Preface

This preface introduces you to the *Oracle Database JDBC Developer's Guide* discussing the intended audience, structure, and conventions of this document. A list of related Oracle documents is also provided.

Audience

The *Oracle Database JDBC Developer's Guide* is intended for developers of Java Database Connectivity (JDBC)-based applications. This book can be read by anyone with an interest in JDBC programming, but assumes at least some prior knowledge of the following:

- Java
- Oracle PL/SQL
- Oracle databases

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Related Documents

For more information, see the following documents in the Oracle Database documentation set:

- *Oracle Database Java Developer's Guide*
- *Oracle Database Development Guide*
- *Oracle Database PL/SQL Packages and Types Reference*
- *Oracle Database PL/SQL Language Reference*
- *Oracle Database SQL Language Reference*

You can also find more information on the following pages:

- http://www.oracle.com/technetwork/documentation/index.html
- http://www.oracle.com/technetwork/java/javase/jdbc/index.htm

Conventions

This section describes the conventions used in the text and code examples of this documentation set. It describes:

- Conventions in Text
- Conventions in Code Examples
- Conventions for Windows Operating Systems

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
Bold	Bold typeface indicates terms that are defined in the text or terms that appear in a glossary, or both.
Italics	Italic typeface indicates book titles or emphasis.
`UPPERCASE monospace (fixed-width) font`	Uppercase monospace typeface indicates elements supplied by the system. Such elements include parameters, privileges, data types, RMAN keywords, SQL keywords, SQL*Plus or utility commands, packages and methods, as well as system-supplied column names, database objects and structures, user names, and roles.
`lowercase monospace (fixed-width) font`	Lowercase monospace typeface indicates executables, file names, 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, user names and roles, program units, and parameter values. **Note:** Some programmatic elements use a mixture of UPPERCASE and lowercase. Enter these elements as shown.
`lowercase italic monospace (fixed-width) font`	Lowercase italic monospace font represents placeholders or variables.

Conventions in Code Examples

Code examples illustrate Java, SQL, and command-line statements. Examples 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
`[ ]`	Brackets enclose one or more optional items. Do not enter the brackets.
`{ }`	Braces enclose two or more items, one of which is required. Do not enter the braces.
`|`	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.

Convention	Meaning
. . .	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
. . .	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, braces, vertical bars, and ellipsis points as shown.
Italics	Italicized text indicates placeholders or variables for which you must supply particular values.
UPPERCASE	Uppercase typeface indicates elements supplied by the system. We show these terms in uppercase in order to distinguish them from terms you define. Unless terms appear in brackets, enter them in the order and with the spelling shown. However, because these terms are not case sensitive, you can enter them in lowercase.
lowercase	Lowercase typeface indicates programmatic elements that you supply. For example, lowercase indicates names of tables, columns, or files. **Note:** Some programmatic elements use a mixture of UPPERCASE and lowercase. Enter these elements as shown.

Conventions for Windows Operating Systems

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

Convention	Meaning
Choose Start >	How to start a program.
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.
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.
Special characters	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.
HOME_NAME	Represents the Oracle home name. The home name can be up to 16 alphanumeric characters. The only special character allowed in the home name is the underscore.

Convention	Meaning
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:\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 the latest Oracle release on a computer with no other Oracle software installed, then the default setting for the first Oracle home directory is `C:\oracle\ora`*nn*, where *nn* is the latest release number. The Oracle home directory is located directly under *ORACLE_BASE*. All directory path examples in this guide follow OFA conventions. Refer to *Oracle Database Platform Guide for Microsoft Windows* for additional information about OFA compliance and for information about installing Oracle products in non-OFA compliant directories.

Changes in This Release for Oracle Database JDBC Developer's Guide

This section contains the changes in this book for Oracle Database Release 23ai.

New Features

This section lists the new features for this release:

> **Note:**
>
> For a complete list of Oracle JDBC driver features, refer to the Feature List section.

- Support for Oracle AI Vector Search

  > **See Also:**
  >
  > Working with Vectors

- Support for Sessionless Transactions

  > **See Also:**
  >
  > Sessionless Transactions

- Support for Jakarta APIs

  > **See Also:**
  >
  > Supported JDK and JDBC Versions

- Support for SNI

  > **See Also:**
  >
  > SNI Support for TLS Connections

- Support for PEM

> **See Also:**
>
> Support for PEM in JDBC

- Support for Caching SSLContext Instance

> **See Also:**
>
> Support for Caching SSLContext Instance

- Support for Fixed Character Semantic

> **See Also:**
>
> Support for Fixed Character Semantic

- JDBC Service Provider Extensions

> **See Also:**
>
> JDBC Service Provider Extensions

- Enhanced Support for token-based authentication

> **See Also:**
>
> Support for Token-Based Authentication for IAM and Support for Token-Based Authentication for Azure AD

- Support for LDAP/LDAPS in the Easy Connect Plus URL

> **See Also:**
>
> Support for LDAP and LDAPS

- Support for RADIUS Challenge-Response Authentication

> **See Also:**
>
> Support for Challenge-Response Authentication

- Kerberos Authentication Enhancements

> **See Also:**
>
> Kerberos Authentication Enhancements

- Support for Kerberos Constrained Delegation

> **See Also:**
>
> Support for Kerberos Constrained Delegation

- Support for Using Alias and Thumbprint of Certificates

> **See Also:**
>
> About Managing Certificates and Wallets

- Performance Enhancement of Standard Update Batching

> **See Also:**
>
> About Processing the Batch

- Support for Pipelined Database Operations

> **See Also:**
>
> Support for Pipelined Database Operations

- Support for Data Load Mode in RSI

> **See Also:**
>
> About Reactive Streams Ingestion (RSI) Modes

- Support for Annotations

> **See Also:**
>
> Support for Annotations

- Support for Oracle True Cache

> **See Also:**
>
> Support for Oracle True Cache

- Support for the Bequeath Protocol

  > **See Also:**
  >
  > Support for the Bequeath Protocol

- Support for the SQL `BOOLEAN` Data Type

  > **See Also:**
  >
  > Class oracle.jdbc.OracleTypes and Table of Mappings

- Enhanced, Cloud-Ready Diagnosability Features

  > **See Also:**
  >
  > Diagnosability in JDBC

- Support for Multi-Pool DRCP

  > **See Also:**
  >
  > Multi-Pool Support in DRCP

- Support for Resumable Cursors with Transparent Application Continuity (TAC)

  > **See Also:**
  >
  > Support for Resumable Cursors

- Transaction Guard Support during `DBMS_ROLLING` Operations
  Transaction Guard support for rolling upgrades, using the `DBMS_ROLLING` package, ensures that commit outcomes are guaranteed across the entire upgrade process.

  > **See Also:**
  >
  > Database Development Guide

- Support for Enqueue and Dequeue of JSON Array Payload Type

  > **See Also:**
  >
  > Package oracle.sql.json

- Support for Passwords of Length 1024 Bytes

> **See Also:**
>
> Support for Longer Passwords

- Starting from Oracle Database Release 23ai, connection pooling support is implicitly provided to JDBC applications that do not use connection pools. So, even if your application does not close a connection explicitly, Database Resident Connection Pool (DRCP) servers are automatically assigned to and from an application connection at run time, when the application initiates and completes database operations.

> **See Also:**
>
> *Oracle Database Development Guide*

Deprecated Features

This section lists the deprecated features in this release.

Deprecation of the JDBC OCI Driver

The JDBC OCI Driver or Type 2 Client Driver, is deprecated in Oracle Database 23ai. Most Java applications that use Open Database Connectivity (ODBC) with Oracle JDBC drivers use the Thin driver. To enable Oracle to allocate resources to better address customer requirements, Oracle is deprecating the JDBC-OCI driver.

Deprecation of Blob, Clob, and BFile Methods

Oracle is deprecating the methods `open()`, `close()`, and `isClosed()` in the interfaces `oracle.jdbc.OracleBlob`, `oracle.jdbc.OracleClob`, and `oracle.jdbc.OracleBfile`.

These methods are replaced with the `openLob()`, `closeLob()` and `isClosedLob()` methods. The method `close()` conflicts with the type `java.lang.AutoCloseable`. Removing the proprietary method `close()` makes it possible for `OracleBlob`, `OracleClob`, and `OracleBfile` interfaces to extend the `AutoCloseable` interface at some future time. The `open()` and `isClosed()` methods will be removed and replaced to maintain rational names for these methods.