Oracle9*i*AS TopLink

Getting Started

Release 2 (9.0.3)

August 2002

Part No. B10061-01

ORACLE®

Oracle9*i*AS TopLink Getting Started, Release 2 (9.0.3)

Part No. B10061-01

Contents

2 Installing and Testing TopLink

Glossary

Index

Send Us Your Comments

Oracle9*i*AS TopLink Getting Started, Release 2 (9.0.3)

Part No. B10061-01

Oracle Corporation welcomes your comments and suggestions on the quality and usefulness of this document. Your input is an important part of the information used for revision.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where?
- Are the examples correct? Do you need more examples?
- What features did you like most?

If you find any errors or have any other suggestions for improvement, please indicate the document title and part number, and the chapter, section, and page number (if available). You can send comments to us in the following ways:

- Electronic mail: iasdocs_us@oracle.com
- FAX: 650-506-7407 Attn: Oracle9*i* Application Server Documentation Manager
- Postal service:
  Oracle Corporation
  Oracle9*i* Application Server Documentation
  500 Oracle Parkway, M/S 2op3
  Redwood Shores, CA 94065
  USA

If you would like a reply, please give your name, address, telephone number, and (optionally) electronic mail address.

If you have problems with the software, please contact your local Oracle Support Services.

Preface

This document provides installation procedures to install and configure TopLink. It also introduces the concepts with which you should be familiar to get the most out of TopLink.

This preface contains the following topics:

- Intended Audience
- Documentation Accessibility
- Structure
- Related Documents
- Conventions

Intended Audience

This document is intended for new users who need to install and configure TopLink.

This document assumes that you are familiar with the concepts of object-oriented programming, the Enterprise JavaBeans (EJB) specification, and with your own particular Java development environment.

The document also assumes that you are familiar with your particular operating system (such as Windows, UNIX, or other). The general operation of any operating system is described in the user documentation for that system, and is not repeated in this manual.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible, with good usability, to the disabled community. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Standards will continue to evolve over time, and Oracle Corporation is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For additional information, visit the Oracle Accessibility Program Web site at `http://www.oracle.com/accessibility/`.

Accessibility of Code Examples in Documentation JAWS, a Windows screen reader, may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, JAWS may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation This documentation may contain links to Web sites of other companies or organizations that Oracle Corporation does not own or control. Oracle Corporation neither evaluates nor makes any representations regarding the accessibility of these Web sites.

Structure

This document includes the following chapters:

Chapter 1, "TopLink Concepts"

This chapter introduces the concepts with which you should be familiar to get the most out of TopLink.

Chapter 2, "Installing and Testing TopLink"

This chapter contains instructions for installing and testing TopLink.

"Glossary"

Provides definitions for words and phrases commonly used in TopLink.

Related Documents

For more information, see these Oracle resources:

Oracle9*i*AS TopLink Getting Started

Provides installation procedures to install and configure TopLink. It also introduces the concepts with which you should be familiar to get the most out of TopLink.

Oracle9*i*AS TopLink Tutorial

Provides tutorials illustrating the use of TopLink. It is written for developers who are familiar with the object-oriented programming and Java development environments.

Oracle9*i*AS TopLink Foundation Library Guide

Introduces TopLink and the concepts and techniques required to build an effective TopLink application. It also gives a brief overview of relational databases and describes who TopLink accesses relational databases from the object-oriented Java domain.

Oracle9*i*AS TopLink Mapping Workbench Reference Guide

Includes the concepts required for using the TopLink Mapping Workbench, a stand-alone application that creates and manages your descriptors and mappings for a project. This document includes information on each Mapping Workbench function and option and is written for developers who are familiar with the object-oriented programming and Java development environments.

Oracle9*i*AS TopLink Container Managed Persistence for Application Servers

Provides information on TopLink container-managed persistence (CMP) support for application servers. Oracle provides an individual document for each application server specifically supported by TopLink CMP.

Oracle9*i*AS TopLink Troubleshooting

Contains general information about TopLink's error handling strategy, the types of errors that can occur, and Frequently Asked Questions (FAQs). It also discusses troubleshooting procedures and provides a list of the exceptions that can occur, the most probable cause of the error condition, and the recommended action.

In North America, printed documentation is available for sale in the Oracle Store at

`http://oraclestore.oracle.com/`

Customers in Europe, the Middle East, and Africa (EMEA) can purchase documentation from

`http://www.oraclebookshop.com/`

Other customers can contact their Oracle representative to purchase printed documentation.

To download free release notes, installation documentation, white papers, or other collateral, please visit the Oracle Technology Network (OTN). You must register online before using OTN; registration is free and can be done at

`http://otn.oracle.com/admin/account/membership.html`

If you already have a username and password for OTN, then you can go directly to the documentation section of the OTN Web site at

`http://otn.oracle.com/docs/index.htm`

Conventions

Conventions in Text

We use various conventions in text to help you more quickly identify special terms. The following table describes those conventions and provides examples of their use.

Convention	Meaning	Example
Italics	Italic typeface indicates book titles or emphasis.	*Oracle9i Database Concepts* Ensure that the recovery catalog and target database do *not* reside on the same disk.

Convention	Meaning	Example
lowercase monospace (fixed-width) font	Lowercase monospace typeface indicates executables, filenames, directory names, and sample user-supplied elements. Such elements include computer and database names, net service names, and connect identifiers, as well as user-supplied database objects and structures, column names, packages and classes, usernames and roles, program units, and parameter values. **Note:** Some programmatic elements use a mixture of UPPERCASE and lowercase. Enter these elements as shown.	Enter `sqlplus` to open SQL*Plus. The password is specified in the `orapwd` file. Back up the datafiles and control files in the `/disk1/oracle/dbs` directory. The `department_id` and `location_id` columns are in the `hr.departments` table. Set the `QUERY_REWRITE_ENABLED` initialization parameter to `true.` Connect as `oe` user. The `JRepUtil` class implements these methods.
lowercase italic monospace (fixed-width) font	Lowercase italic monospace font represents placeholders or variables.	You can specify the *parallel_clause*. Run U*old_release*.SQL where *old_release* refers to the release you installed prior to upgrading.

Conventions in Code Examples

Code examples illustrate SQL, PL/SQL, SQL*Plus, or other command-line statements. They are displayed in a monospace (fixed-width) font and separated from normal text as shown in this example:

```
SELECT username FROM dba_users WHERE username = 'MIGRATE';
```

The following table describes typographic conventions used in code examples and provides examples of their use.

Convention	Meaning	Example
[ ]	Brackets enclose one or more optional items. Do not enter the brackets.	DECIMAL (*digits* [, *precision*])
{ }	Braces enclose two or more items, one of which is required.	{ENABLE | DISABLE}
|	A vertical bar represents a choice of two or more options within brackets or braces. Enter one of the options. Do not enter the vertical bar.	{ENABLE | DISABLE} [COMPRESS | NOCOMPRESS]

Convention	Meaning	Example

. . .	Horizontal ellipsis points indicate either:
	■ That we have omitted parts of the code that are not directly related to the example
	■ That you can repeat a portion of the code
		
		CREATE TABLE ... AS *subquery*;
		SELECT *col1*, *col2*, ... , *coln* FROM employees;

. . .	Vertical ellipsis points indicate that we have omitted several lines of code not directly related to the example.	

Other notation	You must enter symbols other than brackets, vertical bars, and ellipsis points as shown.	acctbal NUMBER(11,2);
		acct CONSTANT NUMBER(4) := 3;

Italics	Italicized text indicates placeholders or variables for which you must supply particular values.	CONNECT SYSTEM/*system_password* DB_NAME = *database_name*

Conventions for Microsoft Windows Operating Systems

The following table describes conventions for Microsoft Windows operating systems and provides examples of their use.

Convention	Meaning	Example
Choose **Start** >	How to start a program.	To start the Oracle Database Configuration Assistant, choose **Start** > **Programs** > **...** .
Case sensitivity and file and directory names	File and directory names are not case sensitive. The following special characters are not allowed: left angle bracket (<), right angle bracket (>), colon (:), double quotation marks ("), slash (/), pipe (|), and dash (-). The special character backslash (\) is treated as an element separator, even when it appears in quotes. If the file name begins with \\, then Windows assumes it uses the Universal Naming Convention. **IMPORTANT NOTE:** File names and directory names *are* case sensitive under UNIX. Where the name of a file or directory is mentioned and the operating system is a non-Windows platform, you must enter the names exactly as they appear unless instructed otherwise.	c:\winnt"\"system32 is the same as C:\WINNT\SYSTEM32

Convention	Meaning	Example
`C:\>`	Represents the Windows command prompt of the current hard disk drive. The escape character in a command prompt is the caret (^). Your prompt reflects the subdirectory in which you are working. Referred to as the *command prompt* in this manual.	`C:\oracle\oradata>`

	The backslash (\) special character is sometimes required as an escape character for the double quotation mark (") special character at the Windows command prompt. Parentheses and the single quotation mark (’) do not require an escape character. Refer to your Windows operating system documentation for more information on escape and special characters.	`C:\>exp scott/tiger TABLES=emp QUERY=\"WHERE job=’SALESMAN’ and sal<1600\"`
		`C:\>imp SYSTEM/`*`password`*` FROMUSER=scott TABLES=(emp, dept)`

`INSTALL_DIR`	Represents the Oracle home installation directory name. The home name can be up to 16 alphanumeric characters. The only special character allowed in the home name is the underscore.	`SET CLASSPATH=`*`INSTALL_DIR`*`\jre\bin`

Convention	Meaning	Example
ORACLE_HOME and *ORACLE_BASE*	In releases prior to Oracle8*i* release 8.1.3, when you installed Oracle components, all subdirectories were located under a top level *ORACLE_HOME* directory that by default used one of the following names: ■ `C:\orant` for Windows NT ■ `C:\orawin95` for Windows 95 ■ `C:\orawin98` for Windows 98 This release complies with Optimal Flexible Architecture (OFA) guidelines. All subdirectories are not under a top level *ORACLE_HOME* directory. There is a top level directory called *ORACLE_BASE* that by default is `C:\oracle`. If you install Oracle9*i* release 1 (9.0.1) on a computer with no other Oracle software installed, then the default setting for the first Oracle home directory is `C:\oracle\ora90`. The Oracle home directory is located directly under *ORACLE_BASE*. All directory path examples in this guide follow OFA conventions. Refer to *Oracle9i Database Getting Starting for Windows* for additional information about OFA compliances and for information about installing Oracle products in non-OFA compliant directories.	Go to the *ORACLE_BASE*`\`*ORACLE_HOME*`\rdbms\admin` directory.

1

TopLink Concepts

Oracle9*i*AS TopLink is a persistence framework that allows Java applications to access relational databases and non-relational data sources. TopLink maps objects and Enterprise Java Beans (EJBs) to the database in a non-intrusive manner and enables the developer to work at the object level.

Benefits of Using TopLink

TopLink enables you to map a Java object model and Enterprise Java Beans (EJBs) to a relational database and non-relational data sources. This bridges the gap between objects and the relations that exist among them and relational databases. Objects are a very flexible way of storing data and relationships between data, so representing objects in a relational database can be complicated.

With TopLink, you can:

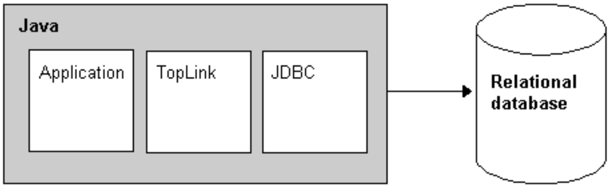
- Reduce development time and cost through using a proven industry-standard persistence framework.
- Develop high-performance and scalable applications using a rich set of proven performance and scalability features.
- Maintain platform independence through using a 100% pure Java product that can run in any Java environment or Java application server.
- Maintain data independence through using TopLink's object-level API and object-level querying.
- Access any database through a compliant JDBC driver and access non-relational data sources through TopLink SDK for Enterprise Information Systems.
- Integrate with leading-edge technology including EJB, XML, JTS, CORBA, and RMI.

TopLink Framework

The TopLink framework provides a flexible, easy-to-maintain mechanism for storing Java objects and Enterprise Java Beans (EJBs) in relational database tables.

Oracle9*i*AS TopLink is a layer of Java code between a Java application and the relational database (as illustrated in Figure 1–1). It acts as an object-oriented wrapper around the relational database and enables an application to map its objects into a series of relational tables.

Figure 1–1 The Oracle9iAS TopLink Framework



In addition to allowing applications to easily use relational databases, this model provides additional benefits:

- The wrapper pattern used by TopLink protects applications from changes in the underlying database. Therefore, changing the database has little effect on the code in an application.
- It provides a well-defined set of methods to store and retrieve data. This standardizes the database access methods of application classes and allows different classes to be used in similar ways.

Relational Databases

Relational databases can store sets of information that share some common characteristics in tables. You can then build and manipulate the various relationships between these tables.

TopLink enables your application to map its objects into a series of relational tables.

Tables, Records, and Fields

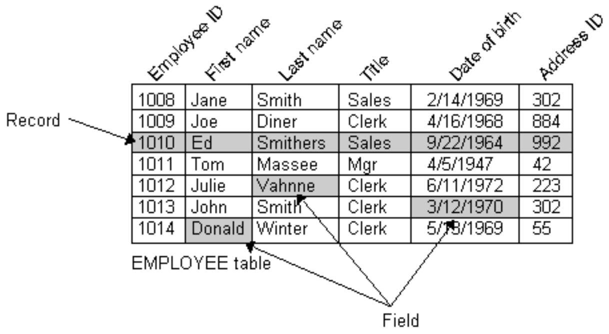
Many TopLink concepts are drawn from the common components of relational databases: tables, records, and fields.

- A *table* is a set of records. A relational database consists of one or more related tables.
- A *record* is a group of related fields treated as a unit in a table.
- A *field* is an individual piece of information in a record.

Fields organize the information in records. A table defines a standard field layout for all of its records. This layout defines both the type of information the field can contain (for example, strings, integers, float, date/time) and what that information represents (for example, employee name, department).

Figure 1–2 shows a simple `Employee` database table. Each row in the table is an employee record. Each column is a different field. Note that the field structure for all records is the same, and that each field can store different kinds of information (text, numbers, dates, etc.).

Figure 1–2 A Simple Database Table



Employee ID	First name	Last name	Title	Date of birth	Address ID
1008	Jane	Smith	Sales	2/14/1969	302
1009	Joe	Diner	Clerk	4/16/1968	884
1010	Ed	Smithers	Sales	9/22/1964	992
1011	Tom	Massee	Mgr	4/5/1947	42
1012	Julie	Vahnne	Clerk	6/11/1972	223
1013	John	Smith	Clerk	3/12/1970	302
1014	Donald	Winter	Clerk	5/18/1969	55

Keys (Primary and Foreign)

A relational database uses *keys* to relate the records in one table to records in another table. A key is a field (or combination of fields) that identifies a record or records in a table. TopLink implements the following keys:

- A *primary key* uniquely identifies a record. For example, the Employee ID column in the sample table in Figure 1–3.

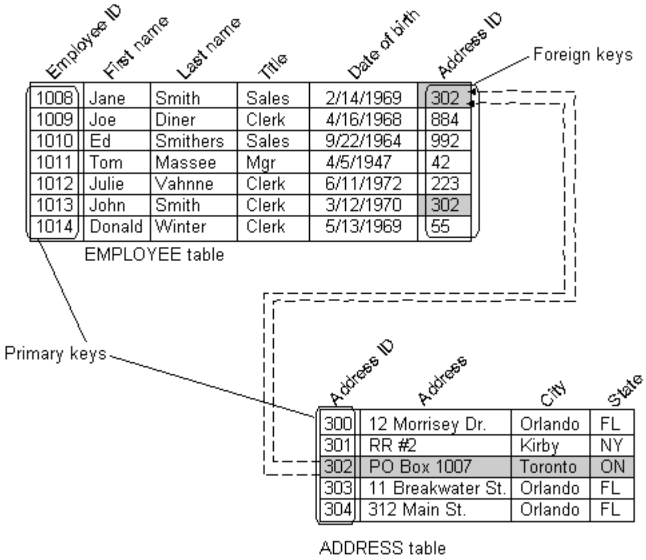
- A *foreign key* identifies a record in another table.

In the sample EMPLOYEE table in Figure 1–3, each employee record contains a foreign key reference to the employee’s home address (that is, Address ID). Notice that **John Smith** and **Jane Smith** both have references to the same home address.

Sequence Numbers

In a relational database, each record must be uniquely identified. Tables without a unique key can use *sequencing* to assign a unique ID to each record. For example, in the ADDRESS table in Figure 1–3, the primary key to the table is a series of sequential numbers (called *sequence numbers*) that identify the records being stored. Each time the application writes a new record to the database, the record receives a new sequence number (the existing value plus one).

Figure 1–3 Primary and Foreign Key Relationships in Tables



Employee ID	First name	Last name	Title	Date of birth	Address ID
1008	Jane	Smith	Sales	2/14/1969	302
1009	Joe	Diner	Clerk	4/16/1968	884
1010	Ed	Smithers	Sales	9/22/1964	992
1011	Tom	Massee	Mgr	4/5/1947	42
1012	Julie	Vahnne	Clerk	6/11/1972	223
1013	John	Smith	Clerk	3/12/1970	302
1014	Donald	Winter	Clerk	5/13/1969	55

Address ID	Address	City	State
300	12 Morrisey Dr.	Orlando	FL
301	RR #2	Kirby	NY
302	PO Box 1007	Toronto	ON
303	11 Breakwater St.	Orlando	FL
304	312 Main St.	Orlando	FL

Java Object Model

A Java object contains the following components:

- *Attributes* store primitive data such as integers, and also store simple Java types such as `String` and `Date`.
- *Relationships* are references to other TopLink-enabled classes. A TopLink-enabled class has a descriptor and can be stored in the database. Because TopLink-enabled classes can be stored in a database, they are called persistent classes.
- *Methods* are paths of execution that can be invoked in a Java environment. Because methods define behavior and are not state, they are not stored in the database.

Note: For concepts and more information on using TopLink with Enterprise JavaBeans (EJBs), refer to *Oracle9iAS TopLink Foundation Library Guide*.

Similarities Between Object and Relational Domains

Object and relational database domains contain the following similarities:

- *Classes* define the structure of objects in the same way that tables define the structure of their records.
- *Objects* store data in attributes the same way that records store data within a set of fields.
- Just as records in one table can reference records in other tables using foreign keys, objects can reference other objects using relationships.
- In most object-oriented languages such as Java, the attributes of a class are *statically typed*, just as the fields in a table are statically typed.

Storing Objects in a Relational Database

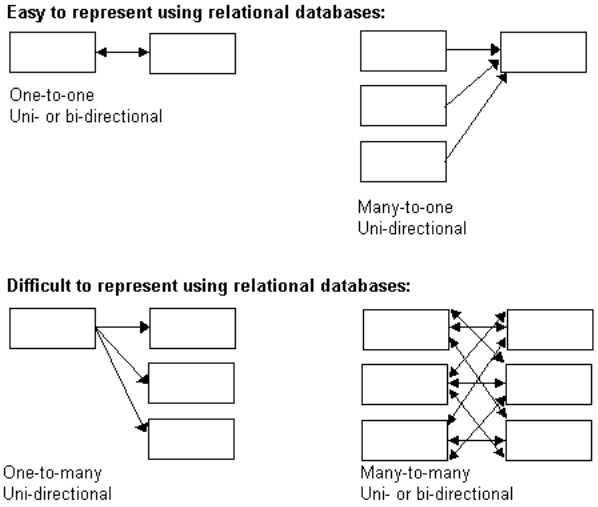
Object technology introduces some concepts that may be difficult to model using relational databases, such as:

- *Inheritance* and *polymorphism* — In Java, inheritance and polymorphism allow an application to work with related classes of objects in a generic fashion. Often the application does not know what classes of objects it is using. It is not easy to implement this level of abstraction in relational databases, because all records in a table must have exactly the same field structure, and foreign keys normally refer to a single table implicitly.

- *Complex relationships* — Relational databases are very good at representing one-to-one and many-to-one unidirectional relationships. However, the more complicated bidirectional and many-to-many relationships, common in Java object models, are much more difficult to represent in relational databases.

TopLink handles all of these concepts and relationships.

Figure 1–4 Common Java Object Relationships

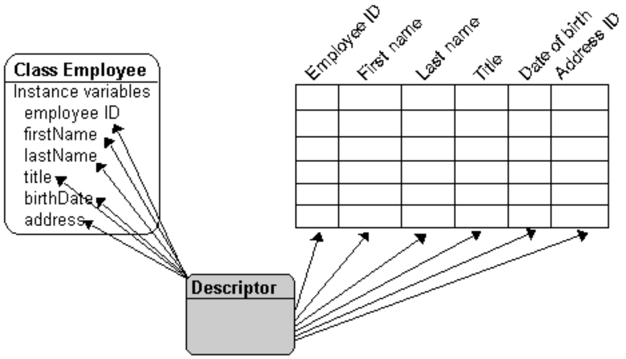


TopLink Descriptors

TopLink uses *descriptors* to take advantage of the similarities between relational databases and objects while accounting for their differences.

A *descriptor* is a set of properties and mappings that describes how an object's data is represented in a relational database. The descriptor contains mappings from the class attributes to the table columns, as well as the transformation routines necessary for storing and retrieving attributes. The descriptor links the Java object to its database representation, as it appears in Figure 1–5.

Figure 1–5 TopLink Descriptor



Every TopLink descriptor is initialized with the following information:

- The Java class it describes, and the corresponding table(s) for storing instances of that class
- The primary key of the table
- A list of query keys (or aliases) for field names
- A description of the object's attributes and relationships; this information is stored in *mappings* (see "Mappings" on page 1-7)
- A set of properties for tailoring the behavior of the descriptor

Mappings

The *mappings* stored in each descriptor define the storage and retrieval of an object's attributes. TopLink uses two types of mappings:

- *Relationship mappings* represent references to other TopLink-enabled objects.
- *Direct mappings* encompass everything else to be stored in an object's table.

Relationship Mappings

Relationship mappings map relationships between TopLink-enabled objects. There are three main types of relationship mappings: *one-to-one*, *one-to-many*, and *many-to-many*. One-to-one mappings store the link from one TopLink-enabled class to another. One-to-many and many-to-many mappings store collections of references to other TopLink-enabled classes.

Direct Mappings

Direct mappings map attributes. There are two types of direct mappings: *direct-to-field* and *transformation type*. Direct-to-field mappings are the simplest to use; they store the attribute in the database table in the field's native format. To map attributes that are not supported by direct-to-field mappings, you must use a transformation type mapping.

Transformation type mappings transform the data from a native Java format to one supported by the relational database. Four transformation type mappings are available: *object type mappings*, *type conversion mappings*, *serialized object mappings*, and *transformation mappings*.

For More Information

To learn how to build simple descriptors, refer to the tutorials and demo programs in the *Oracle9iAS TopLink Tutorials*. After completing the tutorial, refer to the *Oracle9iAS TopLink Mapping Workbench Reference Guide* for more information about the topics discussed in the tutorial.

Database Sessions

A *database session* in TopLink represents an application's dialog with a relational database. The `DatabaseSession` class keeps track of the following information:

- Instances of `Project` and `DatabaseLogin`, which store database login and configuration information
- An instance of `DatabaseAccessor`, which wraps the JDBC connection and handles database access
- The descriptors for each of the application's persistent classes
- The identity maps, which maintain object identity and act as a cache

An application uses the session to login to the database and perform read and write operations on the objects stored therein. The session's lifetime is normally the same as that of the application.

Using TopLink in an Application

After the descriptors have been created, you must write Java code to register the descriptors with the TopLink session. After registering the descriptors, the application is ready to read and write Java objects from the database:

- To read objects from the database, use the database session object
- To write objects to the database, use a unit of work object

Transactions

A *transaction* is a set of database operations that can either be committed (accepted) or rolled back (undone). Transactions can be as simple as inserting an object into a database, but also allow complex operations to be committed or rolled back as a single unit. Unsuccessful transactions can be discarded, leaving the database in its original state.

Unit of Work

A *unit of work* is an object that unifies the database transaction with the changes to the Java objects, defining an object-level transaction. The unit of work enhances database commit performance by updating only the changed portions of an object. Units of work are the preferred method of writing to a database in TopLink.

Reading and Writing Java Objects

Sessions can read objects from the database using the `readObject()` method. Database sessions can write objects to the database using the `writeObject()` method, but note that this method is neither required nor used when using a unit of work. An application typically uses the session to read the instances of a given class from the database and determines which of the instances require changes. The instances requiring changes are then registered with a unit of work. After the changes have been made, the unit of work is used to commit only the changed objects to the database.

This model provides the optimum performance for most applications. Read performance is optimized by using the session because the unit of work does not have to keep track of objects that do not change. Write performance is optimized because the unit of work keeps track of transaction information and writes only the changed portions of an instance to the database.

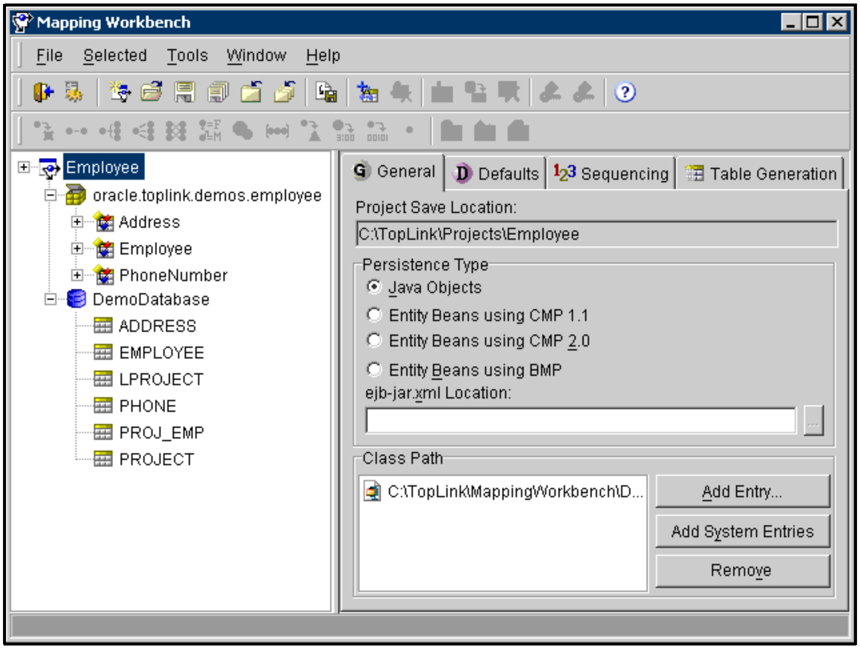
TopLink Mapping Workbench

TopLink Mapping Workbench is a separate tool that provides a graphical method of configuring the descriptors and mappings of a project. It provides many checks to ensure that the descriptor settings are valid, and it also provides advanced functionality for accessing the database and creating a database schema.

TopLink Mapping Workbench does not generate Java code during development, which would be unmanageable if the descriptors changed. Instead, it stores descriptor information in an XML deployment file, which can be read into a Java application using a TopLink method. To support the development of a run-time application, TopLink can generate a `.java` file from the XML file, eliminating the need for TopLink Mapping Workbench files at run time.

The TopLink Mapping Workbench displays all of the project information for a given project, including classes and tables.

Figure 1–6 TopLink Mapping Workbench



Project Save Location: C:\TopLink\Projects\Employee

Persistence Type

● Java Objects

○ Entity Beans using CMP 1.1

○ Entity Beans using CMP 2.0

○ Entity Beans using BMP

ejb-jar.xml Location:

Class Path

C:\TopLink\MappingWorkbench\D...

Refer to *Oracle9iAS TopLink Mapping Workbench Reference Guide* for more information on editing projects and descriptors using the Mapping Workbench.

Design Strategies and Tips

To ensure the best design for your TopLink application, you should follow an iterative step-by-step development process and consider certain database issues, as illustrated in Figure 1–7.

To Design a TopLink Application:

1. Create the object model for the application. Define the object model *before* using Oracle9*i*AS TopLink to map objects. This is very important, because defining persistent mappings for an incorrect or rapidly changing model can be very difficult.

2. Decide how the classes should be implemented in the database. If the application must be integrated with a legacy system, decide how the classes relate to the tables already created. If there is no legacy database to integrate, decide how each class should be stored in the database, and create the database schema. Alternatively, you may use Oracle9*i*AS TopLink to create your initial tables.

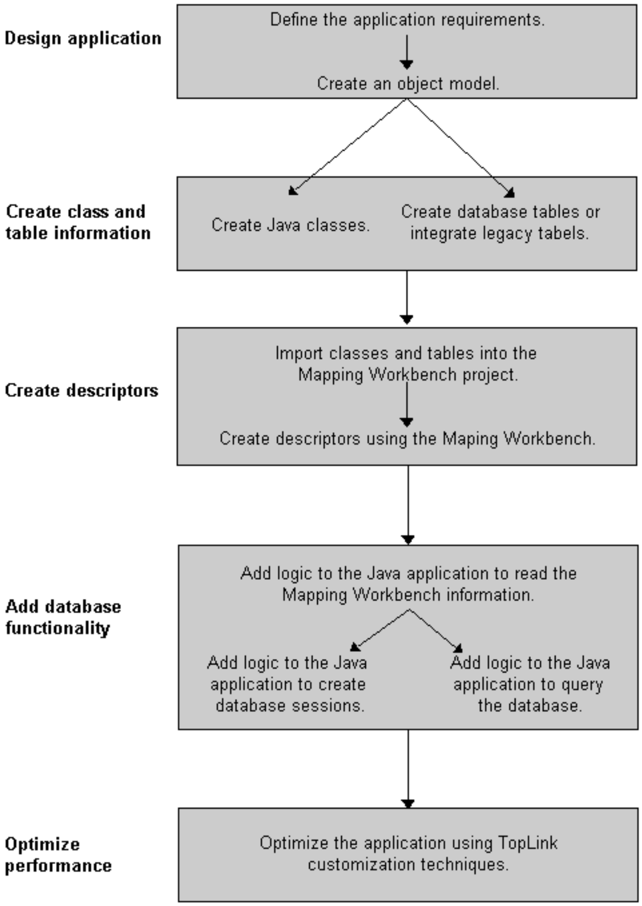
3. Use Oracle9*i*AS TopLink Mapping Workbench to create descriptors and mappings for each of the persistent classes.

> **Tip:** Avoid building all of your model's descriptors in a single iteration. Start with a small subset of your classes, build and test their descriptors, and then gradually add new descriptors and relationships. This enables you to catch common problems before they proliferate through your entire TopLink design.

4. Write Java code to use database sessions. Sessions are used to query for database objects and write objects to the database.

5. Optimize the application. This may mean using advanced TopLink features or writing custom querying routines to access the database in specific ways.

Figure 1–7 Developing a Typical TopLink-enabled Application


Design application
Define the application requirements.
Create an object model.
Create class and table information
Create Java classes.
Create database tables or integrate legacy tabels.
Create descriptors
Import classes and tables into the Mapping Workbench project.
Create descriptors using the Maping Workbench.
Add database functionality
Add logic to the Java application to read the Mapping Workbench information.
Add logic to the Java application to create database sessions.
Add logic to the Java application to query the database.
Optimize performance
Optimize the application using TopLink customization techniques.

Database Considerations

One of the most important decisions when designing a TopLink application is relating Java classes to database information. Although some designers would say that the application’s object model should be designed first, and its design should drive the specification of the database schema, there are some compelling reasons why the relational database schema should be considered during the design of the object model:

1. The design of the TopLink-accessed database is directly dependent on the application’s object model.

   Each decision regarding the attributes and relationships of and between objects affects the database implementation. These two design models are inter-related and thus, must be considered together.

2. Application design decisions are often made arbitrarily.

   As shown in *Oracle9iAS TopLink Foundation Library Guide*, the impact of an otherwise insignificant decision may greatly affect the performance of the database.

3. Relational databases have limitations that impose certain restrictions on the design of the application.

   The application designer can save time by considering the effects of application design decisions on the design of the database.

4. Considering the database keeps the application model simple, which results in much better performance.

   Application designers often opt for maximum re-use when designing new applications. This sometimes results in unnecessarily fragmented object models, in which each bit of functionality is factored into a tiny reusable object. Designing and implementing a database whose schema is based on a fragmented object model can result in extremely inefficient application performance.

Do Not Design in Isolation

Do not let the design of the database drive the design of the object model, but do not design the two in isolation either. For best results, consider the database *during* the design of the application rather than afterwards.

2

Installing and Testing TopLink

This chapter includes the following information on installing TopLink:

- System Requirements
- Installing TopLink
- Third-party License Information
- Configuring TopLink for Your Development Environment
- Testing Your TopLink Installation
- General Troubleshooting

System Requirements

Your system must meet certain minimum requirements to ensure that TopLink runs smoothly.

Operating Systems Requirements

TopLink will run under any of the following operating systems:

- Windows 2000
- Windows NT 4.0 (service pack 6) or newer
- Any fully Java-compatible Unix operating system

Hardware Requirements

TopLink hardware requirements are quite low, and in fact, the hardware required to run both Java and most database applications far exceeds TopLink's requirements.

However, as a guideline, your system should include at least the following to ensure the proper operation of TopLink:

- Pentium®-class processor running at a minimum of 300 MHz
- 192 MB of random access memory (RAM)
- 128 MB free on your hard drive
- Monitor resolution of 1024x768

Software Requirements

TopLink requires the following supporting software:

- a JDBC driver configured to connect with your local database system (see your database administrator for details)
- a Java development environment compatible with the JDBC API, such as:
  - Oracle JDeveloper
  - Sun JDK 1.2 or JDK 1.3
  - IBM VisualAge for Java
  - Borland JBuilder
  - Microsoft Visual J++

  Or any other Java environment compatible with the Sun JDK 1.2 (or higher).

Installing TopLink

Use this procedure to install Oracle9*i*AS TopLink (including the Foundation Library and Mapping Workbench).

> **Note:** If you are installing under Windows NT/2000, you must have administrator privileges. Also, ensure you modify the **System Variables**, not the **User Variables**.
>
> Java package names are case-sensitive. If you are installing under a 32-bit Windows environment, make sure the case sensitivity is turned on.

Before installing TopLink, you should back up all existing project data.

To Install TopLink on Windows:

1. Download the software or insert the CD-ROM.

   If you downloaded the software, you must un-compress the installer.

2. Follow the instructions in the installer.

3. Refer to `<`*`INSTALL_DIR`*`>\doc\index.html` for the latest release notes.

   If you installed to the default directory, `<`*`INSTALL_DIR`*`>` is `C:\<`*`ORACLE_HOME`*`>\toplink`.

4. Check and if necessary, modify the CLASSPATH environment variable. See "Modifying the PATH and CLASSPATH Environment Variables" on page 2-4 for more information.

> **Caution:** In this release, the package structure has been changed to conform with Oracle9*i* standards and conventions for packages. The base package for the entire structure is now **oracle.toplink**.
>
> To facilitate the migration to the new package names, Toplink provides a tool to help you migrate to the new names, if you are upgrading from 2.x, 3.x or 4.6 to Oracle9*i*AS TopLink Release 2 (9.0.3). Consult the Package Renamer for more information.

To Install TopLink on Unix or other non-Windows Operating System:

1. Download the software or insert the CD-ROM.
2. Un-compress the files to a directory on your local file system. This directory is `<INSTALL_DIR>`.
3. Refer to `<INSTALL_DIR>/docs/index.html` for the latest release notes.

   If you installed to the default directory, `<INSTALL_DIR>` is `/<ORACLE_HOME>/toplink`.
4. Check and if necessary, modify the CLASSPATH environment variable. See "Modifying the PATH and CLASSPATH Environment Variables" on page 2-4 for more information.

> **Caution:** In this release, the package structure has been changed to conform with Oracle9*i* standards and conventions for packages. The base package for the entire structure is now **oracle.toplink**.
>
> To facilitate the migration to the new package names, Toplink provides a tool to help you migrate to the new names, if you are upgrading from 2.x, 3.x or 4.6 to Oracle9*i*AS TopLink Release 2 (9.0.3). Consult the Package Renamer for more information.

Modifying the PATH and CLASSPATH Environment Variables

You must properly configure your PATH and CLASSPATH environment variables before using TopLink. If the Mapping Workbench is open, close the Mapping Workbench and re-open it after modifying these settings.

> **Note:** Your individual user variables may override the system variables. Contact your system administrator for more information on setting user and system environment variables.

Use these procedures to configure your PATH and CLASSPATH environment variables for Windows Environment or a Non-Windows Environment.

Windows Environment

Add the following to the beginning of your PATH:

```
<JAVADIR>\bin\
```

where *<JAVADIR>* is the directory of your Java run-time.

Add the following to the beginning of your CLASSPATH:

```
<INSTALL_DIR>\core\lib\toplink.jar
<INSTALL_DIR>\core\lib\xerces.jar
<INSTALL_DIR>\core\lib\antlr.jar
```

where *<INSTALL_DIR>* is the directory which contains TopLink (for example, `C:\<ORACLE_HOME>\toplink`).

Non-Windows Environment

Add the following to the beginning of your PATH:

```
<JAVADIR>\bin\
```

where *<JAVADIR>* is the directory of your Java run-time.

Add the following to the beginning of your CLASSPATH:

```
<INSTALL_DIR>/core/lib/toplink.jar
<INSTALL_DIR>/core/lib/xerces.jar
<INSTALL_DIR>/core/lib/antlr.jar
```

where *<INSTALL_DIR>* is the directory which contains TopLink (for example, `/<ORACLE_HOME>/toplink`).

Third-party License Information

TopLink uses the following software:

ANT

jakarta.apache.org/ant/index.html

Antlr

www.antlr.org/rights.html

TopLink uses Antlr for EJBQL parsing. Antlr (ANother Tool for Language Recognition), is a language tool that provides a framework for constructing recognizers, compilers, and translators from grammatical descriptions containing C++ or Java actions. The ANTLR parser and translator generator is fully in the public domain.

DOM

www.w3.org/Consortium/Legal/copyright-software.html

The Document Object Model (DOM) is a specification that defines some programming language-neutral interfaces that can be used to manipulate XML and HTML documents. W3C maintains this specification. W3C also provides a Java "binding" for these interfaces. TopLink uses this "binding" to parse and manipulate XML documents.

Hypersonic SQL

hsqldb.sourceforge.net/2/General/license.html

This product includes Hypersonic SQL. The HSQL database shipped with the TopLink examples and demos is for development purposes only. Redistributions of source code must retain the copyright notice set forth below, this list of conditions and the following disclaimer. Redistributions in binary form must reproduce the copyright notice set forth below, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

- All advertising materials mentioning features or use of this software must display the following acknowledgment: "This product includes Hypersonic SQL."

- Products derived from this software may not be called “Hypersonic SQL” nor may “Hypersonic SQL” appear in their names without prior written permission of the Hypersonic SQL Group.
- Redistributions of any form whatsoever must retain the following acknowledgment: “This product includes Hypersonic SQL.”

The HSQL software is provided “as is” and any expressed or implied warranties, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose are disclaimed. In no event shall the Hypersonic SQL Group or its contributors be liable for any direct, indirect, incidental, special, exemplary, or consequential damages (including, but not limited to, procurement of substitute goods or services; loss of use, data, or profits; or business interruption). However caused any on any theory of liability, whether in contract, strict liability, or tort (including negligence or otherwise) arising in any way out of the use of this software, even if advised of the possibility of such damage. Copyright © 1995-2000 by the Hypersonic SQL Group. All rights reserved.

Java Runtime Environment v 1.3.1

`java.sun.com/j2se/1.3`

The Java Runtime Environment is maintained by Sun Microsystems, Inc. The TopLink Mapping Workbench runs in a Windows JDK 1.3.1_03 VM by default, and may also be configured to run using other compliant Java 2 VMs. Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc., in the U.S. and other countries.

SAX

`sax.sourceforge.net`

The Simple API for XML (SAX) is an interface for event-based XML parsing, developed collaboratively by the members of the XML-DEV mailing list. TopLink uses this API to interact with the Apache Xerces parser.

xerces.jar

`www.apache.org`

This product includes software developed by the Apache Software Foundation (`http://www.apache.org/`). Copyright © 2000 The Apache Software Foundation. All rights reserved.

1. Redistributions of source code must retain the copyright notice, this list of conditions and the following disclaimer.

2. The end-user documentation included with the redistribution, if any, must include the following acknowledgment: "This product includes software developed by the Apache Software Foundation (`http://www.apache.org/`)." Alternately, this acknowledgment may appear in the software itself, if and wherever such third-party acknowledgments normally appear.
3. The names "Apache" and "Apache Software Foundation" must not be used to endorse or promote products derived from this software without prior written permission.
4. Products derived from this software may not be called "Apache", nor may "Apache" appear in their name, without prior written permission of the Apache Software Foundation.

THIS SOFTWARE IS PROVIDED "AS IS" AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE APACHE SOFTWARE FOUNDATION OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Configuring TopLink for Your Development Environment

This section contains detailed information on configuring TopLink for your specific development environment.

Sun JDK and JRE

By default, TopLink includes Java Runtime Environment 1.3.1_03 from Sun Microsystems for use by the TopLink Mapping Workbench.

To configure TopLink for a different version of the JRE or a JDK installation, change the `setenv.cmd`, at the root of *`<INSTALL_DIR>`* to point `JAVA_HOME` to the new install directory.

For development purposes with the TopLink Foundation Library or to run the TopLink Examples, you must have either JDK 1.2 or 1.3 installed. Ensure the `setenv.cmd`, at the root of *<INSTALL_DIR>* points `JAVA_HOME` to the JDK installation.

IBM VisualAge for Java

When using TopLink with VisualAge, you can speed the development of an application by:

- Importing TopLink Classes
- Importing TopLink Source Files

Importing TopLink Classes

To use TopLink inside the VisualAge environment you must import the TopLink classes.

To Import TopLink Classes:

1. Choose **File > Import**.
2. Select the `toplink.jar` file in the *<INSTALL_DIR>*`\core\lib\` .
3. Select a target project. For a new project, type the name in the **Project** field.
4. Click **Finish**.

Repeat for any other `.jars` that you want to import into VisualAge.

Importing TopLink Source Files

You can import TopLink source files into the VisualAge environment to provide partial source for a project.

> **Note:** You must import the TopLink `.jar` files that contain the class files before importing the related TopLink source files. See "Importing TopLink Classes" on page 2-9 for more information.

To import the TopLink source.jar

1. Choose **File > Import**.
2. Select the `source.jar` file.
3. Specify *<INSTALL_DIR>*`\core\lib\source.jar` as the file name.
4. Select the **Java Source** check box to indicate that you are importing java source files rather than class files.
5. Select the target project into which you imported the related TopLink classes.
6. Click **Finish**.

Make sure your runnable classes using TopLink include the TopLink projects on their CLASSPATH.

> **Note:** If you are using an earlier version of IBM VisualAge for Java, import the TopLink `.jar` files into a new TopLink project.

Testing Your TopLink Installation

Compile and execute the `ConnectTest` class in the `oracle.toplink.tutorials.gettingstarted` package.

To Test the Installation:

Run the `ConnectTest` class from the command line, passing appropriate database login information as parameters as follows:

```
java oracle.toplink.tutorials.gettingstarted.ConnectTest <username> <password>
<database url> <jdbc driver class>
```

or

Modify the `main()` method to contain appropriate database login information as parameters, as in the following example, then recompile and execute the class.

```
public static void main(String[] args)

{
  ConnectTest test = new ConnectTest();
  if (args.length > 0) {
    test.connect(args[0], args[1], args[2], args[3], args[4]);
  }
  // Add your login information below
  else {
    test.connect("<user>","<password>", "oracle.jdbc.driver.OracleDriver",
    "jdbc:oracle:thin:@oraserver:1521:orcl";
  }
}
```

If the code does not run successfully, check the error message and ensure that all of your settings are correct. If necessary, refer to "General Troubleshooting" on page 2-11 for more information.

After installing and testing TopLink, refer to the *Oracle9iAS TopLink Tutorial Guide* for information on the tutorials that will help you learn about TopLink.

General Troubleshooting

After installing TopLink, if you encounter problems either starting the application or connecting to a database, try the following solutions:

- Check the `jdbc.log` file located in the installation folder for the SQL exception being generated.
- Ensure that the driver class name is correct. Many vendors have several different driver classes to choose from.
- Check your login information.
- Ensure that your PATH includes all `.dll` files required by your driver.
- Ensure that your CLASSPATH includes all classes or `.zip/.jar` files required by your JDBC driver.
- Check with your database administrator that:
  - Any drivers that require special set-up in the database server (such as Sybase JConnect) have been set up correctly.
  - Any drivers that require special permissions in the database server have been set up correctly.
  - You are not exceeding the number of available concurrent connections to your database. This may occur during development time, when many people are testing connections.
- If you are using a database servers that requires an extra Windows NT service to be running for JDBC connections, ensure that one is running

- Check with your vendor to ensure that you are using the latest version of both your JDBC driver and the database to which it is connecting.

Refer to the *Oracle9iAS TopLink Troubleshooting Guide* for more information.

Glossary

This glossary contains terms and abbreviations that you should be familiar with when using TopLink.

Application Programming Interface (API)

The specification of how a programmer writing an application accesses the behavior and state of classes and objects. A set of classes that define services for a programmer.

application server

A program whose sole purpose is to provide services to other programs called clients

attribute

A variable of a class or object. In TopLink, *attribute* describes all instance variables of a class. Every attribute contains a single mapping.

bean

A reusable software component that can be combined to create an application. Usually, beans are bundled up with their corresponding `BeanInfo` classes into `.jar` files.

See also JAR.

bean class

The implementation of the bean. The bean is accessed from the client using the home and remote interfaces.

See also home interface and remote interface.

BMP

Bean-Managed Persistence, the bean developer is responsible for storing and retrieving the entity bean.

Compare to CMP.

branch class

Has a persistent superclass and also has subclasses. By default, queries performed on the branch class return instances of the branch class and any of its subclasses. However, the branch class can be configured so that queries on it return only instances of itself without instances of its subclasses.

Compare to leaf class and root class.

class

A category of objects. Classes allow data and method to be grouped together.

class indicator field

A field in the table of the root class that indicates which subclass should be instantiated

CLASSPATH

The search path that Java uses to locate class files on a hard drive

CMP

Container-Managed Persistence, the container is responsible for storing and retrieving the entity bean.

Compare to BMP.

Common Object Request Broker Architecture (CORBA)

Platform-independent technique for programs running on different machines to communicate with each other by creating contracts between objects for implementation as distributed applications.

custom SQL

Refers to any non-TopLink-generated SQL used through TopLink. This includes hard-coded SQL and stored procedure calls.

data definition language (DDL)

The data definition part of the structured query language (SQL). TopLink Mapping Workbench can generate DDL creation scripts that can be used to create tables on the desired database.

database

A large collection of data organized for rapid search and retrieval.

dependent classpath (WebSphere)

Location where non-bean classes are specified. TopLink requires that the bean classes be included here since they are referenced by the project.

deployment descriptor (EJB)

Provides information required by the server to deploy the bean. An “EJB deployment descriptor” is not the same as a “TopLink descriptor.”

See also descriptors.

descriptors

A TopLink object that describes how an object’s attributes and relationships are to be represented in relational database table(s). A “TopLink descriptor” is not the same as an “EJB deployment descriptor”, although it plays a similar role.

Descriptor Editor

A TopLink Mapping Workbench feature used for creating descriptors and mappings.

direct access

By default, TopLink accesses public attributes directly when writing the attributes of the object to the database or reading the attributes of the object from the database.

Compare to method access.

direct mapping

There are two basic ways of storing object attributes directly in a table:

- The information can be stored directly if the attribute type is comparable to a database type.
- If there is no database primitive type that is logically comparable to the attribute’s type, it must be transformed on its way to and from the database

TopLink provides five classes of direct mappings.

Compare to relationship mapping.

expressions

The TopLink equivalent of an SQL conditional clause. TopLink expressions are specified using the `Expression` and `ExpressionBuilder` classes.

field

A piece of information in a database table record.

foreign key

A field or combination of fields that identify records in another table. Foreign keys are used to represent the relationships between records in different tables.

See also target foreign key.

home interface

A standard Java interface that extends `javax.ejb.EJBHome`. The home interface is accessed using JNDI and is used to create, find, and remove beans.

identity map

Used to cache objects for performance and to maintain object identity.

See also object identity.

independent relationship

A relationship in which the source and target are public objects that exist independently; the destruction of one object does not necessarily imply the destruction of the other.

Compare to private relationship.

indirection

An indirection object is one that acts as a stand-in for another object. In TopLink, indirection is implemented through Value Holders, which delay database access through acting as stand-in for any object relationships.

inheritance

Describes how a child class inherits the characteristics of its parent class. TopLink supports multiple approaches to database implementations that preserve the inheritance relationship.

instantiate

A Java term for creating an instance of a class.

JAR

Java archive, a file format used to bundle all components required by a Java application.

JDBC

Java Database Connectivity. Allows relational databases to be accessed from a common Java interface.

JMS

Java Messaging Service

JNDI

Java Naming and Directory Services

JTA

Java Transaction API

JTS

Java Transaction Services

leaf class

Has a persistent superclass in the hierarchy but does not have subclasses; queries performed on the leaf class can return only instances of the leaf class.

Compare to branch class and root class.

method

A component of a Java class that provides a path of execution

method access

The application registers accessor methods for the attribute.

Compare to direct access.

object identity

Ensures that each object is represented by one and only one instance in the application; that is, multiple retrievals of the same object return references to the same object instance, not multiple copies of the same object. Violating object identity can corrupt the object model.

See also identity map.

ODBC

Open Database Connectivity standard; designed by Microsoft to allow relational databases from multiple vendors to be accessed via a standard interface. Supported under most operating systems including Windows NT, Windows95, OS/2, and UNIX.

optimistic locking

Also known as write locking; allows unlimited read access to objects. A client can write an object to the database only if the object has not changed since it was last read.

Compare to pessimistic locking.

OTS

Object Transaction Service

pessimistic locking

Objects are locked before they are edited, which ensures that only one client is editing the object at any given time.

Compare to optimistic locking.

primary key

A field or combination of fields that uniquely identifies a record.

primary key object (EJB)

An object, which along with the home interface class, can uniquely identify an entity bean.

private relationship

A relationship in which the target object is considered to be a private component of the source object; the target object cannot exist without the source and is accessible only via the source object; furthermore, if the source object is destroyed, the target object is destroyed as well.

Compare to independent relationship.

Project Tree

The main interface of the TopLink Mapping Workbench. The Project Tree shows the high level information stored in a project.

query manager

An object, owned by a descriptor, that controls the way the descriptor accesses the database. The query manager generates its own default SQL to access the database in a transparent manner.

query optimization

TopLink supports two forms of query optimization: joining and batch reading. Their purpose is to optimize database access through reducing the number of database calls required to read a group of objects.

record

A group of related fields treated as a unit in a table.

reference

In Java, a pointer to another object.

relational database

A set of tables, the contents of which can be queried through SQL.

relationship

In TopLink, a reference between two TopLink-enabled objects.

relationship mapping

Persistent objects use relationship mappings to store references to instances of other persistent classes. The appropriate mapping class is chosen primarily by the cardinality of the relationship. TopLink provides five classes of relationship mappings.

Compare to direct mapping.

relation table

Contains columns for the primary keys of the source and target table.relational database

a database in the form of tables which have rows and columns to show the relationships between items, and in which information can be cross-referenced between two or more tables to generate a third table.

remote interface

A standard Java interface that extends `javax.ejb.EJBObject`. The client will make method calls on the remote interface to access the bean.

root class

Stores information about all of the instantiable classes in its subclass hierarchy. By default, queries performed on the root class return instances of the root class and its instantiable subclasses. However, the root class can be configured so that queries on it return only instances of itself without instances of its subclasses.

Compare to branch class and leaf class.

sequence numbers

A set of unique, sequential numbers assigned to the rows of a database to provide a unique way of identifying each record

session

The connection between an application and the relational database that stores its persistent objects.

session bean

Provide a particular service to the client and may be stateful or stateless.

SPI

Service Provider Interface

SQL

structured query language

target foreign key

A foreign key where the reference is from the target object's table back to the key from the mapping's descriptor's table.

Compare to foreign key.

Table Editor

A TopLink Mapping Workbench tool for defining a table.

three-tier application model

Consists of client applications, an application server that processes client requests, and a database that stores the data.

unit of work

A transactional TopLink session that allows for a transaction to occur at the object level not only the database level. Changes to objects are not visible globally until the unit of work is committed.

value holder

A wrapping object used by TopLink to delay database access.

verification

A check for the previous value of the target, accomplished through joining the source and target tables.

Index