the object in the DataView and select Navigate from the shortcut menu.

- Double-click a root on the Roots tab if the root references a single user-defined object.

You use the Instance window and not the Object details pane when you want to change a value in an object. See Changing an Object's Values on page 29.

Navigating from One Object to Another

If an object contains a field whose value is an instance of another object, you can navigate directly to that object. When you move the mouse pointer over a reference to another object, the pointer changes to a hand.

To navigate to the referenced object, double-click the reference or right-click it and select Navigate from the shortcut menu.

When you navigate to an instance that has a reference to an instance in another database, that database is opened automatically for you. If you navigate to an object in another database, you will see a database number greater than `1` in the object's DSCO information.

Changing an Object's Values

At times, you might want to change the value of a field in an object, especially if the object is a root. Perhaps you want to test to see what your application will do if an object's field has a certain value. You use an Instance window to change the value of an object's fields. You cannot delete an object.

Primitive types

You can change a value in a field directly if the field is one of the following Java primitive types: `boolean`, `byte`, `char`, `double`, `float`, `int`, `long`, or `short`:

1. Display an Instance window for the object you want to change.
2. Click the value you want to change.

   A text box appears around the value.
3. Change the old value to the new one in the text box.

The new value is saved automatically and permanently in the database.

Object references

If the value in a field is a reference to an object, you can change the value to reference another object by copying and pasting a reference. For example, if a field contains a reference to `object_A` and you want to change it to reference `object_B`:

1 Display an Instance window for an object that contains a reference to `object_B`.

2 Right-click the reference and select Copy from the shortcut menu.

3 Display the Instance window for the object that contains the reference to `object_A`.

4 Right-click the field whose value you want to change and select Paste from the shortcut menu.

   The value of the field is now a reference to `object_B`.

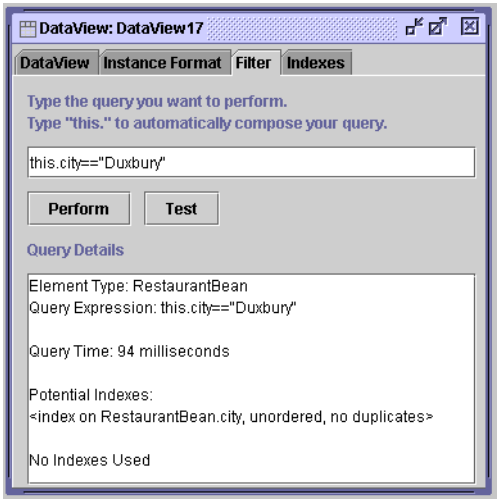
If the value of the field that you are changing is in a root, see Working with Database Roots on page 17 for more information.

Querying Containers

This section explains how to use the ObjectStore to query a container displayed in a DataView. After you execute a query, the DataView is updated to show only the results returned by the query. If you save a DataView after executing a query, the query string is saved as part of the DataView.

To query a container in a DataView tab:

1 Click the Filter tab in the DataView window.



2 Type the keyword `this` followed by a period (`this.`) in the text box.

When you type the period (.), a drop-down list box appears.

3 Select a field or method from the drop-down list box for your query and then press Enter.

4 Continue composing your query using the query syntax described in the *ObjectStore Java API User Guide* and the *PSE Pro API User Guide*.

5 Click Test for details about your query.

Information on how long it will take to execute the query, whether any indexes are used, or if indexes can be created is displayed in the Query Details box.

The query automatically uses any existing index on the container if the field being queried is in the index.

If the container supports the `com.odi.util.IndexedCollection` interface and if the query could benefit from using an index, the ObjectStore will suggest that an index be created. See Creating an Index for a Container on page 31 for more information.

6 Click Perform to execute the query.

The result is displayed in the Container details pane.

After you have executed a query, you can toggle between a filtered view (the results from applying a query to a container) and an unfiltered view of a container by clicking  (Toggle Filter tool) on the DataView tab.

You can also refresh and print the results of your query as you would with any other DataView by clicking  (Refresh tool) and  (Print tool).

Creating an Index for a Container

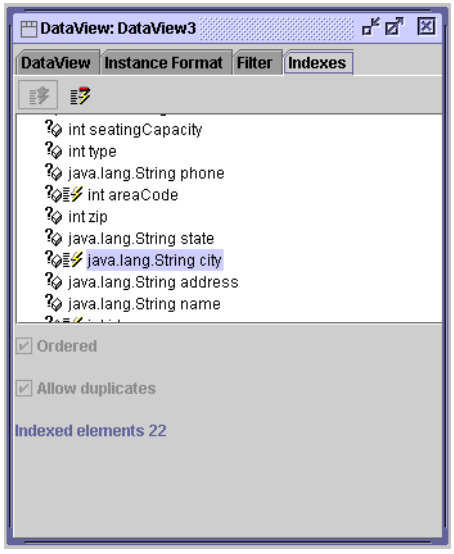
When you create a DataView for a container that supports indexes, an Indexes tab appears automatically in the DataView window. For example, by navigating from a root that points to an `OSTreeSet`, you can create such a DataView.

If you then create a query against the container and click the Test button on the Filter tab, the ObjectStore will suggest indexes that you can create to improve the performance of your query.

Adding an index

To create an index on a container displayed in a DataView:

1 Click the Indexes tab in the DataView window.



2 Select the method or field for which you want to add an index.

3 Click the Ordered check box if you want the elements in your index ordered.

4 Click the Allow Duplicates check box if you want duplicate elements in your index.

5 Click (Add Index tool).

When you add an index to a method or field, the Index icon appears before it on the Indexes tab. Also, the number of indexed elements appears at the bottom of the Indexes tab.

Removing an index

To delete an index from a container displayed in a DataView:

1 Click the Indexes tab.

2 Select the index you want to delete.

3 Click (Delete Index tool).

When you delete an index from a method or field, the Index icon is no longer displayed next to it on the Indexes tab.

Refreshing the Results from a Query

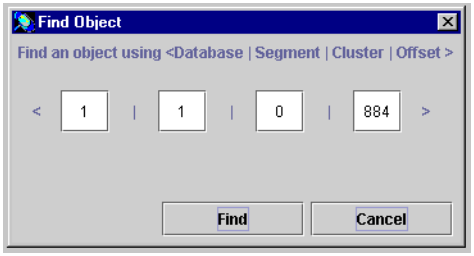
During a session, the results from a query might change over time. To refresh the results from a query that you ran previously, just click  (Refresh tool) on the DataView tab or press F5. The ObjectStore automatically reloads the DataView and reruns the query for you.

Finding an Object in a Database

You can use the Finder tool on the Classes tab toolbar to find a particular object in the database. You use the ObjectStore DSCO notation to specify the object's location.

To find a particular object in the database:

1 Click  (Finder tool) on the Classes tab toolbar. The Find Object dialog box appears.



2 Type the object's location in the boxes, using decimal numbers for the object's DSCO (database, segment, cluster, and offset) location in the database. For example, to find an object that has a location of `<1|1|0|884>` — which means the object is stored in the first database opened during the session, in the first segment, in the first cluster, at offset `884` in the database — you type the numbers `1`,`1`,`0`, and `884`.

3 Click Find.

An Instance window appears displaying the contents of the object that was found.

If the object that you find contains a reference to an object in another database, the ObjectStore automatically opens the second database for you when you navigate to the referenced object.

Exporting Data

You can export the data displayed in the Container details pane of a DataView window to an XML file. To specify which objects and which information to export you must customize the DataView. The ObjectStore exports only that information displayed in the Container details pane.

To export data to an XML file:

1 Display a DataView that contains the objects you want to export.

If you want, apply a filter to the class extent; the Browser will export the objects returned by the query. See Querying Containers on page 30 for information on how to query a class extent.

2 On the Instance Format tab, select the class fields and methods whose data you want to export. The Browser will export the data contained in these fields and methods. See Customizing Your DataView on page 24 for information on how to specify which fields and methods to display.

3 Right-click the DataView and select Export as XML from the shortcut menu.

4 In the Export Dataview as XML dialog box, give the file a name and specify the directory where you want to save the file.

5 Click the Save button.

You can click  (Stop button) to stop exporting the XML data.

Invoking Class Methods

As you are developing your application, it is helpful to test your class methods against the database to confirm that you are getting the results you expect. The ObjectStore can help you with this kind of testing.

You can use the ObjectStore to invoke class methods and display their results in the Container details pane of the DataView tab.

You can invoke only those methods that do not have parameters.

You can invoke class methods the following ways:

- Create a query that uses a method.

  You create queries that use methods the same way that you create queries against a container. The only difference is that you select a method instead

of a field from the drop-down list box on the Filter tab. See Querying Containers on page 30.

- In the Set Class String Format dialog box specify a method.

  When you apply the string format, the default representation of the DataView shows the results of the method. See Specifying the Class String Format on page 14.

- In the Instance Format tab of a DataView, specify a method.

  When you apply the instance format to the DataView, the results from the method are displayed in the Container details pane of the DataView and in the title bar of all Instance windows for that class. See Customizing Your DataView on page 24

Index

E

F

I

J

M

N

O

P

Q

R

S

T

U

W