

# **Oracle® Providers for ASP.NET**

Developer's Guide

ODAC 12c Release 3 (12.1.0.2) for Microsoft Windows

**E59271-01**

December 2014

Oracle Providers for ASP.NET Developer's Guide, ODAC 12c Release 3 (12.1.0.2) for Microsoft Windows  
E59271-01

Copyright © 2007, 2014, Oracle and/or its affiliates. All rights reserved.

Primary Author: Maitreyee Chaliha

Contributing Authors: Janis Greenberg, Sumit Jeloka, Sheela Vasudevan, Kimnari Akiyama, Neeraj Gupta, Sinclair Hsu, Alex Keh, Ashish Shah, Arun Singh

Contributor: The Oracle Database 12c documentation is dedicated to Mark Townsend, who was an inspiration to all who worked on this release.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

---

---

# Contents

<b>Preface</b> .....	ix
Audience .....	ix
Documentation Accessibility .....	ix
Related Documents .....	ix
Passwords in Code Examples .....	x
Conventions .....	x
<b>Changes in This Release for Oracle Providers for ASP.NET Developer's Guide</b> ..	xi
Changes in Oracle Providers for ASP.NET Release 11.2.0.2 .....	xi
Changes in Oracle Providers for ASP.NET Release 11.2.0.1.2 .....	xi
<b>1 Introduction to Oracle Providers for ASP.NET</b>	
<b>Overview of Oracle Providers for ASP.NET</b> .....	1-1
<b>Oracle Providers for ASP.NET Assembly</b> .....	1-4
<b>System Requirements</b> .....	1-4
<b>Oracle Providers for ASP.NET Installation</b> .....	1-5
Database Server Setup .....	1-6
Database Privileges for Setup .....	1-6
Configuring All Oracle Providers for ASP.NET .....	1-6
Configuring Oracle Providers for ASP.NET Individually .....	1-7
General Setup Information .....	1-7
ASP.NET Client Setup .....	1-7
<b>Upgrading Oracle Providers for ASP.NET</b> .....	1-8
Coexistence of Multiple Oracle Providers for ASP.NET Versions .....	1-9
<b>File Locations After Installation</b> .....	1-9
<b>Oracle Providers for ASP.NET Object References</b> .....	1-10
Tables .....	1-10
Roles .....	1-11
Views .....	1-12
OracleMembershipProvider Views .....	1-12
OracleRoleProvider Views .....	1-12
OracleProfileProvider Views .....	1-12
OraclePersonalizationProvider Views .....	1-13
OracleSessionStateStore Views .....	1-13
Stored Procedures .....	1-13

OracleMembershipProvider Stored Procedures .....	1-13
OracleRoleProvider Stored Procedures .....	1-14
OracleProfileProvider Stored Procedures .....	1-15
OraclePersonalizationProvider Stored Procedures .....	1-16
OracleWebEventProvider Stored Procedures .....	1-17
OracleSiteMapProvider Stored Procedures .....	1-17
OracleSessionStateStore Provider Stored Procedures .....	1-17
Synonyms .....	1-18

## 2 OracleMembershipProvider

<b>OracleMembershipProvider Class</b> .....	2-2
OracleMembershipProvider Members .....	2-4
OracleMembershipProvider Constructors .....	2-7
OracleMembershipProvider() .....	2-7
OracleMembershipProvider Static Methods .....	2-8
OracleMembershipProvider Public Properties .....	2-9
ApplicationName .....	2-10
CommandTimeout .....	2-10
EnablePasswordReset .....	2-11
EnablePasswordRetrieval .....	2-11
MaxInvalidPasswordAttempts .....	2-12
MinRequiredNonAlphanumericCharacters .....	2-13
MinRequiredPasswordLength .....	2-13
PasswordAttemptWindow .....	2-14
PasswordCompatMode .....	2-15
PasswordFormat .....	2-16
PasswordStrengthRegularExpression .....	2-16
RequiresQuestionAndAnswer .....	2-17
RequiresUniqueEmail .....	2-18
OracleMembershipProvider Public Methods .....	2-19
ChangePassword .....	2-20
ChangePasswordQuestionAndAnswer .....	2-21
CreateUser .....	2-22
DeleteUser .....	2-23
FindUsersByEmail .....	2-24
FindUsersByName .....	2-25
GeneratePassword .....	2-26
GetAllUsers .....	2-27
GetNumberOfUsersOnline .....	2-28
GetPassword .....	2-28
GetUser .....	2-29
GetUser(Object, bool) .....	2-30
GetUser(string, bool) .....	2-31
GetUserNameByEmail .....	2-31
Initialize .....	2-32
ResetPassword .....	2-33
UnlockUser .....	2-35

UpdateUser .....	2-35
ValidateUser .....	2-36
OracleMembershipProvider Public Events .....	2-38

### 3 OracleRoleProvider

<b>OracleRoleProvider Class</b> .....	3-2
OracleRoleProvider Members .....	3-4
OracleRoleProvider Constructors .....	3-6
OracleRoleProvider() .....	3-6
OracleRoleProvider Static Methods .....	3-7
OracleRoleProvider Public Properties .....	3-8
ApplicationName .....	3-8
CommandTimeout .....	3-9
OracleRoleProvider Public Methods .....	3-10
AddUsersToRoles .....	3-10
CreateRole .....	3-11
DeleteRole .....	3-12
FindUsersInRole .....	3-12
GetAllRoles .....	3-13
GetRolesForUser .....	3-14
GetUsersInRole .....	3-14
Initialize .....	3-15
IsUserInRole .....	3-16
RemoveUsersFromRoles .....	3-16
RoleExists .....	3-17

### 4 OracleSiteMapProvider

<b>OracleSiteMapProvider Class</b> .....	4-2
OracleSiteMapProvider Members .....	4-4
OracleSiteMapProvider Constructors .....	4-6
OracleSiteMapProvider() .....	4-6
OracleSiteMapProvider Static Methods .....	4-7
OracleSiteMapProvider Public Properties .....	4-8
ApplicationName .....	4-8
CommandTimeout .....	4-9
OracleSiteMapProvider Public Methods .....	4-10
BuildSiteMap .....	4-10
Dispose .....	4-11
Initialize .....	4-11

### 5 OracleSessionStateStore

<b>OracleSessionStateStore Class</b> .....	5-2
OracleSessionStateStore Members .....	5-4
OracleSessionStateStore Constructors .....	5-6
OracleSessionStateStore() .....	5-6
OracleSessionStateStore Public Properties .....	5-7

CommandTimeout.....	5-7
OracleSessionStateStore Public Methods.....	5-8
CreateNewStoreData.....	5-8
CreateUninitializedItem.....	5-9
Dispose.....	5-10
EndRequest.....	5-10
GetItem.....	5-10
GetItemExclusive.....	5-12
Initialize.....	5-13
InitializeRequest.....	5-14
ReleaseItemExclusive.....	5-14
RemoveItem.....	5-15
ResetItemTimeout.....	5-16
SetAndReleaseItemExclusive.....	5-16
SetItemExpireCallback.....	5-17

## 6 OracleProfileProvider

<b>OracleProfileProvider Class</b> .....	6-2
OracleProfileProvider Members.....	6-4
OracleProfileProvider Constructors.....	6-6
OracleProfileProvider().....	6-6
OracleProfileProvider Static Methods.....	6-7
OracleProfileProvider Public Properties.....	6-8
ApplicationName.....	6-8
CommandTimeout.....	6-9
OracleProfileProvider Public Methods.....	6-10
DeleteInactiveProfiles.....	6-10
DeleteProfiles.....	6-11
DeleteProfiles(ProfileInfoCollection).....	6-11
DeleteProfiles(string[]).....	6-12
FindInactiveProfilesByUserName.....	6-13
FindProfilesByUserName.....	6-15
GetAllInactiveProfiles.....	6-16
GetAllProfiles.....	6-17
GetNumberOfInactiveProfiles.....	6-18
GetPropertyValues.....	6-18
Initialize.....	6-19
SetPropertyValues.....	6-20

## 7 OracleWebEventProvider

<b>OracleWebEventProvider Class</b> .....	7-2
OracleWebEventProvider Members.....	7-5
OracleWebEventProvider Constructors.....	7-7
OracleWebEventProvider().....	7-7
OracleWebEventProvider Static Methods.....	7-8
OracleWebEventProvider Public Properties.....	7-9
CommandTimeout.....	7-9

OracleWebEventProvider Public Methods .....	7-10
Initialize .....	7-10
ProcessEvent .....	7-11
ProcessEventFlush .....	7-11
Shutdown .....	7-12

## 8 OraclePersonalizationProvider

<b>OraclePersonalizationProvider Class</b> .....	8-2
OraclePersonalizationProvider Members .....	8-4
OraclePersonalizationProvider Constructors .....	8-6
OraclePersonalizationProvider() .....	8-6
OraclePersonalizationProvider Static Methods .....	8-7
OraclePersonalizationProvider Public Properties .....	8-8
ApplicationName .....	8-8
CommandTimeout .....	8-9
OraclePersonalizationProvider Public Methods .....	8-10
FindState .....	8-10
GetCountOfState .....	8-12
Initialize .....	8-13
ResetState .....	8-14
ResetUserState .....	8-15

## 9 OracleCacheDependency Provider

<b>OracleCacheDependency Class</b> .....	9-2
OracleCacheDependency Members .....	9-3
OracleCacheDependency Constructors .....	9-4
OracleCacheDependency(OracleCommand) .....	9-4
OracleCacheDependency Properties .....	9-5
OracleCacheDependency Methods .....	9-6
GetUniqueID .....	9-6

## Index





---

---

# Preface

This document is your primary source of introductory, installation, postinstallation configuration, and usage information for Oracle Providers for ASP.NET.

This Preface contains these sections:

- [Audience](#)
- [Documentation Accessibility](#)
- [Related Documents](#)
- [Passwords in Code Examples](#)
- [Conventions](#)

## Audience

*Oracle Providers for ASP.NET Developer's Guide* is intended for programmers who are developing applications using ASP.NET providers to store application state in Oracle databases.

To use this guide, you must be familiar with Microsoft .NET Framework classes, ASP.NET, and ADO.NET, and have a working knowledge of application programming using Microsoft C#, Visual Basic .NET, or another .NET language.

Although the examples in the documentation and the samples in the sample directory are written in C#, developers can use these examples as models for writing code in other .NET languages.

## 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 these Oracle resources:

- *Oracle Database Installation Guide for Windows*
- *Oracle Database Release Notes for Windows*
- *Oracle Database Platform Guide for Windows*
- *Oracle Database New Features*
- *Oracle Database Concepts*
- *Oracle Database Reference*
- *Oracle Data Provider for .NET Developer's Guide*
- *Oracle Developer Tools for Visual Studio .NET Help*

## Passwords in Code Examples

For simplicity in demonstrating this product, code examples do not perform the password management techniques that a deployed system normally uses. In a production environment, follow the Oracle Database password management guidelines, and disable any sample accounts. See *Oracle Database Security Guide* for password management guidelines and other security recommendations.

## Conventions

The following text conventions are used in this guide:

<b>Convention</b>	<b>Meaning</b>
<b>boldface</b>	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
<i>italic</i>	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
monospace	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

---

---

# Changes in This Release for Oracle Providers for ASP.NET Developer's Guide

This preface contains:

- [Changes in Oracle Providers for ASP.NET Release 11.2.0.2](#)
- [Changes in Oracle Providers for ASP.NET Release 11.2.0.1.2](#)

## Changes in Oracle Providers for ASP.NET Release 11.2.0.2

The following are changes in Oracle Providers for ASP.NET for Release 11.2.0.2.

### New Features

The following feature is new in this release:

- 64-bit Oracle Providers for ASP.NET XCopy for Windows x64

Now available for Windows x64 systems, Oracle Providers for ASP.NET XCopy provides system administrators with a smaller client install size than the standard client, and is easier to configure. XCopy simplifies embedding the ASP.NET providers in customized deployment packages.

**See Also:** XCopy under "[Oracle Providers for ASP.NET Installation](#)" on page 1-5.

## Changes in Oracle Providers for ASP.NET Release 11.2.0.1.2

The following are changes in Oracle Providers for ASP.NET for Release 11.2.0.1.2.

### New Features

The following feature is new in this release:

- ASP.NET 4 Support

Oracle Providers for ASP.NET 4 supports ASP.NET 4.



---

# Introduction to Oracle Providers for ASP.NET

This chapter introduces Oracle Providers for ASP.NET.

This chapter contains these topics:

- [Overview of Oracle Providers for ASP.NET](#)
- [Oracle Providers for ASP.NET Assembly](#)
- [System Requirements](#)
- [Oracle Providers for ASP.NET Installation](#)
- [Upgrading Oracle Providers for ASP.NET](#)
- [File Locations After Installation](#)
- [Oracle Providers for ASP.NET Object References](#)

## Overview of Oracle Providers for ASP.NET

Oracle Providers for ASP.NET is a collection of ASP.NET providers that follow the ASP.NET provider model and uses Oracle Database as the data source.

Microsoft ASP.NET includes a number of services and providers that store application state in databases and other storage media. Developers can store application state, such as shopping cart or user information, in a persistent data source. By storing the application state in a database, applications ensure high availability and reliable access to the data from any Web server in the server farm. Users can retrieve their ASP.NET state data no matter which Web farm computer they access because it is located centrally on the database. If the particular Web farm computer a user is accessing fails, the information is not lost because the ASP.NET data is persisted in the database. ASP.NET developers can now be more productive as they build their Web applications through these ASP.NET services, which are classes that are part of the .NET Framework.

These ASP.NET services are data-source independent, but they can be configured to use a particular ASP.NET provider, which is implemented specifically to store and retrieve data from a specific data source.

Oracle Providers for ASP.NET, like all ASP.NET providers, follow the ASP.NET provider model to provide all the functionality that the ASP.NET services need. By simply configuring the Oracle Providers for ASP.NET as default providers in a configuration file, ASP.NET applications can store various types of application states in an Oracle database. The application states that can be stored are commonly used among Web applications. ASP.NET developers can use these providers, as opposed to creating their own from scratch.

Oracle Providers for ASP.NET are available for both 32-bit and 64-bit (x64) platforms. See "[System Requirements](#)" for more details.

Oracle offers the following providers:

- Membership Provider
- Role Provider
- Site Map Provider
- Session State Provider
- Profile Provider
- Web Event Provider
- Web Parts Personalization Provider
- Cache Dependency Provider

Descriptions of each provider that Oracle offers are as follows:

- Oracle Membership Provider for ASP.NET

The Oracle membership provider enables ASP.NET applications to store the registered user information of a Web site in an Oracle database through the ASP.NET membership service. It provides methods for creating users, deleting users, verifying login credentials, changing passwords, and other tasks associated with managing application users.

**See Also:** ASP.NET membership and membership providers  
<http://msdn.microsoft.com/en-us/library/tw292whz.aspx>

- Oracle Role Provider for ASP.NET

The Oracle role provider enables ASP.NET applications to store and manage Web site-specific role information in an Oracle database through the ASP.NET role service. The Oracle role provider exposes methods for creating roles, deleting roles, adding users to roles, and other tasks associated with managing roles defined in an ASP.NET application.

**See Also:** ASP.NET role management and role providers  
<http://msdn.microsoft.com/en-us/library/9ab2fxh0.aspx>

- Oracle Site Map Provider for ASP.NET

The Oracle site map provider enables ASP.NET applications to store site map information in an Oracle database. The Oracle site map provider reads site map data from the Oracle database to build an upside-down tree of `SiteMapNode` objects, as well as to supply methods for retrieving nodes from the site map.

**See Also:** ASP.NET site navigation and site map provider  
<http://msdn.microsoft.com/en-us/library/ms227558.aspx>

- Oracle Session State Provider for ASP.NET

The Oracle session state provider enables ASP.NET applications to store ASP.NET session information in an Oracle database through the ASP.NET session state service. This provider manages per-user session state, such as a shopping cart for an e-commerce application.

**See Also:** ASP.NET session state and session state providers  
<http://msdn.microsoft.com/en-us/library/ms178581.aspx>

- Oracle Profile Provider for ASP.NET

The Oracle profile provider enables ASP.NET applications to store an individual Web site user's profile information in the Oracle database. The profile provider can write and read Web site user profile properties that are persisted in the database.

**See Also:** ASP.NET profile properties and profile providers  
<http://msdn.microsoft.com/en-us/library/2y3fs9xs.aspx>

- Oracle Web Event Provider for ASP.NET

The Oracle Web event provider enables ASP.NET applications to store events raised by the ASP.NET health monitoring system in the Oracle database. The provider provides buffering and flushing capabilities to minimize database round-trips.

**See Also:** ASP.NET health monitoring and web event provider  
<http://msdn.microsoft.com/en-us/library/ms178701%28VS.80%29.aspx>

- Oracle Web Parts Personalization Provider for ASP.NET

The Oracle Web parts personalization provider enables ASP.NET applications to store personalization data in an Oracle database through the ASP.NET Web parts personalization service.

It connects to an integrated set of controls for creating Web sites that enable end users to modify the content, appearance, and behavior of Web pages directly from a browser.

**See Also:**

- ASP.NET Web Parts  
<http://msdn.microsoft.com/en-us/library/e0s9t4ck.aspx>
- ASP.NET Web Parts Personalization  
<http://msdn.microsoft.com/en-us/library/ms178182.aspx>

- Oracle Cache Dependency Provider for ASP.NET

Oracle cache dependency provider provides automatic invalidation of data that is cached by ASP.NET applications in the `System.Web.Caching.Cache` object, based on changes made in the Oracle database. This provider can provide performance improvements to ASP.NET applications because ASP.NET applications can use the cached database data and fetch data from the database only when it is needed.

**See Also:** *Oracle Data Provider for .NET Developer's Guide* for more information on database change notification

**See Also:** ASP.NET CacheDependency class  
<http://msdn.microsoft.com/en-us/library/system.web.caching.cachedependency.aspx>

## Oracle Providers for ASP.NET Assembly

The Oracle providers for ASP.NET reside in namespaces contained in one assembly: `Oracle.Web.dll`.

[Table 1–1](#) lists the provider types, class names, and namespaces that are part of `Oracle.Web.dll`.

**Table 1–1 Oracle Providers for ASP.NET Namespaces and Providers**

Provider Type	Class Name	Namespace
Membership	<a href="#">OracleMembershipProvider Class</a>	<code>Oracle.Web.Security</code>
Role	<a href="#">OracleRoleProvider Class</a>	<code>Oracle.Web.Security</code>
Site Map	<a href="#">OracleSiteMapProvider Class</a>	<code>Oracle.Web.SiteMap</code>
Session State	<a href="#">OracleSessionStateStore Class</a>	<code>Oracle.Web.SessionState</code>
Profile	<a href="#">OracleProfileProvider Class</a>	<code>Oracle.Web.Profile</code>
Web Event	<a href="#">OracleWebEventProvider Class</a>	<code>Oracle.Web.Management</code>
Personalization	<a href="#">OraclePersonalizationProvider Class</a>	<code>Oracle.Web.Personalization</code>
Cache Dependency	<a href="#">OracleCacheDependency Class</a>	<code>Oracle.Web.Caching</code>

## System Requirements

Oracle Providers for ASP.NET requires the following:

- Microsoft ASP.NET
  - Oracle Providers for ASP.NET 2.0 is supported with ASP.NET for .NET Framework 3.5 Service Pack 1 and higher
  - Oracle Providers for ASP.NET 4 is supported with ASP.NET for .NET Framework 4 and higher
- Windows operating system
  - 32-bit: Windows 7 (Professional, Enterprise, and Ultimate Editions), Windows Server 2008 (Standard, Enterprise, Datacenter, Web, and Foundation Editions), and Windows 8 (Pro and Enterprise Editions).

Oracle supports 32-bit Oracle Providers for ASP.NET on x86, AMD64, and Intel EM64T processors on these operating systems.

- x64: Windows 7 x64 (Professional, Enterprise, and Ultimate Editions), Windows Server 2008 R2 x64 (Standard, Enterprise, Datacenter, Web, and Foundation Editions), Windows Server 2008 x64 (Standard, Enterprise, Datacenter, Web, and Foundation Editions), Windows 8 x64 (Pro and Enterprise Editions), and Windows Server 2012 x64 (Standard, Datacenter, Essentials, and Foundation Editions).
- Oracle supports 32-bit Oracle Providers for ASP.NET and 64-bit Oracle Providers for ASP.NET for Windows x64 on these operating systems.
- Access to Oracle Database 10g Release 2 or later.
  - Oracle Data Provider for .NET, Unmanaged Driver and Oracle Client (installed with Oracle Providers for ASP.NET software). You must use the same version of the ODP.NET and Oracle Client with Oracle Providers for ASP.NET. For example, if you use Oracle Providers for ASP.NET version 11.2, you should use ODP.NET and Oracle Client versions 11.2 as well.



- Oracle Providers for ASP.NET 2.0 depends on Oracle Data Provider for .NET 2.0
- Oracle Providers for ASP.NET 4 depends on Oracle Data Provider for .NET 4

---

---

**Note:** Oracle Data Provider for .NET, Managed Driver does not support Oracle Providers for ASP.NET

---

---

## Oracle Providers for ASP.NET Installation

Oracle Providers for ASP.NET is part of Oracle Data Access Components (ODAC), which can be downloaded from OTN. Beginning with ODAC 11.1.0.6.20, Oracle Providers for ASP.NET can be installed through XCopy or Oracle Universal Installer.

- XCopy

Administrators use XCopy to deploy Oracle Providers for ASP.NET to a large number of computers for production deployments. It has a small footprint and fine grain control during installation and setup.

- Oracle Universal Installer

Developers or users use Oracle Universal Installer for automatic GUI installation. It includes documentation and code samples that are not part of XCopy deployment.

---

---

**Note:** This section describes installation using the Oracle Universal Installer. For the XCopy installation instructions and configuration, refer to the README file that is part of that installation.

---

---

When Oracle Providers for ASP.NET are installed, Oracle Universal Installer automatically registers `Oracle.Web.dll` with the Global Assembly Cache (GAC).

The Oracle Providers for ASP.NET installation creates entries in the `machine.config` file of the computer on which it is installed. These entries provide basic configuration information for the Oracle Membership, Profile, Role, Site Map, Web Parts Personalization, and Web Event Providers. The `machine.config` includes a blank connection string for `OraAspConString`, which a developer can configure to connect to an Oracle database with the Oracle Providers for ASP.NET schema.

**See Also:** "[ASP.NET Client Setup](#)" on page 1-7 for more details

Additionally, Oracle Providers for ASP.NET Dynamic Help is registered with Visual Studio, providing context-sensitive online help that is seamlessly integrated with the Visual Studio Dynamic Help. With Dynamic Help, the user can access Oracle Providers for ASP.NET documentation within Visual Studio by placing the cursor on an Oracle Providers for ASP.NET keyword and pressing the F1 function key.

Once you have installed Oracle Providers for ASP.NET, two additional setup tasks are required, as follows:

- [Database Server Setup](#)
- [ASP.NET Client Setup](#)

**See Also:** *Oracle Database Installation Guide for Windows* for installation instructions

## Database Server Setup

The following sections explain how to configure the providers:

- [Database Privileges for Setup](#)
- [Configuring All Oracle Providers for ASP.NET](#)
- [Configuring Oracle Providers for ASP.NET Individually](#)
- [General Setup Information](#)

### Database Privileges for Setup

To set up the Oracle database, database administrators must grant the following database privileges to the Oracle Providers for ASP.NET schema. These privileges grant the schema privileges to create the tables, views, stored procedures, and other database objects the Oracle Providers for ASP.NET require. These scripts must be run against the database from which the ASP.NET providers will retrieve their stored state information. These SQL scripts can be run using SQL\*Plus or within Oracle Developer Tools for Visual Studio.

Oracle Providers for ASP.NET requires the following privileges during setup:

- Change notification
- Create job
- Create procedure
- Create public synonym
- Create role
- Create session
- Create table
- Create view
- Drop public synonym
- Grant access to and allocate space in an Oracle tablespace

Not all database privileges are required for Oracle Providers for ASP.NET runtime operations. Database administrators may selectively grant and revoke privileges as required. For runtime operations, all providers require that the `CREATE SESSION` privilege be granted to the schema user. In addition, the Site Map and Cache Dependency providers require the `CHANGE NOTIFICATION` privilege during runtime. The remaining privileges can be granted to the schema user just for installation and deinstallation, then revoked for runtime operations.

Errors that occur during the setup script execution may indicate that the user needs to be granted the above privileges. If this is the case, the database administrator must grant these privileges. The Oracle Session State Provider for ASP.NET requires the `CREATE JOB` privilege when Oracle Database 10g or later is the database.

### Configuring All Oracle Providers for ASP.NET

To configure all providers in the database at once, run `InstallAllOracleASPNETProviders.sql`. This script is found in the `ORACLE_BASE\ORACLE_HOME\ASP.NET\sql` directory.

## Configuring Oracle Providers for ASP.NET Individually

Applications may not require all Oracle Providers for ASP.NET. Providers can be set up individually. Except for the Oracle Session State Provider and Oracle Cache Dependency Provider, the following install script must be executed before any other install scripts: `InstallOracleASPNETCommon.sql`. Then, for each Oracle Provider for ASP.NET, a SQL script specific for that provider must be executed (in any order).

These install scripts are found in the `ORACLE_BASE\ORACLE_HOME\ASP.NET\sql` directory.

**Table 1–2 Install Scripts**

Provider	Required Installation Script
Oracle Membership Provider	<code>InstallOracleMembership.sql</code>
Oracle Personalization Provider	<code>InstallOraclePersonalization.sql</code>
Oracle Profile Provider	<code>InstallOracleProfile.sql</code>
Oracle Role Provider	<code>InstallOracleRoles.sql</code>
Oracle Session State Provider	<code>InstallOracleSessionState.sql</code> Note: This provider requires the execution of only the <code>InstallOracleSessionState.sql</code> script. It does not require the execution of <code>InstallOracleASPNETCommon.sql</code> .
Oracle Site Map Provider	<code>InstallOracleSiteMap.sql</code>
Oracle Web Events Provider	<code>InstallOracleWebEvents.sql</code>
Oracle Cache Dependency Provider	No script execution needed

## General Setup Information

When Oracle Providers for ASP.NET installation scripts run, they execute, in turn, corresponding `.plb` scripts that are located in the same directory. The `.plb` scripts create the stored procedures and functions that the providers use. The installation `.sql` scripts must execute where the `.plb` file can be accessed.

Each provider also provides corresponding uninstall scripts to remove database objects that were created from the install scripts. These scripts are prefixed with the word `Uninstall`.

The install and uninstall scripts are the same for Oracle Providers for ASP.NET 2.0 and Oracle Providers for ASP.NET 4.

## ASP.NET Client Setup

Installation configures the `machine.config` file to enable Oracle Providers for ASP.NET system-wide.

If both .NET Framework 2.0 and .NET Framework 4 are present, then the installer configures the `machine.config` files for both in order to enable Oracle Providers for ASP.NET 2.0 and Oracle Providers for ASP.NET 4 system-wide.

After installation, developers must provide the connection information to the database schema that stores and retrieves the ASP.NET state information. This step requires developers to supply the `User Id`, `Password`, `Data Source`, and other connection string information if necessary. In the `machine.config` file, developers can provide an entry similar to the example below.

```
<connectionStrings>
<add name="OraAspNetConString" connectionString="User
Id=aspnet;Password=aspnet;Data Source=oracle; " />
</connectionStrings>
```

Optionally, developers can customize the properties of each ASP.NET provider from within the `<system.web>` section of the `machine.config`.

While Oracle Universal Installer automatically configures the `machine.config`, developers can apply more fine grained application-level control over the Oracle Providers for ASP.NET by using the `web.config` file. This file overrides entries from the `machine.config` file, but only for the specific web application it is a part of. Developers can set up their `web.config` file with the same XML syntax as the `machine.config` file.

Developers can use standard ASP.NET management tools to configure the Oracle Providers for ASP.NET. Specifically, developers can use the Internet Information Services management console. In the ASP.NET Configuration Settings of the console, developers can modify the Oracle provider settings. Alternatively, in a Web Site project in Visual Studio, the ASP.NET Configuration choice under the Website menu item allows developers to set these settings for their specific Web Site projects.

A standard Oracle Providers for ASP.NET installation automatically sets up `machine.config` with default values, but administrators can modify the setup. Users can use the `OraProvCfg` utility to configure the provider-specific entry in the `machine.config` file as follows:

---

---

**Note:** To configure Oracle Providers for ASP.NET 2.0, use the `OraProvCfg` utility under `ORACLE_BASE\ORACLE_HOME\ASP.NET\Bin\2.x`.

To configure Oracle Providers for ASP.NET 4, use the `OraProvCfg` utility under `ORACLE_BASE\ORACLE_HOME\ASP.NET\Bin\4`.

---

---

- To display the `OraProvCfg` utility help:

```
OraProvCfg -help
```

- To add Oracle Providers for ASP.NET-specific entries to the `machine.config` file:

```
OraProvCfg /action:config /product:aspnet /component:all
           /frameworkversion:v2.0.50727
           /providerpath:c:\Oracle\odp.net\bin\2.x\Oracle.Web.dll
```

Where Framework version and Provider path (especially) may need to change accordingly.

- To remove the Oracle Providers for ASP.NET-specific entries from the `machine.config` file:

```
OraProvCfg /action:unconfig /product:aspnet /component:all
           /frameworkversion:v2.0.50727
```

Where Framework version may need to change accordingly.

## Upgrading Oracle Providers for ASP.NET

Administrators who wish to upgrade an older instance of Oracle Providers for ASP.NET to a newer version must upgrade both the Oracle Client and database

schema. Oracle does not support using one Oracle Providers for ASP.NET client version, say 11.2, with another Oracle Providers for ASP.NET database schema version, say 12.1. Both the schema and client providers must be the same version.

The following list discusses upgrading the client and database schema:

- Oracle Client Upgrade:

When installing the latest Oracle Providers for ASP.NET version, the `machine.config` file is automatically updated so that Web applications are directed to use the latest version. If there are applications that use the `web.config` file, then administrators must make sure to update this file.

- Database Schema Upgrade:

Administrators need to execute the new version's Oracle Providers for ASP.NET SQL scripts on the same schema where the older Oracle Providers for ASP.NET schema exists. The scripts are designed to upgrade older schemas or create the schema if none exists. The scripts preserve existing Oracle Providers for ASP.NET data such that no data is lost.

If the database server itself is being upgraded, then administrators can use standard Oracle upgrade procedures with Oracle Providers for ASP.NET data. Data is preserved when performing the upgrade. Encrypted data remains encrypted and usable after the upgrade.

## Coexistence of Multiple Oracle Providers for ASP.NET Versions

If there are multiple ASP.NET applications using a single Web server or a single Oracle Database, then it is not necessary for all of them to use the same Oracle Providers for ASP.NET version. For example, some of the applications may use Oracle Providers for ASP.NET 12.1, and other applications may use the 11.2 version. Individual `web.config` files are used to determine as to which Oracle Providers for ASP.NET version to use for each application.

Each Oracle Providers for ASP.NET version must have a database schema specific to its version. For example, all Oracle Providers for ASP.NET 11.2 applications must be able to access at least one schema built using the 11.2 SQL scripts. These 11.2 provider applications can all share one schema, but they cannot use an 12.1 schema. When using multiple Oracle Providers for ASP.NET versions to access the same database, administrators can create separate schemas for each Oracle Providers for ASP.NET version.

## File Locations After Installation

Oracle Providers for ASP.NET files are installed as follows:

- Oracle.Web.dll (For .NET Framework 2.0)  
`ORACLE_BASE\ORACLE_HOME\ASP.NET\Bin\2.x` directory
- Oracle.Web.dll (For .NET Framework 4)  
`ORACLE_BASE\ORACLE_HOME\ASP.NET\Bin\4` directory
- Configuration utility `OraProvCfg.exe` (For .NET Framework 2.0)  
`ORACLE_BASE\ORACLE_HOME\ASP.NET\Bin\2.x` directory. See "[ASP.NET Client Setup](#)" on page 1-7.
- Configuration utility `OraProvCfg.exe` (For .NET Framework 4)

*ORACLE\_BASE\ORACLE\_HOME\ASP.NET\Bin\4* directory. See "[ASP.NET Client Setup](#)" on page 1-7.

- Configuration (SQL) scripts

*ORACLE\_BASE\ORACLE\_HOME\ASP.NET\SQL* directory. See "[Database Server Setup](#)" on page 1-6.

- Dynamic Help file

*ORACLE\_BASE\ORACLE\_HOME\ASP.NET\Help* directory

- Documentation (html and pdf) and readme file

*ORACLE\_BASE\ORACLE\_HOME\ASP.NET\Doc* directory

## Oracle Providers for ASP.NET Object References

The schema in which the user runs the SQL installation script owns the tables, views, roles, stored procedures, and synonyms that the SQL script creates.

The following schema objects and their tabled information provide descriptions of what privileges each role provides, as well as the relationship between the ASP.NET service methods and the Oracle stored procedure or function.

This section lists the following objects:

- [Tables](#)
- [Roles](#)
- [Views](#)
- [Stored Procedures](#)
- [Synonyms](#)

### Tables

[Table 1–3](#) lists the tables that are used by each provider.

**Table 1–3 Provider Tables**

Oracle Provider	Table
Membership	ora_aspnet_Membership
	ora_aspnet_Applications
	ora_aspnet_Users
Role	ora_aspnet_Roles
	ora_aspnet_UsersInRoles
	ora_aspnet_Applications
	ora_aspnet_Users
Profile	ora_aspnet_Profile
	ora_aspnet_Applications
	ora_aspnet_Users

**Table 1–3 (Cont.) Provider Tables**

Oracle Provider	Table
Personalization	ora_aspnet_Paths
	ora_aspnet_PersonaliznAllUsers
	ora_aspnet_PersonaliznPerUser
	ora_aspnet_Applications
	ora_aspnet_Users
Web Events	ora_aspnet_WebEvents
Site Map	ora_aspnet_SiteMap
	ora_aspnet_Applications
Session State	ora_aspnet_SessionApplications
	ora_aspnet_Sessions

## Roles

There are, at most, three types of database roles created for each provider:

- **BasicAccess** - Provides a database user with access to the provider's basic functionality.
- **ReportAccess** - Provides a database user with report-oriented data gathering capabilities for a provider.
- **FullAccess** - Provides a database user with access to all the database objects associated with a provider.

[Table 1–4](#) lists the roles created for each provider.

**Table 1–4 Roles and Privileges**

Oracle Provider	Oracle Database Role
Membership	ora_aspnet_Mem_BasicAccess
	ora_aspnet_Mem_ReportAccess
	ora_aspnet_Mem_FullAccess
Role	ora_aspnet_Roles_BasicAccess
	ora_aspnet_Roles_ReportAccess
	ora_aspnet_Roles_FullAccess
Profile	ora_aspnet_Prof_BasicAccess
	ora_aspnet_Prof_ReportAccess
	ora_aspnet_Prof_FullAccess
Personalization	ora_aspnet_Pers_BasicAccess
	ora_aspnet_Pers_ReportAccess
	ora_aspnet_Pers_FullAccess
Web Events	ora_aspnet_Wevnt_FullAccess
Site Map	ora_aspnet_Smap_FullAccess
Session	ora_aspnet_Sessn_FullAccess

## Views

The following tables show the views that are created for each provider. The tables also list the provider-specific database roles that provide access to these views.

### OracleMembershipProvider Views

[Table 1-5](#) lists the roles and the view access that the roles provide.

**Table 1-5 OracleMembershipProvider**

Role	View
ora_aspnet_Mem_BasicAccess	(none)
ora_aspnet_Mem_ReportAccess	ora_vw_aspnet_Applications ora_vw_aspnet_Users ora_vw_aspnet_MemUsers
ora_aspnet_Mem_FullAccess	ora_vw_aspnet_Applications ora_vw_aspnet_Users ora_vw_aspnet_MemUsers

### OracleRoleProvider Views

[Table 1-6](#) lists the roles and the view access that the roles provide.

**Table 1-6 OracleRoleProvider**

Role	View
ora_aspnet_Roles_BasicAccess	(none)
ora_aspnet_Roles_ReportAccess	ora_vw_aspnet_Applications ora_vw_aspnet_Users ora_vw_aspnet_Roles ora_vw_aspnet_UIR
ora_aspnet_Roles_FullAccess	ora_vw_aspnet_Applications ora_vw_aspnet_Users ora_vw_aspnet_Roles ora_vw_aspnet_UIR

### OracleProfileProvider Views

[Table 1-7](#) lists the roles and the view access that the roles provide.

**Table 1-7 OracleProfileProvider**

Role	View
ora_aspnet_Prof_BasicAccess	(none)
ora_aspnet_Prof_ReportAccess	ora_vw_aspnet_Applications ora_vw_aspnet_Users ora_vw_aspnet_Profiles
ora_aspnet_Prof_FullAccess	ora_vw_aspnet_Applications ora_vw_aspnet_Users



**Table 1–7 (Cont.) OracleProfileProvider**

Role	View
	ora_vw_aspnet_Profiles

### OraclePersonalizationProvider Views

Table 1–8 lists the roles and the view access that the roles provide.

**Table 1–8 OraclePersonalizationProvider**

Role	View
ora_aspnet_Pers_BasicAccess	(none)
ora_aspnet_Pers_ReportAccess	ora_vw_aspnet_Applications ora_vw_aspnet_Users
ora_aspnet_Pers_FullAccess	ora_vw_aspnet_Applications ora_vw_aspnet_Users

### OracleSessionStateStore Views

Table 1–9 lists the roles and the view access that the roles provide.

**Table 1–9 OracleSessionStateStore**

Role	View
ora_aspnet_Sessn_FullAccess	ora_vew_aspnet_sessions

## Stored Procedures

The following tables list provider-specific database roles and the stored procedures for which the roles provide execution privilege. The tables also list the corresponding ASP.NET service methods that invoke the stored procedures.

### OracleMembershipProvider Stored Procedures

Table 1–10 lists the service methods and stored procedures that a user with the ora\_aspnet\_Mem\_BasicAccess role can execute.

**Table 1–10 ora\_aspnet\_Mem\_BasicAccess Role**

Service Method	Stored Procedure
GetNumberOfUsersOnline	ora_aspnet_Mem_GetNumOfUsersOn
GetPassword	ora_aspnet_Mem_GetPassword
GetUser	ora_aspnet_Mem_GetUserByUid ora_aspnet_Mem_GetUserByName
GetUserNameByEmail	ora_aspnet_Mem_GetUserByEml
UpdateUser	ora_aspnet_Mem_UpdateUser
ValidateUser	ora_aspnet_Mem_GetPwdWithFmt ora_aspnet_Mem_UpdateUserInfo

Table 1–11 lists the service methods and stored procedures that a user with the ora\_aspnet\_Mem\_ReportAccess role can execute.

**Table 1–11 ora\_aspnet\_Mem\_ReportAccess Role**

Service Method	Stored Procedure
FindUsersByEmail	ora_aspnet_Mem_FindUsersByEml
FindUsersByName	ora_aspnet_Mem_FindUsersByName
GetAllUsers	ora_aspnet_Mem_GetAllUsers
GetNumberOfUsersOnline	ora_aspnet_Mem_GetNumOfUsersOn
GetUser	ora_aspnet_Mem_GetUserByUid ora_aspnet_Mem_GetUserByName
GetUserNameByEmail	ora_aspnet_Mem_GetUserByEml

Table 1–12 lists the service methods and stored procedures that a user with the ora\_aspnet\_Mem\_FullAccess role can execute.

**Table 1–12 ora\_aspnet\_Mem\_FullAccess Role**

Service Method	Stored Procedure
All Membership methods	ora_aspnet_Mem_ChangePwdQAndA ora_aspnet_Mem_CreateUser ora_aspnet_Mem_FindUsersByEml ora_aspnet_Mem_FindUsersByName ora_aspnet_Mem_GetAllUsers ora_aspnet_Mem_GetNumOfUsersOn ora_aspnet_Mem_GetPassword ora_aspnet_Mem_GetPwdWithFmt ora_aspnet_Mem_GetUserByEml ora_aspnet_Mem_GetUserByName ora_aspnet_Mem_GetUserByUid ora_aspnet_Mem_ResetPassword ora_aspnet_Mem_SetPassword ora_aspnet_Mem_UnlockUser ora_aspnet_Mem_UpdateUser ora_aspnet_Mem_UpdateUserInfo ora_aspnet_Users_DeleteUser

### OracleRoleProvider Stored Procedures

Table 1–13 lists the service methods and stored procedures that a user with the ora\_aspnet\_Roles\_BasicAccess role can execute.

**Table 1–13 ora\_aspnet\_Roles\_BasicAccess Role**

Service Method	Stored Procedure
GetRolesForUser	ora_aspnet_UIR_GetRolesForUser
IsUserInRole	ora_aspnet_UIR_IsUserInRole

[Table 1–14](#) lists the service methods and stored procedures that a user with the `ora_aspnet_Roles_ReportAccess` role can execute.

**Table 1–14** *ora\_aspnet\_Roles\_ReportAccess Role*

Service Method	Stored Procedure
FindUsersInRole	ora_aspnet_UIR_FindUsersInRole
GetAllRoles	ora_aspnet_Roles_GetAllRoles
GetRolesForUser	ora_aspnet_UIR_GetRolesForUser
GetUsersInRole	ora_aspnet_UIR_GetUsersInRoles
IsUserInRole	ora_aspnet_UIR_IsUserInRole
RoleExists	ora_aspnet_Roles_RoleExists

[Table 1–15](#) lists the service methods and stored procedures that a user with the `ora_aspnet_Roles_FullAccess` role can execute.

**Table 1–15** *ora\_aspnet\_Roles\_FullAccess Role*

Service Method	Stored Procedure
All Role Manager methods	ora_aspnet_Roles_CreateRole
	ora_aspnet_Roles_DeleteRole
	ora_aspnet_Roles_GetAllRoles
	ora_aspnet_Roles_RoleExists
	ora_aspnet_UIR_AddUsersToRoles
	ora_aspnet_UIR_FindUsersInRole
	ora_aspnet_UIR_GetRolesForUser
	ora_aspnet_UIR_GetUsersInRoles
	ora_aspnet_UIR_IsUserInRole
	ora_aspnet_UIR_RemUsersFmRoles

## OracleProfileProvider Stored Procedures

[Table 1–16](#) lists the service methods and stored procedures that a user with the `ora_aspnet_Prof_BasicAccess` role can execute.

**Table 1–16** *ora\_aspnet\_Prof\_BasicAccess Role*

Service Method	Stored Procedure
GetPropertyValues	ora_aspnet_Prof_GetProperties
SetPropertyValues	ora_aspnet_Prof_SetProperties

[Table 1–17](#) lists the service methods and stored procedures that a user with the `ora_aspnet_Prof_ReportAccess` role can execute.

**Table 1–17** *ora\_aspnet\_Prof\_ReportAccess Role*

Service Method	Stored Procedure
GetAllProfiles	ora_aspnet_Prof_GetProfiles
GetAllInactiveProfiles	ora_aspnet_Prof_GetProfiles

**Table 1–17 (Cont.) ora\_aspnet\_Prof\_ReportAccess Role**

Service Method	Stored Procedure
GetNumberOfInactiveProfiles	ora_aspnet_Prof_GetNumOfInactPf
FindProfilesByUserName	ora_aspnet_Prof_GetProfiles
FindInactiveProfilesByUserName	ora_aspnet_Prof_GetProfiles

Table 1–18 lists the service methods and stored procedures that a user with the ora\_aspnet\_Prof\_FullAccess role can execute.

**Table 1–18 ora\_aspnet\_Prof\_FullAccess Role**

Service Method	Stored Procedure
All Profile methods	ora_aspnet_Prof_DeleteInactPf
	ora_aspnet_Prof_DeleteProfiles
	ora_aspnet_Prof_GetNumOfInactPf
	ora_aspnet_Prof_GetProfiles
	ora_aspnet_Prof_GetProperties
	ora_aspnet_Prof_SetProperties

### OraclePersonalizationProvider Stored Procedures

Table 1–19 lists the service methods and stored procedures that a user with the ora\_aspnet\_Pers\_BasicAccess role can execute.

**Table 1–19 ora\_aspnet\_Pers\_BasicAccess Role**

Service Method	Stored Procedure
LoadPersonalizationState	ora_aspnet_PPU_GetPgSettings
	ora_aspnet_PAU_GetPgSettings
ResetPersonalizationState	ora_aspnet_PPU_ResetPgSettings
	ora_aspnet_PAU_ResetPgSettings
SavePersonalizationState	ora_aspnet_PPU_SetPgSettings
	ora_aspnet_PAU_SetPgSettings

Table 1–20 lists the service methods and stored procedures that a user with the ora\_aspnet\_Pers\_ReportAccess role can execute.

**Table 1–20 ora\_aspnet\_Pers\_ReportAccess Role**

Service Method	Stored Procedure
FindState	ora_aspnet_PA_FindState
GetCountOfState	ora_aspnet_PA_GetCountOfState

Table 1–21 lists the service methods and stored procedures that a user with the ora\_aspnet\_Pers\_FullAccess role can execute.

**Table 1–21 ora\_aspnet\_Pers\_FullAccess Role**

Service Method	Stored Procedure
All Personalization methods	ora_aspnet_PA_FindState
	ora_aspnet_PA_GetCountOfState
	ora_aspnet_PA_ResetSharedState
	ora_aspnet_PA_ResetUserState
	ora_aspnet_PAU_GetPgSettings
	ora_aspnet_PAU_ResetPgSettings
	ora_aspnet_PAU_SetPgSettings
	ora_aspnet_PPU_GetPgSettings
	ora_aspnet_PPU_ResetPgSettings
	ora_aspnet_PPU_SetPgSettings

### OracleWebEventProvider Stored Procedures

Table 1–22 lists the service methods and stored procedures that a user with the ora\_aspnet\_Wevnt\_FullAccess role can execute.

**Table 1–22 ora\_aspnet\_Wevnt\_FullAccess Role**

Service Method	Stored Procedure
All Web Event methods	ora_aspnet_LogWebEvents

### OracleSiteMapProvider Stored Procedures

Table 1–23 lists the service methods and stored procedures that a user with the ora\_aspnet\_Smap\_FullAccess role can execute.

**Table 1–23 ora\_aspnet\_Smap\_FullAccess Role**

Service Method	Stored Procedure
All Site Map methods	ora_aspnet_GetSiteMapData

### OracleSessionStateStore Provider Stored Procedures

Table 1–24 lists the service methods and stored procedures that a user with the ora\_aspnet\_Sessn\_FullAccess role can execute.

**Table 1–24 ora\_aspnet\_Sessn\_FullAccess Role**

Service Method	Stored Procedure
All Session State methods	ora_aspnet_SessnApp_SetAppID
	ora_aspnet_Sessn_InsUninitItem
	ora_aspnet_Sessn_RelStateItmEx
	ora_aspnet_Sessn_RmStateItem
	ora_aspnet_Sessn_ResetTimeout
	ora_aspnet_Sessn_UpdStateItem
	ora_aspnet_Sessn_InsStateItem
	ora_aspnet_Sessn_GetStateItem

**Table 1–24 (Cont.) ora\_aspnet\_Sessn\_FullAccess Role**

<b>Service Method</b>	<b>Stored Procedure</b>
	ora_aspnet_Sessn_GetStateItmEx

## Synonyms

Public synonyms are created for all stored procedures so that they can be executed by any user in the database who is granted proper provider-specific roles by the user that owns the stored procedures.

---

---

## OracleMembershipProvider

This chapter describes the `OracleMemberProvider` class.

**See Also:** ASP.NET membership and membership providers  
<http://msdn.microsoft.com/en-us/library/tw292whz.aspx>

This chapter contains the following topic:

- [OracleMembershipProvider Class](#)

---

## OracleMembershipProvider Class

The `OracleMembershipProvider` class enables ASP.NET developers to store Web site user account information in an Oracle database.

### Class Inheritance

`System.Object`

`System.Configuration.Provider.ProviderBase`

`System.Web.Security.MembershipProvider`

`Oracle.Web.Security.OracleMembershipProvider`

### Declaration

```
// C#  
public class OracleMembershipProvider: MembershipProvider
```

### Thread Safety

All public static methods are thread-safe, although instance members are not guaranteed to be thread-safe.

### Remarks

This class allows ASP.NET applications to store and manage user information in an Oracle database.

Note that the term user in this chapter refers to an application or user, not a database user. Thus, the user information that this provider manages is application or user information, not database user information.

### Example

The following code example shows a `web.config` file for an ASP.NET application configured to use `OracleMembershipProvider` as the default provider. This configuration uses the connection string and default attribute values specified in the `machine.config` file.

```
<?xml version="1.0"?>  
<configuration xmlns="http://schemas.microsoft.com/.NetConfiguration/v2.0">  
  <system.web>  
    <membership defaultProvider="OracleMembershipProvider"/>  
  </system.web>  
</configuration>
```

The following is a `web.config` example for an ASP.NET application that uses an `OracleMembershipProvider` with customized settings and an application-specific connection string:

```
<?xml version="1.0"?>  
<configuration xmlns=  
  "http://schemas.microsoft.com/.NetConfiguration/v2.0">  
  <connectionStrings>  
    <add name="my_membership_app_con_string" connectionString=  
      "User Id=scott;Password=tiger;Data Source=Oracle"/>  
  </connectionStrings>  
  <system.web>  
    <!-- Enable and customize OracleMembershipProvider settings -->
```



```
<membership defaultProvider="MyOracleMembershipProvider">
  <providers>
    <add name="MyOracleMembershipProvider"
      type="Oracle.Web.Security.OracleMembershipProvider,
      Oracle.Web, Version=2.112.2.0, Culture=neutral,
      PublicKeyToken=89b483f429c47342"
      connectionStringName="my_membership_app_con_string"
      applicationName="my_membership_app"
      enablePasswordRetrieval="false"
      enablePasswordReset="true"
      requiresQuestionAndAnswer="true"
      requiresUniqueEmail="true"
      passwordFormat="Hashed"
      maxInvalidPasswordAttempts="4"
      minRequiredPasswordLength="9"
      passwordAttemptWindow="8" />
  </providers>
</membership>
</system.web>
</configuration>
```

Note that the `applicationName` attribute should be set to a unique value for each ASP.NET application.

### Requirements

Namespace: Oracle.Web.Security

Assembly: Oracle.Web.dll

Oracle Providers for ASP.NET Version: Oracle Providers for ASP.NET 2.0 and Oracle Providers for ASP.NET 4

### See Also:

- ["Oracle Providers for ASP.NET Assembly" on page 1-4](#)
- [OracleMembershipProvider Members](#)
- [OracleMembershipProvider Constructors](#)
- [OracleMembershipProvider Static Methods](#)
- [OracleMembershipProvider Public Properties](#)
- [OracleMembershipProvider Public Methods](#)
- [OracleMembershipProvider Public Events](#)

## OracleMembershipProvider Members

OracleMembershipProvider members are listed in the following tables.

### OracleMembershipProvider Constructors

The OracleMembershipProvider constructor is listed in [Table 2-1](#).

**Table 2-1 OracleMembershipProvider Constructor**

Constructor	Description
<a href="#">OracleMembershipProvider Constructors</a>	Instantiates a new instance of the OracleMembershipProvider class

### OracleMembershipProvider Static Methods

OracleMembershipProvider static methods are listed in [Table 2-2](#).

**Table 2-2 OracleMembershipProvider Static Methods**

Static Methods	Description
Equals	Inherited from System.Object (Overloaded)
ReferenceEquals	Inherited from System.Object

### OracleMembershipProvider Public Properties

OracleMembershipProvider public properties are listed in [Table 2-3](#).

**Table 2-3 OracleMembershipProvider Public Properties**

Public Properties	Description
<a href="#">ApplicationName</a>	Gets or sets the name of the application that is used to group user information
<a href="#">CommandTimeout</a>	Gets the number of seconds that the command is allowed to execute before it is terminated with an exception
Description	Inherited from System.Configuration.Provider.Providerbase
<a href="#">EnablePasswordReset</a>	Indicates whether the membership provider is configured to allow users to reset their passwords
<a href="#">EnablePasswordRetrieval</a>	Indicates whether the membership provider is configured to allow users to retrieve their passwords
<a href="#">MaxInvalidPasswordAttempts</a>	Gets the number of invalid password or password-answer attempts allowed before the user is locked out
<a href="#">MinRequiredNonAlphanumericCharacters</a>	Gets the minimum number of special characters that must be present in a valid password
<a href="#">MinRequiredPasswordLength</a>	Gets the minimum length required for a password
Name	Inherited from System.Configuration.Provider.Providerbase

**Table 2–3 (Cont.) OracleMembershipProvider Public Properties**

Public Properties	Description
<a href="#">PasswordAttemptWindow</a>	Gets the number of minutes in which a maximum number of invalid password or password-answer attempts are allowed before the user is locked out
<a href="#">PasswordCompatMode</a>	Gets the password compatibility mode.
<a href="#">PasswordFormat</a>	Gets a value indicating the format for storing passwords in the membership data source
<a href="#">PasswordStrengthRegularExpression</a>	Gets the regular expression used to evaluate a password
<a href="#">RequiresQuestionAndAnswer</a>	Gets a value indicating whether the membership provider is configured in such a way that it requires the user to answer a password question for password reset and retrieval
<a href="#">RequiresUniqueEmail</a>	Gets a value indicating whether the membership provider is configured to require a unique e-mail address for each user name

**OracleMembershipProvider Public Methods**

OracleMembershipProvider public methods are listed in [Table 2–4](#).

**Table 2–4 OracleMembershipProvider Public Methods**

Public Methods	Description
<a href="#">ChangePassword</a>	Updates the password for a user
<a href="#">ChangePasswordQuestionAndAnswer</a>	Updates the password question and answer for a user
<a href="#">CreateUser</a>	Adds a new user to the database
<a href="#">DeleteUser</a>	Removes a user from the database
<a href="#">Equals</a>	Inherited from <code>System.Object</code> (Overloaded)
<a href="#">FindUsersByEmail</a>	Returns a collection of users whose e-mail addresses match the specified e-mail address
<a href="#">FindUsersByName</a>	Returns a collection of users that match the specified user name
<a href="#">GeneratePassword</a>	Generates a random password that is at least 14 characters in length
<a href="#">GetAllUsers</a>	Returns a collection of all the users in the database
<a href="#">GetHashCode</a>	Inherited from <code>System.Object</code>
<a href="#">GetNumberOfUsersOnline</a>	Returns the number of users that are currently accessing the application
<a href="#">GetPassword</a>	Returns the password for the specified user name from the database
<a href="#">GetType</a>	Inherited from <code>System.Object</code>
<a href="#">GetUser</a>	Returns user information from the database based on the unique identifier for the user (Overloaded)
<a href="#">GetUserNameByEmail</a>	Returns the user name associated with the specified e-mail address

**Table 2–4 (Cont.) OracleMembershipProvider Public Methods**

Public Methods	Description
<a href="#">Initialize</a>	Initializes the <code>OracleMembership</code> provider with the property values specified in the ASP.NET application configuration file ( <code>web.config</code> )
<a href="#">ResetPassword</a>	Resets a user's password and returns a new automatically generated password
<code>ToString</code>	Inherited from <code>System.Object</code>
<a href="#">UnlockUser</a>	Unlocks a user so that the user can be validated
<a href="#">UpdateUser</a>	Updates information about a user in the database
<a href="#">ValidateUser</a>	Validates the user

**OracleMembershipProvider Public Events**

`OracleMembershipProvider` public event is listed in [Table 2–5](#).

**Table 2–5 OracleMembershipProvider Public Event**

Public Event	Description
<code>ValidatingPassword</code>	Inherited from <code>System.Web.Security.MembershipProvider</code>

**See Also:**

- [OracleMembershipProvider Class](#)

## OracleMembershipProvider Constructors

This constructor instantiates a new instance of the `OracleMembershipProvider` class.

### Overload List:

- [OracleMembershipProvider\(\)](#)

This constructor creates an instance of the `OracleMembershipProvider` class.

#### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleMembershipProvider Class](#)
- [OracleMembershipProvider Members](#)

### OracleMembershipProvider()

This constructor instantiates a new instance of the `OracleMembershipProvider` class.

#### Declaration

```
// C#  
public OracleMembershipProvider();
```

#### Remarks

ASP.NET calls the `OracleMembershipProvider` constructor to create an instance of the `OracleMembershipProvider` class, as specified in the configuration for the application. Initialization values for the `OracleMembershipProvider` object are passed through the `Initialize` method.

This constructor is not intended to be used directly by the application.

#### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleMembershipProvider Class](#)
- [OracleMembershipProvider Members](#)

## OracleMembershipProvider Static Methods

OracleMembershipProvider static methods are listed in [Table 2–6](#).

**Table 2–6 OracleMembershipProvider Static Methods**

Static Methods	Description
Equals	Inherited from System.Object (Overloaded)
ReferenceEquals	Inherited from System.Object

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleMembershipProvider Class](#)
- [OracleMembershipProvider Members](#)

## OracleMembershipProvider Public Properties

OracleMembershipProvider public properties are listed in [Table 2-7](#).

**Table 2-7 OracleMembershipProvider Public Properties**

Public Properties	Description
<a href="#">ApplicationName</a>	Gets or sets the name of the application that is used to group user information
<a href="#">CommandTimeout</a>	Gets the number of seconds that the command is allowed to execute before it is terminated with an exception
Description	Inherited from <code>System.Configuration.Provider.ProviderBase</code>
<a href="#">EnablePasswordReset</a>	Indicates whether the membership provider is configured to allow users to reset their passwords
<a href="#">EnablePasswordRetrieval</a>	Indicates whether the membership provider is configured to allow users to retrieve their passwords
<a href="#">MaxInvalidPasswordAttempts</a>	Gets the number of invalid password or password-answer attempts allowed before the user is locked out
<a href="#">MinRequiredNonAlphanumericCharacters</a>	Gets the minimum number of special characters that must be present in a valid password
<a href="#">MinRequiredPasswordLength</a>	Gets the minimum length required for a password
Name	Inherited from <code>System.Configuration.Provider.ProviderBase</code>
<a href="#">PasswordAttemptWindow</a>	Gets the number of minutes in which a maximum number of invalid password or password-answer attempts are allowed before the user is locked out
<a href="#">PasswordCompatMode</a>	Gets the password compatibility mode.
<a href="#">PasswordFormat</a>	Gets a value indicating the format for storing passwords in the membership data source
<a href="#">PasswordStrengthRegularExpression</a>	Gets the regular expression used to evaluate a password
<a href="#">RequiresQuestionAndAnswer</a>	Gets a value indicating whether the membership provider is configured in such a way that it requires the user to answer a password question for password reset and retrieval
<a href="#">RequiresUniqueEmail</a>	Gets a value indicating whether the membership provider is configured to require a unique e-mail address for each user name

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleMembershipProvider Class](#)
- [OracleMembershipProvider Members](#)

## ApplicationName

This property gets or sets the name of the application that is used to group user information.

**Declaration**

```
// C#  
public override string ApplicationName {get; set;}
```

**Property Value**

The name of the application. If the `applicationName` attribute is not specified in the application configuration file, or if the value is an empty string, then this property is set to the application virtual path.

**Exceptions**

`ArgumentException` - The application name supplied is an empty string or a null reference.

`ProviderException` - The application name supplied exceeds 256 characters.

**Remarks**

The string value of the `ApplicationName` property is used for organizing user information. Multiple ASP.NET applications can use the same database and create duplicate user names because user information is stored uniquely for each application name. This property can be set programmatically, or it can be set declaratively in the Web application configuration file using the `applicationName` attribute. The attribute name in the configuration file is case-sensitive.

The `ApplicationName` property is not thread-safe. It is recommended that the programming code not allow users to set the `ApplicationName` property in Web applications.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleMembershipProvider Class](#)
- [OracleMembershipProvider Members](#)

## CommandTimeout

This property gets the number of seconds that the command is allowed to execute before it is terminated with an exception.

**Declaration**

```
// C#  
public int CommandTimeout {get;}
```



**Property Value**

An int.

**Remarks**

To customize a provider, ASP.NET developers can set an integer value for this property through the `web.config` file using the `commandTimeout` attribute.

The default value is 30 seconds. The attribute name in the configuration file is case-sensitive.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleMembershipProvider Class](#)
- [OracleMembershipProvider Members](#)

**EnablePasswordReset**

This property indicates whether the membership provider is configured to allow users to reset their passwords.

**Declaration**

```
// C#  
public override bool EnablePasswordReset{get;}
```

**Property Value**

Returns `true`, if the membership provider supports password reset; otherwise, it returns `false`. The default is `true`.

**Remarks**

To customize the membership provider, ASP.NET developers can specify a Boolean value for this property through the `web.config` file using the `enablePasswordReset` attribute. The value indicates whether users can use the `ResetPassword` method to overwrite their current password with a new, randomly generated password. The attribute name in the configuration file is case-sensitive.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleMembershipProvider Class](#)
- [OracleMembershipProvider Members](#)

**EnablePasswordRetrieval**

This property indicates whether the membership provider is configured to allow users to retrieve their passwords.

**Declaration**

```
// C#  
public override bool EnablePasswordRetrieval{get;}
```

**Property Value**

Returns `true`, if the membership provider is configured to support password retrieval; otherwise, returns `false`. The default is `false`.

**Remarks**

To customize a membership provider, ASP.NET developers can set a Boolean value for this property through the `web.config` file using the `enablePasswordRetrieval` attribute. The value indicates whether users can use the `GetPassword` method to retrieve their current password from the database. The attribute name in the configuration file is case-sensitive.

If the custom membership provider supports hashed passwords, then the `GetPassword` method returns an exception if the `EnablePasswordRetrieval` property is set to `true` and the password format is set to `Hashed`. In other words, hashed passwords cannot be retrieved.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleMembershipProvider Class](#)
- [OracleMembershipProvider Members](#)

**MaxInvalidPasswordAttempts**

This property gets the number of invalid password or password-answer attempts allowed before the user is locked out.

**Declaration**

```
// C#  
public override int MaxInvalidPasswordAttempts{get;}
```

**Property Value**

The number of invalid password or password-answer attempts allowed before the user is locked out. The default number of attempts is 5.

**Remarks**

To customize a membership provider, ASP.NET developers can set an integer value for this property through the `web.config` file using the `maxInvalidPasswordAttempts` attribute. The attribute name in the configuration file is case-sensitive.

The `MaxInvalidPasswordAttempts` property works in conjunction with the `PasswordAttemptWindow` property. If the number of invalid passwords or password question entries is greater than or equal to the `MaxInvalidPasswordAttempts` property value within the `PasswordAttemptWindow` property value (in minutes), then the user is locked out until the user is unlocked by the `UnlockUser` method. If a valid password or password answer is supplied before the `MaxInvalidPasswordAttempts` value is reached, then the counter that tracks the number of invalid attempts is reset to zero.

Invalid passwords and password-answer attempts accumulate independently. For example, if the `MaxInvalidPasswordAttempts` property is set to 10, and 6 invalid password attempts are made followed by 3 invalid password-answer attempts, 4 more invalid password attempts or 7 more invalid password-answer attempts must be made within the `PasswordAttemptWindow` for the user to be locked out.

If the `RequiresQuestionAndAnswer` property is set to `false`, invalid password-answer attempts are not tracked.

Invalid password and password-answer attempts are tracked in the `ValidateUser`, `ChangePassword`, `ChangePasswordQuestionAndAnswer`, `GetPassword`, and `ResetPassword` methods.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleMembershipProvider Class](#)
- [OracleMembershipProvider Members](#)

## MinRequiredNonAlphanumericCharacters

This property gets the minimum number of special characters that must be present in a valid password.

**Declaration**

```
// C#  
public override int MinRequiredNonAlphanumericCharacters {get;}
```

**Property Value**

The minimum number of special characters that must be present in a valid password. The default value is 1.

**Remarks**

To customize a membership provider, ASP.NET developers can set an integer value for this property through the `web.config` file using the `minRequiredNonAlphanumericCharacters` attribute. The attribute name in the configuration file is case-sensitive.

The `MinRequiredNonAlphanumericCharacters` property returns the minimum number of special, nonalphabetic characters that must be entered to create a valid password for the `OracleMembershipProvider` object.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleMembershipProvider Class](#)
- [OracleMembershipProvider Members](#)

## MinRequiredPasswordLength

This property gets the minimum length required for a password.

**Declaration**

```
// C#  
public override int MinRequiredPasswordLength {get;}
```

**Property Value**

The minimum length required for a password. The default value is 7.

**Remarks**

To customize a membership provider, ASP.NET developers can set an integer value for this property through the `web.config` file using the `minRequiredPasswordLength` attribute. The attribute name in the configuration file is case-sensitive.

The `minRequiredPasswordLength` property gets the minimum number of characters that must be entered to create a valid password for the `OracleMembershipProvider` object.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleMembershipProvider Class](#)
- [OracleMembershipProvider Members](#)

**PasswordAttemptWindow**

This property gets the number of minutes in which a maximum number of invalid password or password-answer attempts are allowed before the user is locked out.

**Declaration**

```
// C#  
public override int PasswordAttemptWindow{get;}
```

**Property Value**

The number of minutes in which a maximum number of invalid password or password-answer attempts are allowed before the user is locked out. The default value is 10.

**Remarks**

To customize a membership provider, ASP.NET developers can set an integer value for this property through the `web.config` file using the `passwordAttemptWindow` attribute. The attribute name in the configuration file is case-sensitive.

The `PasswordAttemptWindow` property works in conjunction with the `MaxInvalidPasswordAttempts` property. If the number of invalid passwords or password question entries is greater than or equal to the `MaxInvalidPasswordAttempts` property value within the `PasswordAttemptWindow` property value (in minutes), then the user is locked out until the user is unlocked by the `UnlockUser` method. If a valid password or password answer is supplied before the `MaxInvalidPasswordAttempts` value is reached, then the counter that tracks the number of invalid attempts is reset to zero.

Invalid password and password-answer attempts accumulate independently. For example, if the `MaxInvalidPasswordAttempts` property is set to 10, and 6 invalid password attempts are made followed by 3 invalid password-answer attempts, 4 more invalid password attempts or 7 more invalid password-answer attempts must be made within the `PasswordAttemptWindow` value for the user to be locked out.

If the `RequiresQuestionAndAnswer` property is set to `false`, then invalid password-answer attempts are not tracked.

Invalid password and password-answer attempts are tracked in the `ValidateUser`, `ChangePassword`, `ChangePasswordQuestionAndAnswer`, `GetPassword`, and `ResetPassword` methods.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly" on page 1-4](#)
- [OracleMembershipProvider Class](#)
- [OracleMembershipProvider Members](#)

**PasswordCompatMode**

This property gets the password compatibility mode.

**Declaration**

```
// C#
public string PasswordCompatMode {get;}
```

**Property Value**

A string.

**Remarks**

The default value is Framework20. The other acceptable value is Framework40. The string value is case-sensitive.

To customize a provider, ASP.NET developers can set a string value for this property through the web.config file using the case-sensitive passwordCompatMode attribute.

When passwordFormat attribute is set to Hashed, the value of System.Web.Security.Membership.HashAlgorithmType property is used to hash password for a Membership user during the creation and validation of the user.

The value for HashAlgorithmType property can be set in the web.config file through the case-sensitive attribute hashAlgorithmType, as in the following example:

```
<membership defaultProvider="OracleMembershipProvider" hashAlgorithmType="SHA1"/>
```

If hashAlgorithmType attribute is not specified in the web.config file, SHA1 will be used. With .NET Framework 2.0, the other valid value for hashAlgorithmType is MD5.

With .NET Framework 4, if a new application that does not have existing Membership users and would like to use one of the other variants of SHA and HMACSHA hash algorithm types, the passwordCompatMode attribute must be set to Framework40 and the hashAlgorithmType attribute must be set to the desired type, such as SHA256, HMACSHA256, HMACSHA384, or HMACSHA512. Nevertheless, SHA1 and MD5 are still supported when passwordCompatMode is set to Framework40.

**Example**

The following is a web.config example that sets hashAlgorithmType to HMACSHA25 and passwordCompatMode to Framework40.

```
<!-- Enable and customize OracleMembershipProvider settings -->
<membership defaultProvider="MyOracleMembershipProvider"
hashAlgorithmType="HMACSHA256">
  <providers>
    <add name="MyOracleMembershipProvider"
type="Oracle.Web.Security.OracleMembershipProvider, Oracle.Web,
Version=4.112.2.0, Culture=neutral, PublicKeyToken=89b483f429c47342"
ConnectionStringName="my_membership_app_con_string"
applicationName="my_membership_app"
enablePasswordRetrieval="false"
```

```
enablePasswordReset="true"
requiresQuestionAndAnswer="true"
requiresUniqueEmail="true"
passwordFormat="Hashed"
maxInvalidPasswordAttempts="4"
minRequiredPasswordLength="9"
    passwordCompatMode="Framework40"
passwordAttemptWindow="8" />
</providers>
</membership>
```

Once one `hashAlgorithmType` is used to create a Membership user, the same `hashAlgorithmType` must be used to validate the user. If `hashAlgorithmType` is changed, the user will not be validated successfully. Thus, the same `hashAlgorithmType` must be used for a given application during its lifetime.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleMembershipProvider Class](#)
- [OracleMembershipProvider Members](#)

## PasswordFormat

This property gets a value indicating the format for storing passwords in the membership data source.

**Declaration**

```
// C#
public override MembershipPasswordFormat PasswordFormat {get;}
```

**Property Value**

The format for storing passwords in the data source. The format can be any one of the `MembershipPasswordFormat` values, such as `Clear`, `Hashed`, or `Encrypted`. The default value is `Hashed`.

**Remarks**

To customize a membership provider, ASP.NET developers can specify a `MembershipPasswordFormat` enumerated value for this property through the `web.config` file using the `passwordFormat` attribute. The attribute name in the configuration file is case-sensitive.

The `PasswordFormat` property indicates that passwords are stored in any one of the following formats: `Clear`, `Encrypted`, or `Hashed`. The format name is case-sensitive. For example, `Clear` is valid, but `clear` is invalid.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleMembershipProvider Class](#)
- [OracleMembershipProvider Members](#)

## PasswordStrengthRegularExpression

This property gets the regular expression used to evaluate a password.

**Declaration**

```
// C#
public override string PasswordStrengthRegularExpression{get;}
```

**Property Value**

The regular expression that is used to evaluate a password. The default is an empty string.

**Remarks**

To customize a membership provider, ASP.NET developers can set a string value for this property through the `web.config` file using the `passwordStrengthRegularExpression` attribute. The attribute name in the configuration file is case-sensitive.

The `PasswordStrengthRegularExpression` property gets the regular expression as criteria to evaluate the password. If the password does not meet the criteria, it is not accepted by the membership provider.

Consider the following example:

```
passwordStrengthRegularExpression="(?.{7,})(?=(.*\d){1,})(?=(.*\W){1,})"
```

The code in the preceding example validates passwords against the following criteria:

- Has at least 7 characters.
- Contains at least 1 digit.
- Contains at least 1 special (nonalphanumeric) character.

The minimum password length defined in `passwordStrengthRegularExpression` must be equal to or greater than the value of the `minRequiredPasswordLength` attribute.

The minimum number of special (nonalphanumeric) characters defined in the `passwordStrengthRegularExpression` attribute must be equal to or greater than the value of the `minRequiredNonalphanumericCharacters` attribute.

The `passwordStrengthRegularExpression` attribute is not used in automatically generated passwords from the `ResetPassword` method.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleMembershipProvider Class](#)
- [OracleMembershipProvider Members](#)

**RequiresQuestionAndAnswer**

This property gets a value indicating whether the membership provider is configured in such a way that it requires the user to answer a password question for password reset and retrieval.

**Declaration**

```
// C#
public override bool RequiresQuestionAndAnswer{get;}
```

**Property Value**

Returns `true`, if a password answer is required for password reset and retrieval; otherwise, returns `false`. The default value is `true`.

**Remarks**

To customize a membership provider, ASP.NET developers can set a Boolean value for this property through the `web.config` file by using the `requiresQuestionAndAnswer` attribute. The value indicates whether users must supply a password answer in order to retrieve their password using the `GetPassword` method, or reset their password using the `ResetPassword` method. The attribute name in the configuration file is case-sensitive.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleMembershipProvider Class](#)
- [OracleMembershipProvider Members](#)

**RequiresUniqueEmail**

This property gets a value indicating whether the membership provider is configured to require a unique e-mail address for each user name.

**Declaration**

```
// C#  
public override bool RequiresUniqueEmail{get;}
```

**Property Value**

Returns `true`, if the membership provider requires a unique e-mail address; otherwise, returns `false`. The default value is `false`.

**Remarks**

To customize a membership provider, ASP.NET developers can specify a Boolean value for the `RequiresUniqueEmail` property through the `web.config` file using the `requiresUniqueEmail` attribute. The attribute name in the configuration file is case-sensitive.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleMembershipProvider Class](#)
- [OracleMembershipProvider Members](#)



## OracleMembershipProvider Public Methods

OracleMembershipProvider public methods are listed in [Table 2–8](#).

**Table 2–8 OracleMembershipProvider Public Methods**

Public Methods	Description
<a href="#">ChangePassword</a>	Updates the password for a user
<a href="#">ChangePasswordQuestionAndAnswer</a>	Updates the password question and answer for a user
<a href="#">CreateUser</a>	Adds a new user to the database
<a href="#">DeleteUser</a>	Removes a user from the database
Equals	Inherited from <code>System.Object</code> (Overloaded)
<a href="#">FindUsersByEmail</a>	Returns a collection of users whose e-mail addresses match the specified e-mail address
<a href="#">FindUsersByName</a>	Returns a collection of users that match the specified user name
<a href="#">GeneratePassword</a>	Generates a random password that is at least 14 characters in length
<a href="#">GetAllUsers</a>	Returns a collection of all the users in the database
GetHashCode	Inherited from <code>System.Object</code>
<a href="#">GetNumberOfUsersOnline</a>	Returns the number of users that are currently accessing the application
<a href="#">GetPassword</a>	Returns the password for the specified user name from the database
GetType	Inherited from <code>System.Object</code>
<a href="#">GetUser</a>	Returns user information from the database based on the unique identifier for the user (Overloaded)
<a href="#">GetUserNameByEmail</a>	Returns the user name associated with the specified e-mail address
<a href="#">Initialize</a>	Initializes the OracleMembership provider with the property values specified in the ASP.NET application configuration file ( <code>web.config</code> )
<a href="#">ResetPassword</a>	Resets a user's password and returns a new automatically generated password
ToString	Inherited from <code>System.Object</code>
<a href="#">UnlockUser</a>	Unlocks a user so that the user can be validated
<a href="#">UpdateUser</a>	Updates information about a user in the database
<a href="#">ValidateUser</a>	Validates the user

### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleMembershipProvider Class](#)
- [OracleMembershipProvider Members](#)

## ChangePassword

This method updates the password for a user.

### Declaration

```
// C#  
public override bool ChangePassword(string userName, string oldPassword,  
    string newPassword);
```

### Parameters

- *userName*  
The user to update the password for.
- *oldPassword*  
The current password for the specified user.
- *newPassword*  
The new password for the specified user.

### Return Value

Returns true if the password was updated successfully; otherwise, returns false.

### Exceptions

*ArgumentNullException* - The *userName*, *oldPassword*, or *newPassword* parameter is null.

*System.Web.Security.MembershipPasswordException* - *userName* was not found in the membership database.

*System.Configuration.Provider.ProviderException* - An error occurred when setting the new password in the database.

*Exception* - An unhandled exception has occurred.

*ArgumentException* - One of the following conditions exists:

- The *userName* parameter is an empty string, contains a comma, or is longer than 256 characters.
- The *oldPassword* parameter is an empty string or is longer than 128 characters.
- The *newPassword* parameter is an empty string, is longer than 128 characters (including the encoded version), is less than the value of the *MinRequiredPasswordLength* property, has a number of nonalphanumeric characters less than the value of *MinRequiredNonAlphanumericCharacters* property, or does not match the regular expression defined in the *PasswordStrengthRegularExpression* property.
- The change-password operation was canceled by a subscriber to the *ValidatingPassword* event, and the *FailureInformation* property was a null reference.

### Remarks

The *ChangePassword* method returns true if the supplied user name and password are valid and the password was updated successfully; otherwise, it returns false.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly" on page 1-4](#)
- [OracleMembershipProvider Class](#)
- [OracleMembershipProvider Members](#)

**ChangePasswordQuestionAndAnswer**

This method updates the password question and answer for a user.

**Declaration**

```
// C#  
public override bool ChangePasswordQuestionAndAnswer(string userName, string  
    password, string newPasswordQuestion, string newPasswordAnswer);
```

**Parameters**

- *userName*  
The user that the password question and answer change for.
- *password*  
The password for the specified user.
- *newPasswordQuestion*  
The new password question for the specified user.
- *newPasswordAnswer*  
The new password answer for the specified user.

**Return Value**

Returns true, if the password question and answer were updated successfully; false, if otherwise.

**Exceptions**

ArgumentException - One of the following conditions exists:

- The *userName* parameter is an empty string, contains a comma, or is longer than 256 characters.
- The *password* parameter is an empty string or is longer than 128 characters.
- The *newPasswordQuestion* parameter is an empty string or is longer than 256 characters.
- The *newPasswordAnswer* parameter is an empty string or is longer than 128 characters (including the encoded version).

**Remarks**

If the user name and password supplied are valid and the password question and answer were updated successfully, then this method returns true. Otherwise, it returns false.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleMembershipProvider Class](#)
- [OracleMembershipProvider Members](#)

## CreateUser

This method adds a new user to the database.

**Declaration**

```
// C#  
public override MembershipUser CreateUser(string userName, string password,  
    string emailAddress, string passwordQuestion, string passwordAnswer, bool  
    isApproved, Object providerUserKey, out MembershipCreateStatus status);
```

**Parameters**

- *userName*  
The user name for the new user.
- *password*  
The password for the new user.
- *emailAddress*  
The email address for the new user.
- *passwordQuestion*  
The password question for the new user.
- *passwordAnswer*  
The password answer for the new user.
- *isApproved*  
A Boolean value that indicates whether the new user is approved to be validated.
- *providerUserKey*  
The unique identifier from the database for the new user.
- *status*  
A MembershipCreateStatus enumeration value indicating whether the user was created successfully.

**Return Value**

A MembershipUser object populated with the information for the newly created user.

**Remarks**

This method returns a MembershipUser object populated with the information for the newly created user. The *status* parameter returns a MembershipCreateStatus value that indicates whether the user was successfully created. If the CreateUser method failed, a MembershipCreateStatus member indicates the cause of the failure.

## MembershipCreateStatus Enumeration

The MembershipCreateStatus enumeration values are listed in [Table 2-9](#).

**Table 2-9 MembershipCreateStatus Enumeration Values**

Member Name	Description
DuplicateEmail	The e-mail address for the application already exists in the database.
DuplicateProviderUserKey	The provider user key for the application already exists in the database.
DuplicateUserName	The user name for the application already exists in the database.
InvalidAnswer	The password answer is not formatted correctly.
InvalidEmail	The e-mail address is not formatted correctly.
InvalidPassword	The password is not formatted correctly.
InvalidProviderUserKey	The provider user key is of an invalid type or format.
InvalidQuestion	The password question is not formatted correctly.
InvalidUserName	The user name was not found in the database.
ProviderError	The provider returned an error that is not described by other MembershipCreateStatus enumeration values.
Success	The user was successfully created.
UserRejected	The user was not created, for a reason defined by the provider.

### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleMembershipProvider Class](#)
- [OracleMembershipProvider Members](#)

## DeleteUser

This method removes a user from the database.

### Declaration

```
// C#
public override bool DeleteUser(string userName, bool deleteAllRelatedData);
```

### Parameters

- *userName*  
The name of the user to delete.
- *deleteAllRelatedData*  
A Boolean value that indicates whether all the data related to the user is to be removed from the database.

### Return Value

Returns `true`, if the user was successfully deleted; `false`, if otherwise or if the user does not exist in the database.

**Exceptions**

*ArgumentException* - The *userName* parameter is an empty string, contains a comma, or is longer than 256 characters.

*ArgumentNullException* - The *userName* parameter is a null reference.

**Remarks**

Leading and trailing spaces are trimmed from the *userName* parameter value. If *deleteAllRelatedData* is true, then all data related to the user in the database such as, data for roles, profiles, and personalization, are also deleted, even if the user does not exist in the Oracle membership database.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleMembershipProvider Class](#)
- [OracleMembershipProvider Members](#)

**FindUsersByEmail**

This method returns a collection of users whose e-mail addresses match the specified e-mail address.

**Declaration**

```
// C#
public override MembershipUserCollection FindUsersByEmail(string emailToMatch,
    int pageIndex, int pageSize, out int totalRecords);
```

**Parameters**

- *emailToMatch*  
The email address to search for.
- *pageIndex*  
The index of the page of results to return. The *PageIndex* is zero-based.
- *pageSize*  
The size of the page of results to return.
- *totalRecords*  
The total number of matched users.

**Return Value**

Returns a *MembershipUserCollection* object that contains *MembershipUser* objects.

**Exceptions**

*ArgumentException* - One of the following conditions exists:

- The *emailToMatch* parameter is an empty string or is longer than 256 characters.
- The *pageIndex* parameter is less than 0.
- The *pageSize* parameter is less than 1 or the page upper bound is larger than `Int32.MaxValue`.

ArgumentNullException - The *emailToMatch*, *pageIndex*, *pageSize*, or *totalRecords* parameter is null.

### Remarks

Leading and trailing spaces are trimmed from the *emailToMatch* parameter value. The results returned by the `FindUsersByEmail` method are constrained by the *pageIndex* and *pageSize* parameters. The *pageSize* parameter identifies the maximum number of `MembershipUser` objects to return in the `MembershipUserCollection` object. The *pageIndex* parameter identifies which page of results to return. Zero identifies the first page, as the value is zero-based. The *totalRecords* parameter is an out parameter for the total number of users that matched the *emailToMatch* value.

The `OracleMembershipProvider` class supports extensive searching by accepting the percent character (%) as a wildcard.

### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleMembershipProvider Class](#)
- [OracleMembershipProvider Members](#)

## FindUsersByName

This method returns a collection of users that match the specified user name.

### Declaration

```
// C#  
public override MembershipUserCollection FindUsersByEmail(string userNameToMatch,  
    int pageIndex, int pageSize, out int totalRecords);
```

### Parameters

- *userNameToMatch*  
The user name to search for.
- *pageIndex*  
The zero-based index of the returned results page.
- *pageSize*  
The size of the returned results page.
- *totalRecords*  
The total number of matched users.

### Return Value

Returns a `MembershipUserCollection` object that contains `MembershipUser` objects.

### Exceptions

ArgumentException - One of the following conditions exists:

- The *userNameToMatch* parameter is an empty string, contains a comma, or is longer than 256 characters.
- The *pageIndex* parameter is less than 0.

- The *pageSize* parameter is less than 1 or the page upper bound is larger than `Int32.MaxValue`.

---

**Note:** The page lower bound is  $(pageIndex * pageSize)$  and the page upper bound is  $(pageIndex * pageSize) + (pageSize - 1)$ .

---

ArgumentNullException - The *userNameToMatch*, *pageIndex*, *pageSize*, or *totalRecords* parameter is null.

### Remarks

Leading and trailing spaces are trimmed from the *userNameToMatch* parameter value.

The results returned by the `FindUsersByName` method are constrained by the *pageIndex* and *pageSize* parameters. The *pageSize* parameter identifies the maximum number of `MembershipUser` objects to return in the `MembershipUserCollection` object. The *pageIndex* parameter identifies which page of results to return. Zero identifies the first page, as the value is zero-based. The *totalRecords* parameter is an out parameter for the total number of users that matched the *userNameToMatch* value.

The `OracleMembershipProvider` class supports extensive search by accepting the percent character (%) as a wildcard.

### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleMembershipProvider Class](#)
- [OracleMembershipProvider Members](#)

## GeneratePassword

This method generates a random password that is at least 14 characters in length.

### Declaration

```
// C#  
public virtual string GeneratePassword( );
```

### Return Value

A random string for a password that is at least 14 characters in length.

### Remarks

The `OracleMembershipProvider` object calls the `GeneratePassword` method to get a randomly generated password that is at least 14 characters but less than 128 characters in length.

The generated password contains only alphanumeric characters and the following punctuation marks: `!@#$$%^&*()_+=[{}];<>|./?`. No hidden or non-printable control characters are included in the generated password.

If the value specified for `MinRequiredPasswordLength` property is greater than 14, then the length of the password returned by the `GeneratePassword` property is the value of the `MinRequiredPasswordLength` property. Otherwise, the length is 14 characters.

The random password generated by the `GeneratePassword` method is not guaranteed to pass the regular expression in the `PasswordStrengthRegularExpression` property.



However, the random password meets the criteria established by the `MinRequiredPasswordLength` and `MinRequiredNonAlphanumericCharacters` properties.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleMembershipProvider Class](#)
- [OracleMembershipProvider Members](#)

## GetAllUsers

This method returns a collection of all the users in the database.

**Declaration**

```
// C#  
public override MembershipUserCollection GetAllUsers(int pageIndex, int pageSize,  
    out int totalRecords);
```

**Parameters**

- *pageIndex*  
The zero-based index of the page of results to return.
- *pageSize*  
The size of the page of results to return.
- *totalRecords*  
The total number of users.

**Return Value**

A `MembershipUserCollection` object that contains `MembershipUser` objects.

**Exceptions**

`ArgumentException` - The *pageIndex* parameter is less than 0, or the *pageSize* parameter is less than 1 or the page upper bound is larger than `Int32.MaxValue`.

---

---

**Note:** The page lower bound is  $(pageIndex * pageSize)$  and the page upper bound is  $(pageIndex * pageSize) + (pageSize - 1)$ .

---

---

`ArgumentNullException` - The *pageIndex*, *pageSize*, or *totalRecords* parameter is null.

**Remarks**

The results returned by the `GetAllUsers` method are constrained by the *pageIndex* and *pageSize* parameters. The *pageSize* parameter identifies the maximum number of `MembershipUser` objects to return in the `MembershipUserCollection` object. The *pageIndex* parameter identifies which page of results to return. Zero identifies the first page, as the value is zero-based. The *totalRecords* parameter is an out parameter for the total number of users for the configured applicationName.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleMembershipProvider Class](#)
- [OracleMembershipProvider Members](#)

**GetNumberOfUsersOnline**

This method returns the number of users that are currently accessing the application.

**Declaration**

```
// C#  
public override int GetNumberOfUsersOnline();
```

**Return Value**

An integer value indicating the total number of users currently accessing the application.

**Remarks**

The `GetNumberOfUsersOnline` method returns the number of users of the current application whose last activity date and time is greater than the current date and time less the value (in minutes) of the `Membership.UserIsOnlineTimeWindow` property.

The count includes only users that are associated with the configured `applicationName`.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleMembershipProvider Class](#)
- [OracleMembershipProvider Members](#)

**GetPassword**

This method returns the password for the specified user name from the database.

**Declaration**

```
// C#  
public override string GetPassword(string userName, string passwordAnswer);
```

**Parameters**

- *userName*  
The user to retrieve the password for.
- *passwordAnswer*  
The password answer for the user.

**Return Value**

A password string for the specified user name.

## Exceptions

`ArgumentNullException` - The *userName* parameter is null or the *passwordAnswer* parameter is null when the *RequiresQuestionAndAnswer* property is true.

`System.Web.Security.MembershipPasswordException` - The *passwordAnswer* parameter is invalid or the user identified by *userName* is being locked.

`System.Configuration.Provider.ProviderException` - The *userName* parameter is not found in the membership database, or an error occurred when retrieving the password from the membership database.

`NotSupportedException` - *EnablePasswordRetrieval* property is set to false.

`ArgumentException` - One of the following conditions exists:

- The *userName* parameter is an empty string, contains a comma, or is longer than 256 characters.
- The *passwordAnswer* parameter is an empty string and the *RequiresQuestionAndAnswer* property is true, or the *passwordAnswer* parameter is longer than 128 characters (including the encoded version).

## Remarks

Leading and trailing spaces are trimmed from the *userName* and *passwordAnswer* parameter values.

The `GetPassword` method requires that the `EnablePasswordRetrieval` property be set to true. However, if the `PasswordFormat` property is set to `Hashed`, then a `ProviderException` is thrown when the provider is initialized. In other words, the `GetPassword` method cannot retrieve `Hashed` passwords. By default, the `EnablePasswordRetrieval` property is set to false.

If the `RequiresQuestionAndAnswer` property is set to true and an incorrect password answer is supplied to the `GetPassword` method, then the internal counter that tracks invalid password-answer attempts is incremented by one. This can result in the user being locked out and unable to log on until the lock status is cleared by a call to the `UnlockUser` method. If the correct password answer is supplied and the user is not currently locked out, then the internal counter that tracks invalid password-answer attempts is reset to zero.

### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleMembershipProvider Class](#)
- [OracleMembershipProvider Members](#)

## GetUser

This method returns user information from the database based on the unique identifier for the user.

### Overload List:

- [GetUser\(Object, bool\)](#)

This method returns user information from the database based on the supplied unique identifier.

- [GetUser\(string, bool\)](#)

This method returns user information from the database based on the supplied the user name.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleMembershipProvider Class](#)
- [OracleMembershipProvider Members](#)

## GetUser(Object, bool)

This method returns user information from the database based on the supplied unique identifier.

**Declaration**

```
// C#  
public override MembershipUser GetUser(Object providerUserKey,  
    bool userIsOnline);
```

**Parameters**

- *providerUserKey*

The unique identifier of the user for whom information is being retrieved.

- *userIsOnline*

A Boolean value that indicates whether the method updates the last-activity date/time stamp for the user. If the value is set to `true`, it is updated; otherwise, the method returns user information without updating the last-activity date/time stamp.

**Return Value**

A `MembershipUser` object populated with the specified user's information from the database.

**Exceptions**

`ArgumentException` - The *providerUserKey* parameter is not of type GUID.

`ArgumentNullException` - The *providerUserKey* parameter is null.

**Remarks**

The `GetUser` method provides an option to update the last-activity date/time stamp for the user.

The `GetUser` method returns a `MembershipUser` object populated with information about the specified user. If the user name is not found in the database, then the `GetUser` method returns a null reference.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleMembershipProvider Class](#)
- [OracleMembershipProvider Members](#)

## GetUser(string, bool)

This method returns user information from the database based on the supplied user name.

### Declaration

```
// C#  
public override MembershipUser GetUser(string userName, bool userIsOnline);
```

### Parameters

- *userName*

The name of the user to get information for.

- *userIsOnline*

A Boolean value that indicates whether the method updates the last-activity date/time stamp for the user. If the value is set to `true`, it is updated; otherwise, the method returns user information without updating the last-activity date/time stamp.

### Return Value

A `MembershipUser` object populated with the specified user's information from the database.

### Exceptions

`ArgumentException` - The *userName* parameter is an empty string, contains a comma, or is longer than 256 characters.

`ArgumentNullException` - The *userName* parameter is null.

### Remarks

The `GetUser` method provides an option to update the last-activity date/time stamp for the user.

The `GetUser` method returns a `MembershipUser` object populated with information about the specified user. If the user name is not found in the database, then the `GetUser` method returns a null reference.

#### See Also:

- ["Oracle Providers for ASP.NET Assembly" on page 1-4](#)
- [OracleMembershipProvider Class](#)
- [OracleMembershipProvider Members](#)

## GetUserNameByEmail

This method returns the user name associated with the specified e-mail address.

### Declaration

```
// C#  
public override string GetUserNameByEmail(string emailAddress);
```

### Parameters

- *emailAddress*

The email address to search for.

### Return Value

The user name associated with the specified e-mail address. If no match is found, then it returns a null reference.

### Exceptions

`ArgumentException` - E-mail address exceeds 256 characters.

`System.Configuration.Provider.ProviderException` - More than one user with the same e-mail address exists in the database and the `RequiresUniqueEmail` property is true.

### Remarks

If the value of the `RequiresUniqueEmail` property is true, then a unique e-mail address must be associated with each user name.

#### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleMembershipProvider Class](#)
- [OracleMembershipProvider Members](#)

## Initialize

This method initializes the `OracleMembership` provider with the property values specified in the ASP.NET application configuration file (`web.config`).

### Declaration

```
// C#  
public override void Initialize(string name, NameValueCollection config);
```

### Parameters

- *name*  
The name of the `OracleMembership` provider instance to initialize.
- *config*  
A collection of name/value pairs representing the provider-specific attributes specified in the configuration for this provider.

### Exceptions

`ArgumentNullException` - The *config* parameter is a null value.

`InvalidOperationException` - An attempt is made to call the `Initialize` method on a provider after the provider has already been initialized.

`HttpException` - The current trust level is less than Low.

`System.Configuration.Provider.ProviderException` - One of the following is true in the application configuration file:

- The `enablePasswordRetrieval`, `enablePasswordReset`, `requiresQuestionAndAnswer`, or `requiresUniqueEmail` attribute is set to a value other than a Boolean value.

- The `maxInvalidPasswordAttempts` or `passwordAttemptWindow` attribute is set to a value that is not a positive integer.
- The `minRequiredPasswordLength` attribute is set to a value that is not a positive integer, or the value is greater than 128.
- The `minRequiredNonalphanumericCharacters` attribute is set to a negative integer, or the value is greater than 128.
- The value for the `passwordStrengthRegularExpression` attribute is not a valid regular expression.
- The value for the `applicationName` attribute is greater than 256 characters.
- The value for `passwordFormat` attribute is an invalid `MembershipPasswordFormat` enumeration value.
- The `passwordFormat` attribute is set to `Hashed`, and the `enablePasswordRetrieval` attribute is set to `true`.
- The `passwordFormat` attribute is set to `Encrypted`, and the `machineKey` configuration element specifies `AutoGenerate` for the `decryptionKey` attribute.
- The `connectionStringName` attribute is empty or does not exist in the application configuration file.
- The value of the connection string for the `connectionStringName` attribute value is empty, or the specified `connectionStringName` does not exist in the application configuration file.
- The value for the `commandTimeout` attribute is set to a negative integer.
- The application configuration file for this `OracleMembershipProvider` instance contains an unrecognized attribute.

### Remarks

The `Initialize` method is not intended to be called directly by the application.

#### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleMembershipProvider Class](#)
- [OracleMembershipProvider Members](#)

## ResetPassword

This method resets a user's password and returns a new automatically generated password.

### Declaration

```
// C#
public override string ResetPassword(string userName, string passwordAnswer);
```

### Parameters

- *userName*  
The user to reset the password for.
- *passwordAnswer*  
The password answer for the specified user.

**Return Value**

The new password for the specified user.

**Exceptions**

`ArgumentNullException` - The `userName` parameter is a null reference, or the `passwordAnswer` parameter is null when the `RequiresQuestionAndAnswer` property is true.

`System.Web.Security.MembershipPasswordException` - The `passwordAnswer` parameter is invalid or the user identified by the `userName` parameter is locked out.

`ArgumentException` - One of the following conditions exists:

- The `userName` parameter is an empty string, contains a comma, or is longer than 256 characters.
- The `passwordAnswer` parameter is an empty string and `RequiresQuestionAndAnswer` property is true, or the `passwordAnswer` parameter is longer than 128 characters (including the encoded version).

`System.Configuration.Provider.ProviderException` - One of the following conditions exists:

- `userName` was not found in the membership database.
- The reset-password operation was canceled by a subscriber to the `ValidatingPassword` event and the `FailureInformation` property was a null reference.
- An error occurred when resetting the password in the membership database.

`NotSupportedException` - The `EnablePasswordReset` property is set to false.

`Exception` - An unhandled exception occurred.

**Remarks**

Leading and trailing spaces are trimmed from the `userName` and `passwordAnswer` parameter values.

The `ResetPassword` method is most commonly used when the `PasswordFormat` property is set to `Hashed`. If a user forgets a password that is in hashed format, the password cannot be retrieved. However, the provider can reset the password to a new, automatically generated password if the user supplies the correct password answer. The `ResetPassword` method requires that the `EnablePasswordReset` property is set to true. If an incorrect password answer is supplied to the `ResetPassword` method, then the internal counter that tracks invalid password attempts is incremented by one. This can result in the user being locked out and unable to log on until the lock status is cleared by a call to the `UnlockUser` method. If the correct password answer is supplied and the user is not currently locked out, then the internal counter that tracks invalid password-answer attempts is reset to zero.

The random password generated by the `ResetPassword` method is not guaranteed to pass the regular expression in the `PasswordStrengthRegularExpression` property. However, the random password will meet the criteria established by the `MinRequiredPasswordLength` and `MinRequiredNonAlphanumericCharacters` properties.



**See Also:**

- ["Oracle Providers for ASP.NET Assembly" on page 1-4](#)
- [OracleMembershipProvider Class](#)
- [OracleMembershipProvider Members](#)

## UnlockUser

This method unlocks a user so that the user can be validated.

**Declaration**

```
// C#  
public override bool UnLockUser(string userName);
```

**Parameters**

- *userName*

The name of the user to be unlocked.

**Return Value**

Returns true, if the user was successfully unlocked; false, if otherwise.

**Exceptions**

*ArgumentException* - The *userName* parameter is an empty string, contains a comma, or is longer than 256 characters.

*ArgumentNullException* - The *userName* parameter is null.

**Remarks**

Leading and trailing spaces are trimmed from the *userName* parameter value.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly" on page 1-4](#)
- [OracleMembershipProvider Class](#)
- [OracleMembershipProvider Members](#)

## UpdateUser

This method updates information about a user in the database.

**Declaration**

```
// C#  
public override void UpdateUser(MembershipUser membershipUser);
```

**Parameters**

- *membershipUser*

A *MembershipUser* object populated with user information.

**Exceptions**

*ArgumentException* - One of the following conditions exists:

- The `userName` property of `membershipUser` is an empty string, contains a comma, or is longer than 256 characters.
- The `email` property of `membership User` is an empty string and the `Requires Unique Email` property is set to `true`, or the `mail` property is longer than 256 characters.

Argument Null Exception - The `membership User` parameter is null, or the `surname` or `mail` property of the `membership User` parameter is null.

`System.Configuration.Provider.ProviderException` - One of the following conditions exists:

- The `surname` property of the `membership User` parameter is not found in the membership database.
- The `mail` property of the `membership User` parameter is equal to an existing e-mail address in the membership database, and the `Requires Unique Email` property is set to `true`.
- An error occurred when updating the user in the membership database.

### Remarks

The specified user's `Mail`, `Comment`, `IsApproved`, `Last Login Date`, and `LastActivityDate` property values can be modified, and then updated by the `UpdateUser` method. However, the password for a user cannot. To update the password for a user, use the `ChangePassword` method of the `MembershipUser` class.

#### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleMembershipProvider Class](#)
- [OracleMembershipProvider Members](#)

## ValidateUser

This method validates the user.

### Declaration

```
// C#  
public override bool ValidateUser(string userName, string password);
```

### Parameters

- `userName`  
The name of the user to be validated.
- `password`  
The password for the specified user.

### Return Value

Returns `true` if the specified user name and password are valid; returns `false` if they are not valid or the user does not exist in the database.

**Remarks**

Leading and trailing spaces are trimmed from the *userName* and *password* parameter values.

When a user is successfully validated, the last activity date and last sign-in date values are updated to the current date and time in the database.

The `ValidateUser` method returns `false` on any user who was created with the *isApproved* parameter set to `false`.

If an incorrect password is supplied to the `ValidateUser` method, then the internal counter that tracks invalid password attempts is incremented by one. This can result in the user being locked out and unable to log on until the lock status is cleared by a call to the `UnlockUser` method. If the correct password is supplied and the user is not currently locked out, then the internal counters that track invalid password and password-answer attempts are reset to zero.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleMembershipProvider Class](#)
- [OracleMembershipProvider Members](#)

## OracleMembershipProvider Public Events

OracleMembershipProvider public event is listed in [Table 2–10](#).

**Table 2–10 OracleMembershipProvider Public Events**

Public Event	Description
ValidatingPassword	Inherited from System.Web.Security.MembershipProvider

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleMembershipProvider Class](#)
- [OracleMembershipProvider Members](#)

---

---

## OracleRoleProvider

This chapter describes the OracleRoleProvider class.

**See Also:** ASP.NET role management and role providers  
<http://msdn.microsoft.com/en-us/library/9ab2fxh0.aspx>

This chapter contains the following topic:

- [OracleRoleProvider Class](#)

## OracleRoleProvider Class

The `OracleRoleProvider` class allows ASP.NET applications to store role and user information in an Oracle database.

### Class Inheritance

`System.Object`

`System.Configuration.Provider.ProviderBase`

`System.Web.Security.RoleProvider`

`Oracle.Web.Security`

### Declaration

```
// C#  
public class OracleRoleProvider : RoleProvider
```

### Thread Safety

All public static methods are thread-safe, although instance members are not guaranteed to be thread-safe.

### Remarks

This class allows ASP.NET applications to store and manage role information in an Oracle database.

Note that the role information that this provider manages are application roles and not database roles.

### Example

The following is a `web.config` example for an ASP.NET application that uses the `OracleRoleProvider` class as the default provider. This configuration uses the connection string and default attribute values specified in the `machine.config` file:

```
<?xml version="1.0"?>  
<configuration xmlns="http://schemas.microsoft.com/.NetConfiguration/v2.0">  
  <system.web>  
    <roleManager enabled="true" defaultProvider="OracleRoleProvider"/>  
  </system.web>  
</configuration>
```

The following is a `web.config` example for an ASP.NET application that uses an `OracleRoleProvider` as the default provider with customized settings and an application-specific connection string.

```
<?xml version="1.0"?>  
<configuration xmlns=  
  "http://schemas.microsoft.com/.NetConfiguration/v2.0">  
  <connectionStrings>  
    <add name="my_role_app_con_string" connectionString=  
      "User Id=scott;Password=tiger;Data Source=Oracle"/>  
  </connectionStrings>  
  <system.web>  
    <!-- Enable and customize OracleRoleProvider -->  
    <roleManager enabled="true" defaultProvider="MyOracleRoleProvider">  
      <providers>
```

```
<add name="MyOracleRoleProvider"
      type="Oracle.Web.Security.OracleRoleProvider,
           Oracle.Web, Version=2.112.2.0, Culture=neutral,
           PublicKeyToken=89b483f429c47342"
      connectionStringName="my_role_app_con_string"
      applicationName="my_role_app" />
</providers>
</roleManager>
</system.web>
</configuration>
```

Note that the `applicationName` attribute should be set to a unique value for each ASP.NET application.

### Requirements

Namespace: `Oracle.Web.Security`

Assembly: `Oracle.Web.dll`

Oracle Providers for ASP.NET Version: Oracle Providers for ASP.NET 2.0 and Oracle Providers for ASP.NET 4

### See Also:

- ["Oracle Providers for ASP.NET Assembly" on page 1-4](#)
- [OracleRoleProvider Members](#)
- [OracleRoleProvider Constructors](#)
- [OracleRoleProvider Static Methods](#)
- [OracleRoleProvider Public Properties](#)
- [OracleRoleProvider Public Methods](#)

## OracleRoleProvider Members

OracleRoleProvider members are listed in the following tables.

### OracleRoleProvider Constructors

The OracleRoleProvider constructor is listed in [Table 3-1](#).

**Table 3-1 OracleRoleProvider Constructor**

Constructor	Description
<a href="#">OracleRoleProvider Constructors</a>	Instantiates a new instance of the OracleRoleProvider class

### OracleRoleProvider Static Methods

OracleRoleProvider static methods are listed in [Table 3-2](#).

**Table 3-2 OracleRoleProvider Static Methods**

Static Methods	Description
Equals	Inherited from System.Object (Overloaded)
ReferenceEquals	Inherited from System.Object

### OracleRoleProvider Public Properties

OracleRoleProvider public properties are listed in [Table 3-3](#).

**Table 3-3 OracleRoleProvider Public Properties**

Public Properties	Description
<a href="#">ApplicationName</a>	Gets or sets the name of the application that stores the role provider information
<a href="#">CommandTimeout</a>	Gets the number of seconds that the command is allowed to execute before it is terminated with an exception
Description	Inherited from System.Configuration.Provider.Providerbase
Name	Inherited from System.Configuration.Provider.Providerbase

### OracleRoleProvider Public Methods

OracleRoleProvider public methods are listed in [Table 3-4](#).

**Table 3-4 OracleRoleProvider Public Methods**

Public Methods	Description
<a href="#">AddUsersToRoles</a>	Adds the specified users to the specified roles
<a href="#">CreateRole</a>	Adds a new role to the database
<a href="#">DeleteRole</a>	Deletes a role in the database
Equals	Inherited from System.Object (Overloaded)
<a href="#">FindUsersInRole</a>	Returns an array of user names that match the specified user name
<a href="#">GetAllRoles</a>	Returns an array of all the roles for an application
GetHashCode	Inherited from System.Object



**Table 3–4 (Cont.) OracleRoleProvider Public Methods**

<b>Public Methods</b>	<b>Description</b>
<a href="#">GetRolesForUser</a>	Returns an array of role names for the specified user
<a href="#">GetType</a>	Inherited from <code>System.Object</code>
<a href="#">GetUsersInRole</a>	Returns an array of users in the specified role name
<a href="#">Initialize</a>	Initializes <code>OracleRoleProvider</code> with the property values specified in the ASP.NET application configuration file
<a href="#">IsUserInRole</a>	Indicates whether the specified user is in the specified role
<a href="#">RemoveUsersFromRoles</a>	Removes the specified array of users from the specified array of role names
<a href="#">RoleExists</a>	Indicates whether the specified role name exists in the database
<a href="#">ToString</a>	Inherited from <code>System.Object</code>

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleRoleProvider Class](#)

## OracleRoleProvider Constructors

OracleRoleProvider constructors create instances of the OracleRoleProvider class.

### Overload List:

- [OracleRoleProvider\(\)](#)

This constructor creates an instance of the OracleRoleProvider class.

#### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleRoleProvider Class](#)
- [OracleRoleProvider Members](#)

### OracleRoleProvider()

This constructor creates an instance of the OracleRoleProvider class.

#### Declaration

```
// C#  
public OracleRoleProvider();
```

#### Remarks

This constructor creates a new instance of the OracleRoleProvider class.

#### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleRoleProvider Class](#)
- [OracleRoleProvider Members](#)

## OracleRoleProvider Static Methods

The OracleRoleProvider static methods are listed in [Table 3-5](#).

**Table 3-5 OracleRoleProvider Static Methods**

Static Methods	Description
Equals (Overloaded)	Inherited from System.Object
ReferenceEquals	Inherited from System.Object

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleRoleProvider Class](#)
- [OracleRoleProvider Members](#)

## OracleRoleProvider Public Properties

The OracleRoleProvider public properties are listed in [Table 3–6](#).

**Table 3–6 OracleRoleProvider Public Properties**

Public Properties	Description
<a href="#">ApplicationName</a>	Gets or sets the name of the application that stores the role provider information
<a href="#">CommandTimeout</a>	Gets the number of seconds that the command is allowed to execute before it is terminated with an exception
Description	Inherited from System.Configuration.Provider.Providerbase
Name	Inherited from System.Configuration.Provider.Providerbase

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleRoleProvider Class](#)
- [OracleRoleProvider Members](#)

### ApplicationName

This property gets or sets the name of the application that stores the role provider information.

**Declaration**

```
// C#
public override string ApplicationName{get; set;};
```

**Property Value**

The name of the application. If the applicationName attribute is not specified in the application configuration file, or if the value is an empty string, then this property is set to the application virtual path.

**Exceptions**

HttpException - The user setting the ApplicationName property does not have high ASP.NET hosting permission.

System.Configuration.Provider.ProviderException - The ApplicationName property is set to a string greater than 256 characters.

**Remarks**

The string value of the ApplicationName property is used to associate user names and role names with different applications. Multiple applications can use the same database to store user names and role names without running into any conflict between duplicate names. This property can be set programmatically, or it can be set declaratively in the Web application configuration file using the applicationName attribute. The attribute name in the configuration file is case-sensitive.

The ApplicationName property is not thread-safe. It is recommended that the programming code not allow users to set the ApplicationName property in Web applications.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly" on page 1-4](#)
- [OracleRoleProvider Class](#)
- [OracleRoleProvider Members](#)

**CommandTimeout**

This property gets the number of seconds that the command is allowed to execute before it is terminated with an exception.

**Declaration**

```
// C#  
public int CommandTimeout {get;}
```

**Property Value**

An int.

**Remarks**

To customize a provider, ASP.NET developers can set an integer value for this property through the `web.config` file using the `commandTimeout` attribute.

The default value is 30 seconds. The attribute name in the configuration file is case-sensitive.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly" on page 1-4](#)
- [OracleRoleProvider Class](#)
- [OracleRoleProvider Members](#)

## OracleRoleProvider Public Methods

The `OracleRoleProvider` public methods are listed in [Table 3-7](#).

**Table 3-7 OracleRoleProvider Public Methods**

Public Method	Description
<a href="#">AddUsersToRoles</a>	Adds the specified users to the specified roles
<a href="#">CreateRole</a>	Adds a new role to the database
<a href="#">DeleteRole</a>	Deletes a role in the database
<code>Equals</code>	Inherited from <code>System.Object</code> (Overloaded)
<a href="#">FindUsersInRole</a>	Returns an array of user names that match the specified user name
<a href="#">GetAllRoles</a>	Returns an array of all the roles for an application
<code>Get Hash Code</code>	Inherited from <code>System.Object</code>
<a href="#">GetRolesForUser</a>	Returns an array of role names for the specified user
<code>Getup</code>	Inherited from <code>System.Object</code>
<a href="#">GetUsersInRole</a>	Returns an array of users in the specified role name
<a href="#">Initialize</a>	Initializes <code>OracleRoleProvider</code> with the property values specified in the ASP.NET application configuration file
<a href="#">IsUserInRole</a>	Indicates whether the specified user is in the specified role
<a href="#">RemoveUsersFromRoles</a>	Removes the specified array of users from the specified array of role names
<a href="#">RoleExists</a>	Indicates whether the specified role name exists in the database
<code>ToString</code>	Inherited from <code>System.Object</code>

### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleRoleProvider Class](#)
- [OracleRoleProvider Members](#)

### AddUsersToRoles

This method adds the specified users to the specified roles.

#### Declaration

```
// C#
public override void AddUsersToRoles(string[] userNames, string[] roleNames);
```

#### Parameters

- `userNames`  
An array of user names to be added to roles.
- `roleNames`  
An array of role names to add the user names to.

**Exceptions**

`ArgumentNullException` - One of the users in *userNames* or one of the roles in *roleNames* is null.

`ArgumentException` - Either the *roleNames* or *userNames* parameter is an empty string, contains a comma, is longer than 256 characters, or contains a duplicate element.

`System.Configuration.Provider.ProviderException` - One or more role names were not found, or one or more user names are already associated with one or more role names.

`OracleException` - An Oracle-related error has occurred.

**Remarks**

This method adds one or more user names to one or more of the specified roles. The updates are performed in a transaction. If an error occurs, then the transaction is rolled back and no updates are made.

If one of the user names does not exist in the database, then the user name will be created and added to the database.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleRoleProvider Class](#)
- [OracleRoleProvider Members](#)

**CreateRole**

This method adds a new role to the database.

**Declaration**

```
// C#
public override void CreateRole(string roleName);
```

**Parameters**

- *roleName*

The role name to be created in the database.

**Exceptions**

`ArgumentNullException` - The *roleName* parameter is null.

`ArgumentException` - The *roleName* parameter is an empty string, contains a comma, or is longer than 256 characters.

`System.Configuration.Provider.ProviderException` - The role name already exists in the database.

`OracleException` - An Oracle-related error has occurred.

**Remarks**

This method creates a new role in the database.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly" on page 1-4](#)
- [OracleRoleProvider Class](#)
- [OracleRoleProvider Members](#)

## DeleteRole

This method deletes a role in the database.

**Declaration**

```
// C#  
public override bool DeleteRole(string roleName, bool throwOnPopulatedRole);
```

**Parameters**

- *roleName*  
The role name to be deleted from the database.
- *throwOnPopulatedRole*  
A Boolean value that, if set to `true`, causes an exception if the role contains any user names. If the value is set to `false`, deletes the role from the database.

**Return Value**

Returns `true` if the specified role was successfully deleted; otherwise, returns `false`.

**Exceptions**

`ArgumentNullException` - The *roleName* parameter is null.

`System.Configuration.Provider.ProviderException` - The role name contains at least one user name and the *throwOnPopulatedRole* parameter is set to `true`.

`OracleException` - An Oracle-related error has occurred.

`ArgumentException` - The *roleName* parameter is an empty string, contains a comma, or is longer than 256 characters:

**Remarks**

If the *throwOnPopulatedRole* parameter is set to `false`, then it deletes the specified role from the database regardless of whether it contains any users. If the *throwOnPopulatedRole* parameter is set to `true`, then an exception is thrown if the specified role in the database contains any users, but the role is not deleted.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly" on page 1-4](#)
- [OracleRoleProvider Class](#)
- [OracleRoleProvider Members](#)

## FindUsersInRole

This method returns an array of user names that match the specified user name, for the specified role name.



## Declaration

```
// C#  
public override string[ ] FindUsersInRole(string roleName, string  
    userNameToMatch);
```

## Parameters

- *roleName*  
The role name being searched for in the database.
- *userNameToMatch*  
The user name being searched for.

## Return Value

A string array that contains user names in the specified role that match the specified *userNameToMatch* parameter.

## Exceptions

*ArgumentNullException* - The *roleName* or *userNameToMatch* parameter is null.

*OracleException* - An Oracle-related error has occurred.

*System.Configuration.Provider.ProviderException* - The role name does not exist in the database.

*ArgumentException* - One of the following conditions exists:

- The *roleName* parameter is an empty string, contains a comma, or is longer than 256 characters.
- The *userNameToMatch* parameter is an empty string or is longer than 256 characters.

## Remarks

This method returns an array of user names that match the specified user name, for the specified role name. This method supports Oracle wildcard characters. If the *userNameToMatch* parameter is set to *"oraUser%"*, then an array is returned for users such as *"oraUser1"*, *"oraUser2"*, and so on. However, if the *userNameToMatch* parameter is set to *"oraUser"*, then the array is returned with just the username *"oraUser"*, if there is an *"oraUser"*.

### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleRoleProvider Class](#)
- [OracleRoleProvider Members](#)

## GetAllRoles

This method returns an array of all the roles for an application.

## Declaration

```
// C#  
public override string[ ] GetAllRoles();
```

**Return Value**

A string array containing all the role names in a database for the application.

**Exceptions**

*OracleException* - An Oracle related error has occurred.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleRoleProvider Class](#)
- [OracleRoleProvider Members](#)

**GetRolesForUser**

This method returns an array of role names for the specified user.

**Declaration**

```
// C#  
public override string[] GetRolesForUser(string userName);
```

**Parameters**

- *userName*

The user name for which an array of role names is returned.

**Return Value**

An array of role names for the specified user name.

**Exceptions**

*ArgumentNullException* - The *userName* parameter is null.

*ArgumentException* - The *userName* parameter contains a comma or is longer than 256 characters.

*OracleException* - An Oracle-related error has occurred.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleRoleProvider Class](#)
- [OracleRoleProvider Members](#)

**GetUsersInRole**

This method returns an array of users in the specified role name.

**Declaration**

```
// C#  
public override string[] GetUsersInRole(string roleName);
```

**Parameters**

- *roleName*

The role name for which an array of users is returned.

### Return Value

An array of user names in the specified role name.

### Exceptions

`ArgumentNullException` - The *roleName* parameter is null.

`OracleException` - An Oracle-related error has occurred.

`System.Configuration.Provider.ProviderException` - The role name does not exist in the database.

`ArgumentException` - The *roleName* parameter is an empty string, contains a comma, or is longer than 256 characters:

#### See Also:

- ["Oracle Providers for ASP.NET Assembly" on page 1-4](#)
- [OracleRoleProvider Class](#)
- [OracleRoleProvider Members](#)

## Initialize

This method initializes the `OracleRoleProvider` instance with the property values specified in the ASP.NET application configuration file (`web.config`).

### Declaration

```
// C#  
public override void Initialize(string name, NameValueCollection config);
```

### Parameters

- *name*

The name of the `OracleRoleProvider` instance to initialize.

- *config*

A `Systems.Collections.Specialized.NameValueCollection` object that contains the names and values of configuration options for the role provider.

### Exceptions

`System.Web.HttpException` - ASP.NET is not running at medium trust or higher.

`ArgumentNullException` - The *config* parameter is null.

`System.Configuration.Provider.ProviderException` - The `connectionStringName` attribute is empty or does not exist in the configuration file, the `applicationName` attribute exceeds 256 characters, or the configuration file contains an invalid attribute.

### Remarks

The `Initialize` method is not intended to be called directly by the application.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly" on page 1-4](#)
- [OracleRoleProvider Class](#)
- [OracleRoleProvider Members](#)

**IsUserInRole**

This method indicates whether the specified user is in the specified role.

**Declaration**

```
// C#  
public override bool IsUserInRole(string userName, string roleName);
```

**Parameters**

- *userName*  
The user name being searched for.
- *roleName*  
The role name being searched in.

**Return Value**

Returns `true` if the specified user name is in the specified role name; otherwise, returns `false`.

**Exceptions**

`ArgumentNullException` - The *userName* or *roleName* parameter is null.

`OracleException` - An Oracle-related error has occurred.

`ArgumentException` - One of the following conditions exists:

- The *roleName* parameter is an empty string, contains a comma, or is longer than 256 characters.
- The *userName* parameter contains a comma or is longer than 256 characters.

**Remarks**

This method determines if the specified user name exists in the specified role name in the database.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly" on page 1-4](#)
- [OracleRoleProvider Class](#)
- [OracleRoleProvider Members](#)

**RemoveUsersFromRoles**

This method removes the specified array of users from the specified array of role names.

**Declaration**

```
// C#
```

```
public override void RemoveUsersFromRoles(string[] userNames, string[] roleNames);
```

### Parameters

- *userNames*  
An array of user names to remove from the role names.
- *roleNames*  
An array of role names to remove the user names from.

### Exceptions

*ArgumentNullException* - One of the users in the *userNames* parameter or one of the roles in the *roleNames* parameter is null.

*OracleException* - An Oracle-related error has occurred.

*System.Configuration.Provider.ProviderException* - One or more of the role names or user names does not exist in the database, or one or more of the user names is not associated a role name.

*ArgumentException* - The *roleNames* or *userNames* parameter is an empty string, contains a comma, is longer than 256 characters, or contains a duplicate element.

### Remarks

This method removes the specified array of user names from the specified array of role names. The updates are made within a transaction. If an error occurs, the transaction is rolled back.

#### See Also:

- ["Oracle Providers for ASP.NET Assembly" on page 1-4](#)
- [OracleRoleProvider Class](#)
- [OracleRoleProvider Members](#)

## RoleExists

This method indicates whether the specified role name exists in the database.

### Declaration

```
// C#  
public override bool RoleExists(string roleName);
```

### Parameters

- *roleName*  
The role name being searched for in the database.

### Return Value

Returns true if the role name exists; otherwise, returns false.

### Exceptions

*ArgumentNullException* - The *roleName* parameter is null

*OracleException* - An Oracle-related error has occurred.

`ArgumentException` - The *roleName* parameter is an empty string, contains a comma, or is longer than 256 characters.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleRoleProvider Class](#)
- [OracleRoleProvider Members](#)

---

---

# OracleSiteMapProvider

This chapter describes the `OracleSiteMapProvider` class.

**See Also:** ASP.NET site navigation and site map provider  
<http://msdn.microsoft.com/en-us/library/ms227558.aspx>

This chapter contains the following topic:

- [OracleSiteMapProvider Class](#)

---

## OracleSiteMapProvider Class

This class allows ASP.NET applications to retrieve site map information from an Oracle database.

### Class Inheritance

System.Object

System.Configuration.Provider.ProviderBase

System.Web.SiteMapProvider

System.Web.StaticSiteMapProvider

Oracle.Web.SiteMap.OracleSiteMapProvider

### Declaration

```
// C#
Public class OracleSiteMapProvider: StaticSiteMapProvider, IDisposable
```

### Thread Safety

All public static methods are thread-safe, although instance members are not guaranteed to be thread-safe.

### Remarks

This class allows ASP.NET applications to read and load site map information from an Oracle database.

### Examples

The following is a web.config example for an ASP.NET application that uses OracleSiteMapProvider as the default provider. This configuration uses the connection string and default attribute values specified in the machine.config file.

```
<?xml version="1.0"?>
<configuration xmlns="http://schemas.microsoft.com/.NetConfiguration/v2.0">
  <system.web>
    <siteMap defaultProvider="OracleSiteMapProvider"/>
  </system.web>
</configuration>
```

The following is a web.config example for an ASP.NET application that uses OracleSiteMapProvider as the default provider, with customized settings and an application-specific connection string:

```
<?xml version="1.0"?>
<configuration xmlns=
  "http://schemas.microsoft.com/.NetConfiguration/v2.0">
  <connectionStrings>
    <add name="my_sitemap_app_con_string" connectionString=
      "User Id=scott;Password=tiger;Data Source=Oracle"/>
  </connectionStrings>
  <system.web>
    <!-- Enable and customize OracleSiteMapProvider -->
    <siteMap defaultProvider="CustomOracleSiteMapProvider">
      <providers>
        <add name="CustomOracleSiteMapProvider"
```



```
        type="Oracle.Web.SiteMap.OracleSiteMapProvider,  
            Oracle.Web, Version=2.112.2.0, Culture=neutral,  
            PublicKeyToken=89b483f429c47342"  
        connectionStringName="my_sitemap_app_con_string"  
        applicationName="my_sitemap_app"  
        securityTrimmingEnabled="false"/>  
    </providers>  
</sitemap>  
</system.web>  
</configuration>
```

Note that the `applicationName` attribute should be set to a unique value for each ASP.NET application.

### Requirements

Namespace: `Oracle.Web.SiteMap`

Assembly: `Oracle.Web.dll`

Oracle Providers for ASP.NET Version: Oracle Providers for ASP.NET 2.0 and Oracle Providers for ASP.NET 4

`OracleSiteMapProvider` requires the Change Notification privilege with Oracle Database 10g release 2 (10.2) and later.

### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleSiteMapProvider Members](#)
- [OracleSiteMapProvider Constructors](#)
- [OracleSiteMapProvider Public Properties](#)
- [OracleSiteMapProvider Public Methods](#)

## OracleSiteMapProvider Members

OracleSiteMapProvider members are listed in the following tables.

### OracleSiteMapProvider Constructors

The OracleSiteMapProvider constructor is listed in [Table 4-1](#).

**Table 4-1 OracleSiteMapProvider Constructor**

Constructor	Description
<a href="#">OracleSiteMapProvider Constructors</a>	Instantiates a new instance of the OracleSiteMapProvider class

### OracleSiteMapProvider Static Methods

OracleSiteMapProvider static methods are listed in [Table 4-2](#).

**Table 4-2 OracleSiteMapProvider Static Methods**

Static Methods	Description
Equals	Inherited from System.Object (Overloaded)
ReferenceEquals	Inherited from System.Object

### OracleSiteMapProvider Public Properties

OracleSiteMapProvider public properties are listed in [Table 4-3](#).

**Table 4-3 OracleSiteMapProvider Public Properties**

Public Properties	Description
<a href="#">ApplicationName</a>	Gets or sets the name of the application that differentiates site map data for different applications
<a href="#">CommandTimeout</a>	Gets the number of seconds that the command is allowed to execute before it is terminated with an exception
CurrentNode	Inherited from System.Web.SiteMapProvider (Read-Only)
Description	Inherited from System.Configuration.Provider.Providerbase
EnableLocalization	Inherited from System.Web.SiteMapProvider
Name	Inherited from System.Configuration.Provider.Providerbase
ParentProvider	Inherited from System.Web.SiteMapProvider
ResourceKey	Inherited from System.Web.SiteMapProvider
RootNode	Inherited from System.Web.SiteMapProvider (Read-Only)
RootProvider	Inherited from System.Web.SiteMapProvider
SecurityTrimmingEnabled	Inherited from System.Web.SiteMapProvider

### OracleSiteMapProvider Public Methods

OracleSiteMapProvider public methods are listed in [Table 4-4](#).

**Table 4–4 OracleSiteMapProvider Public Methods**

Public Methods	Description
<a href="#">BuildSiteMap</a>	Builds a SiteMap tree of the SiteMapNode objects by loading site map data from an Oracle database
<a href="#">Dispose</a>	Releases all the resources for this instance
FindSiteMapNode	Inherited from System.Web.StaticSiteMapProvider
FindSiteMapNodeFromKey	Inherited from System.Web.StaticSiteMapProvider
GetChildNodes	Inherited from System.Web.StaticSiteMapProvider
GetCurrentNodeAndHintAncestorNodes	Inherited from System.Web.SiteMapProvider
GetCurrentNodeAndHintNeighborhoodNodes	Inherited from System.Web.SiteMapProvider
GetParentNode	Inherited from System.Web.StaticSiteMapProvider
GetParentNodeRelativeToCurrentNodeAndHintDownFromParent	Inherited from System.Web.SiteMapProvider
GetParentNodeRelativeToNodeAndHintDownFromParent	Inherited from System.Web.SiteMapProvider
HintAncestorNodes	Inherited from System.Web.SiteMapProvider
IsAccessibleToUser	Inherited from System.Web.SiteMapProvider
GetHashCode	Inherited from System.Object
GetType	Inherited from System.Object
<a href="#">Initialize</a>	Initializes the OracleSiteMapProvider instance with the property values specified in the ASP.NET application configuration file
ToString	Inherited from System.Object

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleSiteMapProvider Class](#)

## OracleSiteMapProvider Constructors

This constructor instantiates a new instance of the `OracleSiteMapProvider` class.

### Overload List:

- [OracleSiteMapProvider\(\)](#)

This constructor creates an instance of the `OracleSiteMapProvider` class.

#### See Also:

- [OracleSiteMapProvider Class](#)
- [OracleSiteMapProvider Members](#)

### OracleSiteMapProvider()

This constructor instantiates a new instance of the `OracleSiteMapProvider` class.

#### Declaration

```
// C#  
public OracleSiteMapProvider();
```

#### Remarks

The `OracleSiteMapProvider` constructor is called by ASP.NET to create an instance of the `OracleSiteMapProvider` class as specified in the configuration file for the application. Initialization values for the `OracleSiteMapProvider` object are passed through the `Initialize` method.

This constructor is not intended to be used directly by the application.

#### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleSiteMapProvider Class](#)
- [OracleSiteMapProvider Members](#)

## OracleSiteMapProvider Static Methods

OracleSiteMapProvider static methods are listed in [Table 4-5](#).

**Table 4-5 OracleSiteMapProvider Static Methods**

Static Methods	Description
Equals	Inherited from System.Object (Overloaded)
ReferenceEquals	Inherited from System.Object

## OracleSiteMapProvider Public Properties

OracleSiteMapProvider public properties are listed in [Table 4-6](#).

**Table 4-6 OracleSiteMapProvider Public Properties**

Public Properties	Description
<a href="#">ApplicationName</a>	Gets or sets the name of the application that differentiates site map data for different applications
<a href="#">CommandTimeout</a>	Gets the number of seconds that the command is allowed to execute before it is terminated with an exception
CurrentNode	Inherited from <code>System.Web.SiteMapProvider</code>
Description	Inherited from <code>System.Configuration.Provider.Providerbase</code>
EnableLocalization	Inherited from <code>System.Web.SiteMapProvider</code>
Name	Inherited from <code>System.Configuration.Provider.Providerbase</code>
ParentProvider	Inherited from <code>System.Web.SiteMapProvider</code>
ResourceKey	Inherited from <code>System.Web.SiteMapProvider</code>
RootNode	Inherited from <code>System.Web.SiteMapProvider</code>
RootProvider	Inherited from <code>System.Web.SiteMapProvider</code>
SecurityTrimmingEnabled	Inherited from <code>System.Web.SiteMapProvider</code>

### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleSiteMapProvider Class](#)
- [OracleSiteMapProvider Members](#)

### ApplicationName

This property gets or sets the name of the application that differentiates site map data for different applications.

#### Declaration

```
// C#
public string ApplicationName{get; set;}
```

#### Property Value

The name of the application. If the `applicationName` attribute is not specified in the application configuration file, or if the value is an empty string, then this property is set to the application virtual path.

#### Exceptions

`HttpException` - The `ApplicationName` property was set by a user that does not have high ASP.NET hosting permission.

`System.Configuration.Provider.ProviderException` - The application name supplied exceeds 256 characters.

`ArgumentException` - The application name supplied is an empty string or a null reference.

### Remarks

The string value of the `ApplicationName` property is used for organizing user information.

Multiple ASP.NET applications can use the same data source and create duplicate user names because user information is stored uniquely for each application name. This property can be set programmatically, or it can be set declaratively in the configuration file for the Web application using the `applicationName` attribute. The attribute name in the configuration file is case-sensitive.

The `ApplicationName` property is not thread-safe. It is recommended that program code not allow users to set the `ApplicationName` property in Web applications.

### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleSiteMapProvider Class](#)
- [OracleSiteMapProvider Members](#)

## CommandTimeout

This property gets the number of seconds that the command is allowed to execute before it is terminated with an exception.

### Declaration

```
// C#  
public int CommandTimeout {get;}
```

### Property Value

An `int`.

### Remarks

To customize a provider, ASP.NET developers can set an integer value for this property through the `web.config` file using the `commandTimeout` attribute.

The default value is 30 seconds. The attribute name in the configuration file is case-sensitive.

### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleSiteMapProvider Class](#)
- [OracleSiteMapProvider Members](#)

## OracleSiteMapProvider Public Methods

OracleSiteMapProvider public methods are listed in [Table 4-7](#).

**Table 4-7 OracleSiteMapProvider Public Methods**

Public Methods	Description
<a href="#">BuildSiteMap</a>	Builds a SiteMap tree of the SiteMapNode objects by loading site map data from an Oracle database
<a href="#">Dispose</a>	Releases all the resources for this instance
FindSiteMapNode	Inherited from System.Web.StaticSiteMapProvider
FindSiteMapNodeFromKey	Inherited from System.Web.StaticSiteMapProvider
GetChildNodes	Inherited from System.Web.StaticSiteMapProvider
GetCurrentNodeAndHintAncestorNodes	Inherited from System.Web.SiteMapProvider
GetCurrentNodeAndHintNeighborhoodNodes	Inherited from System.Web.SiteMapProvider
GetParentNode	Inherited from System.Web.StaticSiteMapProvider
GetParentNodeRelativeToCurrentNodeAndHintDownFromParent	Inherited from System.Web.SiteMapProvider
GetParentNodeRelativeToNodeAndHintDownFromParent	Inherited from System.Web.SiteMapProvider
HintAncestorNodes	Inherited from System.Web.SiteMapProvider
IsAccessibleToUser	Inherited from System.Web.SiteMapProvider
GetHashCode	Inherited from System.Object
GetType	Inherited from System.Object
<a href="#">Initialize</a>	Initializes the OracleSiteMapProvider instance with the property values specified in the ASP.NET application's configuration file
ToString	Inherited from System.Object

### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleSiteMapProvider Class](#)
- [OracleSiteMapProvider Members](#)

### BuildSiteMap

This method builds a SiteMap tree of the SiteMapNode objects by loading site map data from the Oracle database.

#### Declaration

```
// C#
Public override SiteMapNode BuildSiteMap();
```

#### Return Value

The root SiteMapNode of the site map navigation structure.



**Exceptions**

`InvalidOperationException` - The `OracleSiteMapProvider` instance is not initialized.

`ProviderException` - One of the following conditions exists:

- Root node is not found.
- Multiple root nodes are found.
- Parent node is not found for any node.

`ConfigurationErrorsException` - One of the following conditions exists:

- The roles of the `SiteMapNode` contain characters that are not valid.
- A URL is parsed for a `SiteMapNode` that is not unique.
- A `SiteMapNode` object was encountered with a duplicate value for `Key`.
- An error occurred while parsing the URL of a `SiteMapNode`.

**Remarks**

This method fetches site map data from the database and builds a tree of site map nodes in memory. The `OracleSiteMapProvider` object could choose to subscribe to database change notifications to get notified about the changes in the site map data in the database. This method is thread-safe.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleSiteMapProvider Class](#)
- [OracleSiteMapProvider Members](#)
- *Oracle Data Provider for .NET Developer's Guide* for more information about change notification

**Dispose**

This method releases all the resources for this instance.

**Declaration**

```
// C#
public override void Dispose();
```

**Remarks**

This method releases all the resources for this instance when the application domain is closed.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleSiteMapProvider Class](#)
- [OracleSiteMapProvider Members](#)

**Initialize**

This method initializes the `OracleSiteMapProvider` instance with the property values specified in the ASP.NET application configuration file (`web.config`).

### Declaration

```
// C#  
Public override void Initialize(string name, NameValueCollection config);
```

### Parameters

- *name*  
The name of the OracleSiteMapProvider instance to initialize.
- *config*  
A `Systems.Collections.Specialized.NameValueCollection` object that contains the names and values of configuration options for the site map provider.

### Exceptions

`ArgumentNullException` - The *config* parameter is null.

`InvalidOperationException` - A SiteMapProvider is already initialized.

`ProviderException` - One of the following exists:

- The `connectionStringName` attribute is null or empty in the configuration file.
- The connection string corresponding to the value of the `connectionStringName` attribute is null or empty.
- The configuration file contains an unrecognized attribute.
- An error occurred during initialization of the provider.

### Remarks

The `Initialize` method is not intended to be called directly by the application.

#### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleSiteMapProvider Class](#)
- [OracleSiteMapProvider Members](#)

---

---

## OracleSessionStateStore

This chapter describes the `OracleSessionStateStore` class.

**See Also:** ASP.NET session state and session state providers  
<http://msdn.microsoft.com/en-us/library/ms178581.aspx>

This chapter contains the following topic:

- [OracleSessionStateStore Class](#)

---

## OracleSessionStateStore Class

The `OracleSessionStateStore` class allows ASP.NET applications to store session information in an Oracle database.

### Class Inheritance

`System.Object`

`System.Configuration.Provider.ProviderBase`

`System.Web.SessionState.SessionStateStoreProviderBase`

`Oracle.Web.SessionState`

### Declaration

```
// C#
public class OracleSessionStateStore : SessionStateStoreProviderBase
```

### Thread Safety

All public static methods are thread-safe, although instance members are not guaranteed to be thread-safe.

### Remarks

This class allows ASP.NET applications to store and manage session state information in an Oracle database.

Note that the session information that this provider manages is application session information, not database session information.

Expired session data is periodically deleted from the database.

### Example

The following is a `web.config` example for an ASP.NET application that uses `OracleSessionStateStore` as the default provider with customized settings and an application-specific connection string:

```
<?xml version="1.0"?>
<configuration xmlns=
  "http://schemas.microsoft.com/.NetConfiguration/v2.0">
  <connectionStrings>
    <add name="my_sessionstate_app_con_string" connectionString=
      "User Id=scott;Password=tiger;Data Source=Oracle"/>
  </connectionStrings>
  <system.web>
    <!-- Enable and customize OracleSessionStateProvider -->
    <sessionState mode="Custom" customProvider="MyOracleSessionStateStore">
      <providers>
        <add name="MyOracleSessionStateStore"
          type="Oracle.Web.SessionState.OracleSessionStateStore,
            Oracle.Web, Version=2.112.2.0, Culture=neutral,
            PublicKeyToken=89b483f429c47342"
          connectionStringName="my_sessionstate_app_con_string"/>
      </providers>
    </sessionState>
  </system.web>
</configuration>
```

**Requirements**

Namespace: Oracle.Web.SessionState

Assembly: Oracle.Web.dll

Oracle Providers for ASP.NET Version: Oracle Providers for ASP.NET 2.0 and Oracle Providers for ASP.NET 4

**See Also:**

- ["Oracle Providers for ASP.NET Assembly" on page 1-4](#)
- [OracleSessionStateStore Members](#)
- [OracleSessionStateStore Constructors](#)
- [OracleSessionStateStore Public Methods](#)
- [OracleSessionStateStore Public Properties](#)

## OracleSessionStateStore Members

OracleSessionStateStore members are listed in the following tables.

### OracleSessionStateStore Constructors

The OracleSessionStateStore constructor is listed in [Table 5-1](#).

**Table 5-1 OracleSessionStateStore Constructor**

Constructor	Description
<a href="#">OracleSessionStateStore Constructors</a>	Instantiates a new instance of the OracleSessionStateStore class

### OracleSessionStateStore Public Properties

OracleSessionStateStore public properties are listed in [Table 5-2](#).

**Table 5-2 OracleSessionStateStore Public Properties**

Public Properties	Description
<a href="#">CommandTimeout</a>	Gets the number of seconds that the command is allowed to execute before it is terminated with an exception
Description	Inherited from System.Configuration.Provider.Providerbase
Name	Inherited from System.Configuration.Provider.Providerbase

### OracleSessionStateStore Public Methods

The OracleSessionStateStore public methods are listed in [Table 5-3](#).

**Table 5-3 OracleSessionStateStore Public Methods**

Public Methods	Description
<a href="#">CreateNewStoreData</a>	Creates a new SessionStateStoreData object for the current request
<a href="#">CreateUninitializedItem</a>	Adds a new session state item to the database
<a href="#">Dispose</a>	Releases all the resources for this instance
<a href="#">EndRequest</a>	Allows the OracleSessionStateStore object to perform any cleanup that may be required for the current request
<a href="#">GetItem</a>	Returns a read-only session item from the database
<a href="#">GetItemExclusive</a>	Locks and returns a session item from the database
<a href="#">Initialize</a>	Initializes the provider with the property values specified in the ASP.NET application configuration file
<a href="#">InitializeRequest</a>	Performs any per-request initializations that the OracleSessionStateStore provider requires
<a href="#">ReleaseItemExclusive</a>	Releases the lock on a session item in the database, if multiple attempts to retrieve the session item fail
<a href="#">RemoveItem</a>	Removes the specified session item from the database
<a href="#">ResetItemTimeout</a>	Resets the expiration date and timeout for a session item in the database
<a href="#">SetAndReleaseItemExclusive</a>	Updates the session time information in the database with the specified session item and releases the lock

**Table 5-3 (Cont.) OracleSessionStateStore Public Methods**

Public Methods	Description
<a href="#">SetItemExpireCallback</a>	Returns a <code>false</code> value to indicate that callbacks for expired sessions are not supported

**See Also:**

- "Oracle Providers for ASP.NET Assembly" on page 1-4
- [OracleSessionStateStore Class](#)

## OracleSessionStateStore Constructors

The `OracleSessionStateStore` constructor instantiates a new instance of the `OracleSessionStateStore` class.

### Overload List:

- [OracleSessionStateStore\(\)](#)

This constructor creates an instance of the `OracleSessionStateStore` class.

#### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleSessionStateStore Class](#)
- [OracleSessionStateStore Members](#)

### OracleSessionStateStore()

This constructor instantiates a new instance of the `OracleSessionStateStore` class.

#### Declaration

```
// C#  
public OracleSessionStateStore();
```

#### Remarks

This constructor creates a new instance of the `OracleSessionStateStore` class.

#### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleSessionStateStore Class](#)
- [OracleSessionStateStore Members](#)



## OracleSessionStateStore Public Properties

The OracleSessionStateStore public properties are listed in [Table 5-4](#).

**Table 5-4 OracleSessionStateStore Public Properties**

Public Properties	Description
<a href="#">CommandTimeout</a>	Gets the number of seconds that the command is allowed to execute before it is terminated with an exception
Description	Inherited from System.Configuration.Provider.Providerbase
Name	Inherited from System.Configuration.Provider.Providerbase

### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleSessionStateStore Class](#)
- [OracleSessionStateStore Members](#)

### CommandTimeout

This property gets the number of seconds that the command is allowed to execute before it is terminated with an exception.

### Declaration

```
// C#
public int CommandTimeout {get;}
```

### Property Value

An int.

### Remarks

To customize a provider, ASP.NET developers can set an integer value for this property through the web.config file using the commandTimeout attribute.

The default value is 30 seconds. The attribute name in the configuration file is case-sensitive.

### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleSessionStateStore Class](#)
- [OracleSessionStateStore Members](#)

## OracleSessionStateStore Public Methods

The `OracleSessionStateStore` public methods are listed in [Table 5-5](#).

**Table 5-5 OracleSessionStateStore Public Methods**

Public Methods	Description
<a href="#">CreateNewStoreData</a>	Creates a new <code>SessionStateStoreData</code> object for the current request
<a href="#">CreateUninitializedItem</a>	Adds a new session state item to the database
<a href="#">Dispose</a>	Releases all the resources for this instance
<a href="#">EndRequest</a>	Allows the <code>OracleSessionStateStore</code> object to perform any cleanup that may be required for the current request
<a href="#">GetItem</a>	Returns a read-only session item from the database
<a href="#">GetItemExclusive</a>	Locks and returns a session item from the database
<a href="#">Initialize</a>	Initializes the provider with the property values specified in the ASP.NET application configuration file
<a href="#">InitializeRequest</a>	Performs any per-request initializations that the <code>OracleSessionStateStore</code> provider requires
<a href="#">ReleaseItemExclusive</a>	Releases the lock on a session item in the database, if multiple attempts to retrieve the session item fail
<a href="#">RemoveItem</a>	Removes the specified session item from the database
<a href="#">ResetItemTimeout</a>	Resets the expiration date and timeout for a session item in the database
<a href="#">SetAndReleaseItemExclusive</a>	Updates the session time information in the database with the specified session item and releases the lock
<a href="#">SetItemExpireCallback</a>	Returns a <code>false</code> value to indicate that callbacks for expired sessions are not supported

### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleSessionStateStore Class](#)
- [OracleSessionStateStore Members](#)

### CreateNewStoreData

This method creates a new `SessionStateStoreData` object for the current request.

#### Declaration

```
// C#
public override SessionStateStoreData CreateNewStoreData(HttpContext context,
    int timeout);
```

#### Parameters

- `context`  
The `HttpContext` object for the current request.
- `timeout`

The timeout value for the `SessionStateStoreData` object that is created.

### Return Value

A new `SessionStateStoreData` object for the current request.

### Remarks

This method creates a new `SessionStateStoreData` object for the current request based on the `HttpContext` and `timeout` values. The `SessionStateModule` calls this method at the beginning of a request for an ASP.NET page, if the request does not contain a session ID or if the request contains a session ID for a session that is not found in the database. This method creates a new `SessionStateStoreData` object with an empty `ISessionStateItemCollection` object, an `HttpStaticObjectsCollection` collection, and the specified timeout value.

#### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleSessionStateStore Class](#)
- [OracleSessionStateStore Members](#)

## CreateUninitializedItem

This method adds a new session state item to the database.

### Declaration

```
// C#  
public override void CreateUninitializedItem(HttpContext context, string id,  
    int timeout);
```

### Parameters

- *context*  
The `HttpContext` object for the current request.
- *id*  
The session ID for the current request.
- *timeout*  
The timeout value for the current request.

### Exceptions

`ArgumentNullException` - The input parameter is null.

`OracleException` - An Oracle-related error has occurred.

### Remarks

This method adds an uninitialized session state entry into the database and is called when the `cookieless` and `regenerateExpiredId` attributes are both set to `true`.

After a new session ID is created, this method is called to store an uninitialized entry with this new session ID in the database. The browser is redirected to the URL containing the new session ID. The new session ID exists in the database, so there is no conflict with an expired session ID.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleSessionStateStore Class](#)
- [OracleSessionStateStore Members](#)

## Dispose

This method releases all the resources for this instance.

**Declaration**

```
// C#  
public override void Dispose();
```

**Remarks**

This method releases all the resources for this instance when the application domain is closed.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleSessionStateStore Class](#)
- [OracleSessionStateStore Members](#)

## EndRequest

This method allows the `OracleSessionStateStore` object to perform any cleanup that may be required for the current request.

**Declaration**

```
// C#  
public override void EndRequest(HttpContext context);
```

**Parameters**

- *context*

The `HttpContext` object for the current request.

**Remarks**

This method is called by the `SessionStateModule` object at the end of a request.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleSessionStateStore Class](#)
- [OracleSessionStateStore Members](#)

## GetItem

This method returns a read-only session item from the database.

## Declaration

```
// C#  
public override SessionStateStoreData GetItem(HttpContext context, string id,  
    out bool locked, out TimeSpan lockAge, out Object lockId,  
    out SessionStateActions actions);
```

## Parameters

- *context*  
The `HttpContext` object for the current request.
- *id*  
The session ID for the current request.
- *locked*  
A Boolean value that is `true` if the session item is locked in the database; otherwise, it is `false`.
- *lockAge*  
A `TimeSpan` object that indicates the amount of time the session item has been locked in the database.
- *lockId*  
A lock identifier object.
- *actions*  
A `SessionStateActions` enumeration value that indicates whether the session is uninitialized and cookieless.

## Return Value

A `SessionStateStoreData` object that contains session information from the database.

## Exceptions

`ArgumentNullException` - The input parameter is null.

`OracleException` - An Oracle-related error has occurred.

`System.Configuration.Provider.ProviderException` - The session state information is invalid and might be corrupted.

## Remarks

This method returns a read-only `SessionStateStoreData` object from the database and updates the expiration date of the session item. This method is called by the session state service at the beginning of a request. It is called only if the `EnableSessionState` attribute in the page is set to `ReadOnly`.

If no session data is found, then the *locked* out parameter is set to `false` and a null reference is returned. The session state service then calls the `CreateNewStoreData` method to create a new session item in the database.

If the session data is locked in the database, then the *locked* out parameter is set to `true`, the *lockAge* parameter is set to the amount of time the session item has been locked in the database, the *lockId* parameter is set to the lock identifier and a null reference is returned. The session state service then keeps calling this method at half-second intervals. If the *lockAge* value exceeds the `HttpRuntimeSection.ExecutionTimeout` value, then the session state service calls the

`ReleaseItemExclusive` method to release the lock. It then calls the `GetItem` method again.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleSessionStateStore Class](#)
- [OracleSessionStateStore Members](#)

## GetItemExclusive

This method locks a session item and returns it from the database.

### Declaration

```
// C#
public override SessionStateStoreData GetItemExclusive(HttpContext context,
    string id, out bool locked, out TimeSpan lockAge, out Object lockId,
    out SessionStateActions actions);
```

### Parameters

- *context*  
The `HttpContext` object for the current request.
- *id*  
The session ID for the current request.
- *locked*  
A Boolean value that is true if the session item was successfully locked in the database; otherwise, it is false.
- *lockAge*  
A `TimeSpan` object that indicates the amount of time the session item has been locked in the database.
- *lockId*  
A lock identifier object.
- *actions*  
A `SessionStateActions` enumeration value that indicates whether the session is uninitialized and cookieless.

### Return Value

A `SessionStateStoreData` object that contains session information from the database.

### Exceptions

`ArgumentNullException` - The input parameter is null.

`OracleException` - An Oracle-related error has occurred.

`System.Configuration.Provider.ProviderException` - The session state information is invalid and might be corrupted.

### Remarks

This method returns a `SessionStateStoreData` object from the database and updates the expiration date of the session item. This method is called only if the attribute in the page is set to the default value of true. The session item is retrieved only if no other requests are currently using it. The session item in the database is locked for the duration of the request.

If no session data is found, the `locked` out parameter is set to false and a null reference is returned. The session state service then calls the `CreateNewStoreData` method to create a `newsession` item in the database.

If the session data is locked in the database, then the `locked` parameter is set to true, the `lockAge` parameter is set to the amount of time the session item has been locked in the database, the `lockId` parameter is set to the lock identifier and a null reference is returned. The session state service then keeps calling this method at half-second intervals. If the `lockAge` value exceeds the `ExecutionTimeout` value, then the session state service calls the `ReleaseItemExclusive` method to release the lock. It then calls the `GetItemExclusive` method again.

#### See Also:

- ["Oracle Providers for ASP.NET Assembly" on page 1-4](#)
- [OracleSessionStateStore Class](#)
- [OracleSessionStateStore Members](#)

## Initialize

This method initializes the provider with the property values specified in the ASP.NET application configuration file (`web.config`).

### Declaration

```
// C#  
public override void Initialize(string name, NameValueCollection config);
```

### Parameters

- `name`  
The name of the provider instance to initialize.
- `config`  
A `Systems.Collections.Specialized.NameValueCollection` object that contains the names and values of configuration options for the provider.

### Exceptions

`ArgumentNullException` - The `config` parameter is null.

`System.Configuration.Provider.ProviderException` - The `connectionStringName` attribute is empty or does not exist in the configuration file, or an invalid attribute is found in the configuration file.

### Remarks

The `Initialize` method is not intended to be called directly by the application.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly" on page 1-4](#)
- [OracleSessionStateStore Class](#)
- [OracleSessionStateStore Members](#)

**InitializeRequest**

This method performs any per-request initializations that the `OracleSessionStateStore` provider requires.

**Declaration**

```
// C#  
public override void InitializeRequest(HttpContext context);
```

**Parameters**

- *context*

The `HttpContext` object for the current request.

**Exceptions**

`ArgumentNullException` - The *context* parameter is null.

**Remarks**

This method is called by the session state service before calling any other methods.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly" on page 1-4](#)
- [OracleSessionStateStore Class](#)
- [OracleSessionStateStore Members](#)

**ReleaseItemExclusive**

This method forcibly releases the lock on a session item in the database, if multiple attempts to retrieve the session item fail.

**Declaration**

```
// C#  
public override void ReleaseItemExclusive(HttpContext context, string id,  
    Object lockId);
```

**Parameters**

- *context*

The `HttpContext` object for the current request.

- *id*

The session ID for the current request.

- *lockId*

The lock identifier for the current request.



### Exceptions

`ArgumentNullException` - The input parameter is null.

`OracleException` - An Oracle-related error has occurred.

### Remarks

This method is called by the session state service to release the lock on a session item in the database and update the expiration date. The `SessionStateModule` calls this method at the end of a request if the session values are unchanged or when the lock has exceeded the `HttpRequestSection.ExecutionTimeout` property value.

#### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleSessionStateStore Class](#)
- [OracleSessionStateStore Members](#)

## RemoveItem

This method removes the specified session item from the database.

### Declaration

```
// C#  
public override void RemoveItem(HttpContext context, string id, Object lockId,  
    SessionStateStoreData item);
```

### Parameters

- *context*  
The `HttpContext` object for the current request.
- *id*  
The session ID for the current request.
- *lockId*  
The lock identifier for the current request.
- *item*  
The session item to remove from the database.

### Exceptions

`ArgumentNullException` - The input parameter is null.

`OracleException` - An Oracle-related error has occurred.

### Remarks

The session state service calls this method to remove the specified session item from the database. An application can call the `Session.Abandon` method to cancel a session.

#### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleSessionStateStore Class](#)
- [OracleSessionStateStore Members](#)

## ResetItemTimeout

This method resets the expiration date and timeout for a session item in the database.

### Declaration

```
// C#  
public override void ResetItemTimeout(HttpContext context, string id);
```

### Parameters

- *context*  
The `HttpContext` object for the current request.
- *id*  
The session ID for the current request.

### Exceptions

`ArgumentNullException` - The input parameter is null.

`OracleException` - An Oracle-related error has occurred.

### Remarks

The session state service calls this method to reset the expiration date and timeout for a session item in the database, to the current date and time.

#### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleSessionStateStore Class](#)
- [OracleSessionStateStore Members](#)

## SetAndReleaseItemExclusive

This method updates the session time information in the database with the specified session item, and releases the lock.

### Declaration

```
// C#  
public override void SetAndReleaseItemExclusive(HttpContext context, string id,  
    SessionStateStoreDataItem item, Object lockId, bool newItem);
```

### Parameters

- *context*  
The `HttpContext` object for the current request.
- *id*  
The session ID for the current request.
- *item*  
The session item containing new values to update the session item in the database with.
- *lockId*

The lock identifier for the current request.

- *newItem*

A Boolean value that indicates whether the session item is new in the database. A false value indicates an existing item.

### Exceptions

*ArgumentNullException* - The input parameter is null.

*OracleException* - An Oracle-related error has occurred.

### Remarks

If the session items have been modified, the session state service calls this method at the end of a request, to either create a new item or update an existing session item in the database with the provided session values. This method also updates the expiration time for the session item and releases the lock on the session data.

#### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleSessionStateStore Class](#)
- [OracleSessionStateStore Members](#)

## SetItemExpireCallback

This method returns a false value to indicate that callbacks for expired sessions are not supported.

### Declaration

```
// C#  
public override bool SetItemExpireCallback(SessionStateItemExpireCallback  
    expireCallback);
```

### Parameters

- *expireCallback*

The delegate for the *Session\_OnEnd* event defined in the *Global.asax* file.

### Return Value

A false value.

### Remarks

This method always returns a false value to indicate that callbacks for expired sessions are not supported.

#### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleSessionStateStore Class](#)
- [OracleSessionStateStore Members](#)



---

---

## OracleProfileProvider

This chapter describes OracleProfileProvider class.

**See Also:** ASP.NET profile properties and profile providers  
<http://msdn.microsoft.com/en-us/library/2y3fs9xs.aspx>

This chapter contains the following topic:

- [OracleProfileProvider Class](#)

---

## OracleProfileProvider Class

OracleProfileProvider enables ASP.NET developers to easily store Web site user profile information in an Oracle database.

### Class Inheritance

System.Object

System.Configuration.Provider.ProviderBase

System.Configuration.SettingsProvider

System.Web.Profile.ProfileProvider

Oracle.Web.Profile.OracleProfileProvider

### Declaration

```
// C#  
public class OracleProfileProvider: ProfileProvider
```

### Thread Safety

All public static methods are thread-safe, although instance members are not guaranteed to be thread-safe.

### Remarks

This class allows ASP.NET applications to store and manage profile information in an Oracle database.

### Example

The following is a `web.config` file example for an ASP.NET application that uses `OracleProfileProvider` as the default provider. This configuration uses the connection string and default attribute values specified in the `machine.config` file. Profile properties are specified in the properties section.

This example also enables anonymous identification and allows anonymous users to set properties.

```
<?xml version="1.0"?>  
<configuration xmlns="http://schemas.microsoft.com/.NetConfiguration/v2.0">  
  <system.web>  
    <anonymousIdentification enabled="true"/>  
    <profile enabled="true" defaultProvider="OracleProfileProvider">  
      <!-- Profile properties -->  
      <properties>  
        <add name="hire_date" allowAnonymous="true" type="DateTime"/>  
        <add name="location" allowAnonymous="true"  
          defaultValue="Redwood Shores"/>  
        <add name="experience" allowAnonymous="true" type="int"/>  
      </properties>  
    </profile>  
  </system.web>  
</configuration>
```

The following is a `web.config` file example for an ASP.NET application that uses an `OracleProfileProvider` with customized settings and an application-specific connection string. Profile properties are specified in the properties section. This

example also enables anonymous identification and allows anonymous users to set properties.

```
<?xml version="1.0"?>
<configuration xmlns=
  "http://schemas.microsoft.com/.NetConfiguration/v2.0">
  <connectionStrings>
    <add name="my_profile_app_con_string" connectionString=
      "User Id=scott;Password=tiger;Data Source=Oracle"/>
  </connectionStrings>
  <system.web>
    <!-- Enable and customize OracleProfileProvider settings -->
    <anonymousIdentification enabled="true"/>
    <profile enabled="true" defaultProvider="MyOracleProfileProvider">
      <providers>
        <add name="MyOracleProfileProvider"
          type="Oracle.Web.Profile.OracleProfileProvider,
            Oracle.Web, Version=2.112.2.0, Culture=neutral,
            PublicKeyToken=89b483f429c47342"
          connectionStringName="my_profile_app_con_string"
          applicationName="my_profile_app"/>
      </providers>
    <!-- Profile properties -->
    <properties>
      <add name="hire_date" allowAnonymous="true" type="DateTime"/>
      <add name="location" allowAnonymous="true"
        defaultValue="Redwood Shores"/>
      <add name="experience" allowAnonymous="true" type="int"/>
    </properties>
  </profile>
</system.web>
</configuration>
```

Note that the applicationName attribute should be set to a unique value for each ASP.NET application.

## Requirements

Namespace: Oracle.Web.Profile

Assembly: Oracle.Web.dll

Oracle Providers for ASP.NET Version: Oracle Providers for ASP.NET 2.0 and Oracle Providers for ASP.NET 4

### See Also:

- ["Oracle Providers for ASP.NET Assembly" on page 1-4](#)
- [OracleProfileProvider Members](#)
- [OracleProfileProvider Constructors](#)
- [OracleProfileProvider Static Methods](#)
- [OracleProfileProvider Public Properties](#)
- [OracleProfileProvider Public Methods](#)

## OracleProfileProvider Members

OracleProfileProvider members are listed in the following tables.

### OracleProfileProvider Constructors

The OracleProfileProvider constructor is listed in [Table 6-1](#).

**Table 6-1 OracleProfileProvider Constructor**

Constructor	Description
<a href="#">OracleProfileProvider Constructors</a>	Instantiates a new instance of the OracleProfileProvider class

### OracleProfileProvider Static Methods

OracleProfileProvider static methods are listed in [Table 6-2](#).

**Table 6-2 OracleProfileProvider Static Methods**

Static Methods	Description
Equals	Inherited from System.Object (Overloaded)
ReferenceEquals	Inherited from System.Object

### OracleProfileProvider Public Properties

OracleProfileProvider public properties are listed in [Table 6-3](#).

**Table 6-3 OracleProfileProvider Public Properties**

Public Properties	Description
<a href="#">ApplicationName</a>	Gets or sets the name of the application that groups the profile information
<a href="#">CommandTimeout</a>	Gets the number of seconds that the command is allowed to execute before it is terminated with an exception
Description	Inherited from System.Configuration.Provider.Providerbase
Name	Inherited from System.Configuration.Provider.Providerbase

### OracleProfileProvider Public Methods

OracleProfileProvider public methods are listed in [Table 6-4](#).

**Table 6-4 OracleProfileProvider Public Methods**

Public Methods	Description
<a href="#">DeleteInactiveProfiles</a>	Deletes user profile data that has its last activity date on or before the specified date and time
<a href="#">DeleteProfiles</a>	Deletes profile properties and information from the data source for the supplied profile collection or list of user names (Overloaded)
Equals	Inherited from System.Object (Overloaded)
<a href="#">FindInactiveProfilesByUserName</a>	Retrieves inactive profile information for the specified user name
<a href="#">FindProfilesByUserName</a>	Retrieves profile information for the specified user name



**Table 6–4 (Cont.) OracleProfileProvider Public Methods**

Public Methods	Description
<a href="#">GetAllInactiveProfiles</a>	Retrieves all profile information for profiles with the last activity date on or before the specified date and time
<a href="#">GetAllProfiles</a>	Retrieves all profile information from the data source
GetHashCode	Inherited from System.Object
<a href="#">GetNumberOfInactiveProfiles</a>	Returns the count of profiles where the last activity date is on or before the specified date and time
<a href="#">GetPropertyValues</a>	Retrieves profile properties and values from the Oracle profile database
GetType	Inherited from System.Object
<a href="#">Initialize</a>	Initializes the OracleProfileProvider instance with the property values specified in the ASP.NET application configuration file (web.config)
<a href="#">SetPropertyValues</a>	Updates the Oracle profile database with the specified profile property values
ToString	Inherited from System.Object

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleProfileProvider Class](#)

## OracleProfileProvider Constructors

This constructor instantiates a new instance of the `OracleProfileProvider` class.

### Overload List:

- [OracleProfileProvider\(\)](#)

This constructor creates an instance of the `OracleProfileProvider` class.

#### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleProfileProvider Class](#)
- [OracleProfileProvider Members](#)

## OracleProfileProvider()

This constructor instantiates a new instance of the `OracleProfileProvider` class.

### Declaration

```
// C#  
public OracleProfileProvider();
```

### Remarks

This constructor is called by ASP.NET to create an instance of the `OracleProfileProvider` class as specified in the configuration file for the application. Initialization values for an `OracleProfileProvider` instance are passed through the `Initialize` method.

This constructor is not intended to be used directly by the application.

#### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleProfileProvider Class](#)
- [OracleProfileProvider Members](#)

## OracleProfileProvider Static Methods

OracleProfileProvider static methods are listed in [Table 6-5](#).

**Table 6-5 OracleProfileProvider Static Methods**

Static Methods	Description
Equals	Inherited from System.Object (Overloaded)
ReferenceEquals	Inherited from System.Object

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleProfileProvider Class](#)
- [OracleProfileProvider Members](#)

## OracleProfileProvider Public Properties

OracleProfileProvider public properties are listed in [Table 6–6](#).

**Table 6–6 OracleProfileProvider Public Properties**

Public Properties	Description
<a href="#">ApplicationName</a>	Gets or sets the name of the application that groups the profile information
<a href="#">CommandTimeout</a>	Gets the number of seconds that the command is allowed to execute before it is terminated with an exception
Description	Inherited from <code>System.Configuration.Provider.Providerbase</code>
Name	Inherited from <code>System.Configuration.Provider.Providerbase</code>

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleProfileProvider Class](#)
- [OracleProfileProvider Members](#)

### ApplicationName

This property gets or sets the name of the application that groups the profile information.

**Declaration**

```
// C#
public override string ApplicationName{get; set;}
```

**Property Value**

The name of the application. If the `applicationName` attribute is not specified in the application configuration file, or if the value is an empty string, then this property is set to the application virtual path.

**Exceptions**

`HttpException` - The `ApplicationName` property was set by a caller that does not have high ASP.NET hosting permission.

`System.Configuration.Provider.ProviderException` - The application name supplied exceeds 256 characters.

`ArgumentException` - The application name supplied is an empty string or a null reference.

**Remarks**

The string value of the `ApplicationName` property is used for organizing user information.

Multiple ASP.NET applications can use the same data source and create duplicate user names because user information is stored uniquely for each application name. This property can be set programmatically, or it can be set declaratively in the configuration file for the Web application using the `applicationName` attribute. The attribute name in the configuration file is case-sensitive.

---

The `ApplicationName` property is not thread-safe. It is recommended that application code not allow users to set the `ApplicationName` property in Web applications.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleProfileProvider Class](#)
- [OracleProfileProvider Members](#)

**CommandTimeout**

This property gets the number of seconds that the command is allowed to execute before it is terminated with an exception.

**Declaration**

```
// C#  
public int CommandTimeout {get;}
```

**Property Value**

An `int`.

**Remarks**

To customize a provider, ASP.NET developers can set an integer value for this property through the `web.config` file using the `commandTimeout` attribute.

The default value is 30 seconds. The attribute name in the configuration file is case-sensitive.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleProfileProvider Class](#)
- [OracleProfileProvider Members](#)

## OracleProfileProvider Public Methods

OracleProfileProvider public methods are listed in [Table 6-7](#).

**Table 6-7 OracleProfileProvider Public Methods**

Public Methods	Description
<a href="#">DeleteInactiveProfiles</a>	Deletes user profile data that has its last activity date on or before the specified date and time
<a href="#">DeleteProfiles</a>	Deletes profile properties and information from the data source for the supplied profile collection or list of user names (Overloaded)
Equals	Inherited from System.Object (Overloaded)
<a href="#">FindInactiveProfilesByUserName</a>	Retrieves inactive profile information for the specified user name
<a href="#">FindProfilesByUserName</a>	Retrieves profile information for the specified user name
<a href="#">GetAllInactiveProfiles</a>	Retrieves all profile information for profiles with the last activity date on or before the specified date and time
<a href="#">GetAllProfiles</a>	Retrieves all profile information from the data source
GetHashCode	Inherited from System.Object
<a href="#">GetNumberOfInactiveProfiles</a>	Returns the count of profiles where the last activity date is on or before the specified date and time
<a href="#">GetPropertyValues</a>	Retrieves profile properties and values from the Oracle profile database
GetType	Inherited from System.Object
<a href="#">Initialize</a>	Initializes the OracleProfileProvider instance with the property values specified in the ASP.NET application configuration file (web.config)
<a href="#">SetPropertyValues</a>	Updates the Oracle profile database with the specified profile property values
ToString	Inherited from System.Object

### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleProfileProvider Class](#)
- [OracleProfileProvider Members](#)

### DeleteInactiveProfiles

This method deletes user profile data that has its last activity date on or before the specified date and time.

#### Declaration

```
// C#
public override int DeleteInactiveProfiles(ProfileAuthenticationOption
    profileAuthenticationOption, DateTime inactiveSinceDateTime);
```

**Parameters**

- *profileAuthenticationOption*  
The options are *Anonymous*, *Authenticated*, or *All*, to indicate which profiles to delete.
- *inactiveSinceDateTime*  
The cut-off date and time that indicate a profile is inactive.

**Return Value**

An integer value that indicates the number of inactive profiles deleted from the data source.

**Remarks**

This method deletes inactive profile data from the data source for the application specified by the *applicationName* attribute in the configuration file. The *profileAuthenticationOption* parameter specifies whether to search only anonymous profiles, only authenticated profiles, or all profiles. This method deletes any profile with a last activity date and time occurring on or before the specified *inactiveSinceDateTime* parameter value.

The delete profiles operation is a transactional operation. If an error is encountered, the transaction is rolled back and no changes are made.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleProfileProvider Class](#)
- [OracleProfileProvider Members](#)

**DeleteProfiles**

This method deletes profile properties and information from the data source for the supplied profile collection or list of user names.

**Overload List**

- [DeleteProfiles\(ProfileInfoCollection\)](#)  
This method deletes profile properties and information from the data source for the supplied profile collection.
- [DeleteProfiles\(string\[\]\)](#)  
This method deletes profile properties and information from the data source for the supplied list of user names.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleProfileProvider Class](#)
- [OracleProfileProvider Members](#)

**DeleteProfiles(ProfileInfoCollection)**

This method deletes profile properties and information from the data source for the supplied profile collection.

**Declaration**

```
// C#  
public override int DeleteProfiles(ProfileInfoCollection profileInfoCollection);
```

**Parameters**

- *profileInfoCollection*

A *ProfileInfoCollection* object that contains profile information for profiles to be deleted.

**Return Value**

An integer value indicating the number of profiles that were deleted from the data source.

**Exceptions**

*ArgumentException* - One of the following conditions exists:

- The value of *Count* in the *profileInfoCollection* parameter is 0.
- One of the *ProfileInfo* objects in the *profileInfoCollection* collection has an invalid *UserName* property, such as an empty string, exceeds 256 characters, or contains a comma.

*ArgumentNullException* - One of the following conditions exists:

- The *profileInfoCollection* parameter is a null reference.
- One of the *ProfileInfo* objects in *profileInfoCollection* collection has a *UserName* property that is a null reference.

**Remarks**

This method deletes all profile properties and information for the supplied profile collection from the data source for the application specified by the *applicationName* attribute in the configuration file. A *ProfileInfoCollection* object can be obtained from the *GetAllProfiles*, *GetAllInactiveProfiles*, *FindProfilesByUserName*, and *FindInactiveProfilesByUserName* methods.

The value returned may be different from the *Count* value of the supplied collection, because some of the profiles in the supplied collection are no longer found in the data source.

The delete profiles operation is a transactional operation. If an error is encountered, the transaction is rolled back and no changes are made.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleProfileProvider Class](#)
- [OracleProfileProvider Members](#)

**DeleteProfiles(string[])**

This method deletes profile properties and information from the data source for the supplied list of user names.

**Declaration**

```
// C#
```



```
public override int DeleteProfiles(string[] userNames);
```

### Parameters

- *userNames*

A string array of user names whose profiles are to be deleted.

### Return Value

An integer value indicating the number of profiles that were deleted from the data source.

### Exceptions

*ArgumentNullException* - The *userNames* parameter is a null reference or one of the items in *userNames* array has a null reference.

*ArgumentException* - One of the following conditions exists:

- The length of the *userNames* array is 0.
- One of the items in the *userNames* array has an invalid user name, such as an empty string, exceeds 256 characters, or contains a comma.
- There are duplicated user names in the *userNames* array.

### Remarks

This method deletes all profile properties and information from the data source for the supplied list of user names for the application specified by the *applicationName* attribute in the configuration file.

The value returned may be different from the length of the supplied string array of user names because some of the profiles are no longer found in the data source.

The delete profiles operation is a transactional operation. If an error is encountered, then the transaction is rolled back and no changes are made.

#### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleProfileProvider Class](#)
- [OracleProfileProvider Members](#)

## FindInactiveProfilesByUserName

This method retrieves inactive profile information for the specified user name.

### Declaration

```
// C#
public override ProfileInfoCollection FindInactiveProfilesByUserName
    (ProfileAuthenticationOption profileAuthenticationOption, string userName,
    DateTime inactiveSinceDateTime, int pageIndex, int pageSize,
    out int totalRecords);
```

### Parameters

- *profileAuthenticationOption*

Anonymous, Authenticated, or All profiles to be searched to find inactive profiles.

- *userName*  
The user name to match.
- *inactiveSinceDateTime*  
The cut-off date and time that indicate a profile is inactive.
- *pageIndex*  
The zero-based index of the results page.
- *pageSize*  
The size of the page of the results page.
- *totalRecords*  
The total number of profiles.

### Return Value

A *ProfileInfoCollection* object that contains inactive user profiles where the user name matches the supplied user name.

### Exceptions

*ArgumentException* - One of the following conditions exists:

- The *userName* parameter is an empty string or exceeds 256 characters.
- The *pageSize* parameter is less than 1.
- The *pageIndex* parameter is less than 0 or *pageIndex* multiplied by *pageSize* is larger than the *Int32.MaxValue*.

*ArgumentNullException* - The *userName* parameter is a null reference.

### Remarks

This method retrieves inactive profiles from the data source for the application specified by the *applicationName* attribute in the configuration file. The *profileAuthenticationOption* parameter specifies whether to search only anonymous profiles, only authenticated profiles, or all profiles. The *OracleProfileProvider* object searches for a match of the supplied *userName* parameter using the LIKE keyword and supports wildcard characters using the percent sign (%). This method retrieves profiles with a last activity date and time on or before the specified *inactiveSinceDateTime* parameter value.

The results returned by this method are constrained by the *pageIndex* and *pageSize* parameters. The *pageSize* parameter identifies the number of *ProfileInfo* objects to return in the *ProfileInfoCollection* object. The *pageIndex* parameter identifies which page of results to return. The *totalRecords* parameter is an out parameter for the total number of inactive user profiles that match the *userName* and *inactiveSinceDateTime* parameters.

#### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleProfileProvider Class](#)
- [OracleProfileProvider Members](#)

## FindProfilesByUserName

This method retrieves profile information for the specified user name.

### Declaration

```
// C#
public override ProfileInfoCollection FindProfilesByUserName
    (ProfileAuthenticationOption profileAuthenticationOption, string userName,
    int pageIndex, int pageSize, out int totalRecords);
```

### Parameters

- *profileAuthenticationOption*  
Anonymous, Authenticated, or All profiles to be searched to find active profiles.
- *userName*  
The user name to match.
- *pageIndex*  
The zero-based index of the results page.
- *pageSize*  
The size of the page of results page.
- *totalRecords*  
The total number of profiles.

### Return Value

A ProfileInfoCollection object that contains user profiles where the user name matches the supplied user name.

### Exceptions

ArgumentException - One of the following conditions exists:

- The *userName* parameter is an empty string or exceeds 256 characters.
- The *pageSize* parameter value is less than 1.
- The *pageIndex* parameter value is less than 0 or *pageIndex* multiplied by *pageSize* is larger than `Int32.MaxValue`.

ArgumentNullException - The *userName* parameter is a null reference.

### Remarks

This method retrieves profiles from the data source for the application specified by the *applicationName* attribute in the configuration file. The *profileAuthenticationOption* parameter specifies whether to search only anonymous profiles, only authenticated profiles, or all profiles. The OracleProfileProvider object searches for a match of the *userName* parameter supplied using the LIKE keyword and supports wildcard characters using the percent sign (%).

The results returned by this method are constrained by the *pageIndex* and *pageSize* parameters. The *pageSize* parameter identifies the number of ProfileInfo objects to return in the ProfileInfoCollection object. The *pageIndex* parameter identifies which results page to return. The *totalRecords* parameter is an out parameter for the total number of inactive user profiles that matched the *userName* parameter.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleProfileProvider Class](#)
- [OracleProfileProvider Members](#)

**GetAllInactiveProfiles**

This method retrieves all profile information for profiles with the last activity date on or before the specified date and time.

**Declaration**

```
// C#
public override ProfileInfoCollection GetAllInactiveProfiles
    (ProfileAuthenticationOption profileAuthenticationOption, DateTime
    inactiveSinceDateTime, int pageIndex, int pageSize, out int totalRecords);
```

**Parameters**

- *profileAuthenticationOption*  
Anonymous, Authenticated, or All profiles to be searched.
- *inactiveSinceDateTime*  
The cut-off date and time that indicate a profile is inactive.
- *pageIndex*  
The zero-based index of the results page.
- *pageSize*  
The size of the page of the results page.
- *totalRecords*  
The total number of profiles.

**Return Value**

A *ProfileInfoCollection* object that contains inactive user profiles that matches the supplied inactive date and time.

**Exceptions**

*ArgumentException* - One of the following conditions exists:

- The *pageSize* parameter value is less than 1.
- The *pageIndex* parameter value is less than 0 or *pageIndex* multiplied by *pageSize* is larger than *Int32.MaxValue*.

**Remarks**

This method retrieves inactive profiles from the data source for the application specified by the *applicationName* attribute in the configuration file. The *profileAuthenticationOption* parameter specifies whether to search only anonymous profiles, only authenticated profiles, or all profiles. This method retrieves profiles with a last activity date and time on or before the specified *inactiveSinceDateTime* parameter value.

The returned results are constrained by the *pageIndex* and *pageSize* parameters. The *pageSize* parameter identifies the number of *ProfileInfo* objects to return in the *ProfileInfoCollection* object. The *pageIndex* parameter identifies which page of results to return. Zero identifies the first page, as the value is zero-based. The *totalRecords* parameter is an out parameter for the total number of inactive user profiles that match the *inactiveSinceDateTime* parameter.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleProfileProvider Class](#)
- [OracleProfileProvider Members](#)

## GetAllProfiles

This method retrieves all profile information from the data source.

**Declaration**

```
// C#
public override ProfileInfoCollection GetAllProfiles(ProfileAuthenticationOption
    profileAuthenticationOption, int pageIndex, int pageSize,
    out int totalRecords);
```

**Parameters**

- *profileAuthenticationOption*  
Anonymous, Authenticated, or All profiles to be searched.
- *pageIndex*  
The 0-based index of the results page.
- *pageSize*  
The size of the page of the results page
- *totalRecords*  
The total number of profiles.

**Return Value**

A *ProfileInfoCollection* object that contains all user profiles from the data source.

**Exceptions**

*ArgumentException* - One of the following conditions exists:

- The *pageSize* parameter is less than 1.
- The *pageIndex* parameter is less than 0 or *pageIndex* multiplied by *pageSize* is larger than *Int32.MaxValue*.

**Remarks**

This method retrieves all profiles from the data source for the application specified by the *applicationName* attribute in the configuration file. The *profileAuthenticationOption* parameter specifies whether to search only anonymous profiles, only authenticated profiles, or all profiles.

The returned results are constrained by the *pageIndex* and *pageSize* parameters. The *pageSize* parameter identifies the number of *ProfileInfo* objects to return in the *ProfileInfoCollection* object. The *pageIndex* parameter identifies which page of results to return. The *totalRecords* parameter is an out parameter for the total number of user profiles retrieved.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleProfileProvider Class](#)
- [OracleProfileProvider Members](#)

## GetNumberOfInactiveProfiles

This method returns the count of profiles where the last activity date is on or before the specified date and time.

**Declaration**

```
// C#
public override int GetNumberOfInactiveProfiles
    (ProfileAuthenticationOption profileAuthenticationOption,
     DateTime inactiveSinceDateTime);
```

**Parameters**

- *profileAuthenticationOption*  
Anonymous, Authenticated, or All profiles to be searched.
- *inactiveSinceDateTime*  
The cut-off date and time that indicate a profile is inactive.

**Return Value**

An integer value indicating the number of user profiles that match the inactive date and time supplied.

**Remarks**

This method returns a count of inactive profiles from the data source for the application specified by the *applicationName* attribute in the configuration file. The *profileAuthenticationOption* parameter specifies whether to search only anonymous profiles, only authenticated profiles, or all profiles. Of the searched user profiles, any profiles with a last activity date and time on or before the specified *inactiveSinceDateTime* parameter value are counted.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleProfileProvider Class](#)
- [OracleProfileProvider Members](#)

## GetPropertyValues

This method retrieves profile properties and values from the Oracle profile database.

## Declaration

```
// C#  
public override SettingsPropertyValueCollection GetPropertyValues(SettingsContext  
    settingsContext, SettingsPropertyCollection settingsPropertyCollection);
```

## Parameters

- *settingsContext*  
The SettingsContext object that contains user profile information.
- *settingsPropertyCollection*  
The SettingsPropertyCollection object that contains profile information for the properties to be retrieved.

## Return Value

A SettingsPropertyValueCollection object that contains profile property information and values.

## Remarks

This method retrieves profile properties and values from the Oracle database for the user profile specified by the context. Profile properties and values are returned as a collection of SettingsPropertyValue objects.

### See Also:

- ["Oracle Providers for ASP.NET Assembly" on page 1-4](#)
- [OracleProfileProvider Class](#)
- [OracleProfileProvider Members](#)

## Initialize

This method initializes the OracleProfileProvider instance with the property values specified in the ASP.NET application configuration file (web.config).

## Declaration

```
// C#  
public override void Initialize(string name, NameValueCollection config);
```

## Parameters

- *name*  
The name of the OracleProfileProvider instance to initialize.
- *config*  
A collection of the name/value pairs representing the provider-specific attributes specified in the configuration for the provider.

## Exceptions

ArgumentNullException - The *config* parameter is a null reference.

HttpException - The current trust level is less than Low.

InvalidOperationException - An attempt is made to call the Initialize method on a provider that has already been initialized.

`ArgumentNullException` - The `config` parameter is a null.

`System.Configuration.Provider.ProviderException` - One of the following conditions is true in the application configuration file:

- The `connectionStringName` attribute is empty or does not exist in the application configuration file.
- The value of the connection string for the `connectionStringName` attribute is empty or the specified `connectionStringName` value does not exist in the application configuration file.
- The `applicationName` attribute value exceeds 256 characters.
- The application configuration file for this `OracleProfileProvider` instance contains an unrecognized attribute.

### Remarks

The `Initialize` method sets options and property values for the provider instance, including provider-specific values and options specified in the machine configuration file (`machine.config`) or the ASP.NET application configuration file (`web.config`).

The `Initialize` method is not intended to be called directly by the application.

#### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleProfileProvider Class](#)
- [OracleProfileProvider Members](#)

## SetPropertyValues

This method updates the Oracle profile database with the specified profile property values.

### Declaration

```
// C#
public override void SetPropertyValues(SettingsContext settingsContext,
    SettingsPropertyValueCollection settingsPropertyValueCollection);
```

### Parameters

- `settingsContext`  
The `SettingsContext` object that contains user profile information.
- `settingsPropertyValueCollection`  
A `SettingsPropertyValueCollection` object that contains profile information and values for updating the user profile properties.

### Remarks

ASP.NET profile services use this method to update profile properties and values in the Oracle database for the user profile specified by the context.

Property values are set at the data source for the application specified by the `applicationName` attribute in the configuration file. Profile properties and values to be updated are specified as a collection of `SettingsPropertyValue` objects.



**See Also:**

- ["Oracle Providers for ASP.NET Assembly" on page 1-4](#)
- [OracleProfileProvider Class](#)
- [OracleProfileProvider Members](#)



---

---

# OracleWebEventProvider

This chapter describes the OracleWebEventProvider class.

**See Also:** ASP.NET health monitoring and web event provider  
[http://msdn.microsoft.com/en-us/library/ms178701\(VS.80\).aspx](http://msdn.microsoft.com/en-us/library/ms178701(VS.80).aspx)

This chapter contains the following topic:

- [OracleWebEventProvider Class](#)

## OracleWebEventProvider Class

The `OracleWebEventProvider` class allows ASP.NET applications to store Web events in an Oracle database.

### Class Inheritance

`System.Object`

`System.Configuration.Provider.ProviderBase`

`System.Web.Management.WebEventProvider`

`System.Web.Management.BufferedWebEventProvider`

`Oracle.Web.Management.OracleWebEventProvider`

### Declaration

```
// C#  
public class OracleWebEventProvider: BufferedWebEventProvider
```

### Thread Safety

All public static methods are thread-safe, although instance members are not guaranteed to be thread-safe.

### Remarks

This class allows ASP.NET applications to store Web event information in an Oracle database.

### Example

The following is a `web.config` example for an ASP.NET application that uses the `OracleWebEventProvider` class as the default provider. This configuration uses the connection string and default attribute values specified in the `machine.config` file.

Applications must provide any required configuration entries for event mapping, buffer modes, and rules in the `web.config` file, because the `machine.config` file does not provide these configuration entries. The following `web.config` file provides an example:

```
<?xml version="1.0"?>  
<configuration xmlns="http://schemas.microsoft.com/.NetConfiguration/v2.0">  
  <system.web>  
    <healthmonitoring enabled="true"/>  
    <bufferModes>  
      <add name="Notification"  
        maxBufferSize="1000"  
        maxFlushSize="200"  
        urgentFlushThreshold="500"  
        regularFlushInterval="00:00:6"  
        urgentFlushInterval="00:00:03"  
        maxBufferThreads="1"/>  
    </bufferModes>  
    <eventMappings>  
      <add name="CustomEvent"  
        type="CustomEventSource.CustomEvent, CustomEventSource"/>  
    </eventMappings>  
    <rules>
```

```

        <add name="CustomRule"
            eventName="CustomEvent "
            provider="OracleWebEventProvider"
            minInterval="00:00:00" />
    </rules>
</healthMonitoring>
</system.web>
</configuration>

```

The following is a `web.config` example for an ASP.NET application that uses an `OracleWebEventProvider` class as the default provider, using customized settings for the connection string name and application name, and an application-specific connection string, along with other configurations as described in the previous example:

```

<?xml version="1.0"?>
<configuration xmlns=
    "http://schemas.microsoft.com/.NetConfiguration/v2.0">
    <connectionStrings>
        <add name="my_webevent_app_con_string" connectionString=
            "User Id=scott;Password=tiger;Data Source=Oracle" />
    </connectionStrings>
    <system.web>
        <!-- Enable and customize OracleWebEventProvider -->
        <healthMonitoring enabled="true">
            <providers>
                <add name="CustomOracleWebEventProvider"
                    type="Oracle.Web.Management.OracleWebEventProvider,
                        Oracle.Web, Version=2.112.2.0, Culture=neutral,
                        PublicKeyToken=89b483f429c47342"
                    connectionStringName="my_webevent_app_con_string"
                    bufferMode="CustomBufferMode">
            </providers>
            <bufferModes>
                <add name="CustomBufferMode"
                    maxBufferSize="1000"
                    maxFlushSize="200"
                    urgentFlushThreshold="500"
                    regularFlushInterval="00:00:06"
                    urgentFlushInterval="00:00:03"
                    maxBufferThreads="1" />
            </bufferModes>
            <eventMappings>
                <add name="CustomEvent"
                    type="CustomEventSource.CustomEvent, CustomEventSource" />
            </eventMappings>
            <rules>
                <add name="CustomRule"
                    eventName="CustomEvent "
                    provider="CustomOracleWebEventProvider"
                    minInterval="00:00:00" />
            </rules>
        </healthMonitoring>
    </system.web>
</configuration>

```

Note that the `applicationName` attribute should be set to a unique value for each ASP.NET application.

### **Requirements**

Namespace: Oracle.Web.Management

Assembly: Oracle.Web.dll

Oracle Providers for ASP.NET Version: Oracle Providers for ASP.NET 2.0 and Oracle Providers for ASP.NET 4

#### **See Also:**

- ["Oracle Providers for ASP.NET Assembly" on page 1-4](#)
- [OracleWebEventProvider Members](#)
- [OracleWebEventProvider Constructors](#)
- [OracleWebEventProvider Static Methods](#)
- [OracleWebEventProvider Public Properties](#)
- [OracleWebEventProvider Public Methods](#)

## OracleWebEventProvider Members

OracleWebEventProvider members are listed in the following tables.

### OracleWebEventProvider Constructors

The OracleWebEventProvider constructor is listed in [Table 7-1](#).

**Table 7-1 OracleWebEventProvider Constructor**

Constructor	Description
<a href="#">OracleWebEventProvider Constructors</a>	Instantiates a new instance of the OracleWebEventProvider class

### OracleWebEventProvider Static Methods

OracleWebEventProvider static methods are listed in [Table 7-2](#).

**Table 7-2 OracleWebEventProvider Static Methods**

Static Methods	Description
Equals	Inherited from System.Object (Overloaded)
ReferenceEquals	Inherited from System.Object

### OracleWebEventProvider Public Properties

OracleWebEventProvider public properties are listed in [Table 7-3](#).

**Table 7-3 OracleWebEventProvider Public Properties**

Public Properties	Description
BufferMode	Inherited from System.Web.Management.BufferedWebEventProvider
<a href="#">CommandTimeout</a>	Gets the number of seconds that the command is allowed to execute before it is terminated with an exception
Description	Inherited from System.Configuration.Provider.ProviderBase
Name	Inherited from System.Configuration.Provider.ProviderBase
UseBuffering	Inherited from System.Web.Management.BufferedWebEventProvider

### OracleWebEventProvider Public Methods

OracleWebEventProvider public methods are listed in [Table 7-4](#).

**Table 7-4 OracleWebEventProvider Public Methods**

Public Method	Description
<a href="#">Initialize</a>	Initializes the OracleWebEventProvider instance with the property values specified in the ASP.NET application configuration file
<a href="#">ProcessEvent</a>	Processes the event passed to it as an argument
<a href="#">ProcessEventFlush</a>	Flushes the information passed to it as an argument
<a href="#">Shutdown</a>	Releases all resources
Flush	Inherited from System.BufferedWebEventProvider

**Table 7-4 (Cont.) OracleWebEventProvider Public Methods**

<b>Public Method</b>	<b>Description</b>
Equals(Overloaded)	Inherited from System.Object
GetHashCode	Inherited from System.Object
GetType	Inherited from System.Object
ToString	Inherited from System.Object

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleWebEventProvider Class](#)



## OracleWebEventProvider Constructors

This constructor creates an instance of the `OracleWebEventProvider` class.

### Overload List:

- [OracleWebEventProvider\(\)](#)

This constructor creates an instance of the `OracleWebEventProvider` class.

### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleWebEventProvider Class](#)
- [OracleWebEventProvider Members](#)

## OracleWebEventProvider()

This constructor creates an instance of the `OracleWebEventProvider` class.

### Declaration

```
// C#  
public OracleWebEventProvider();
```

### Remarks

This constructor creates a new instance of the `OracleWebEventProvider` class.

### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleWebEventProvider Class](#)
- [OracleWebEventProvider Members](#)

## OracleWebEventProvider Static Methods

The `OracleWebEventProvider` static methods are listed in [Table 7-5](#).

**Table 7-5** *OracleWebEventProvider Static Methods*

Static Methods	Description
<code>Equals</code>	Inherited from <code>System.Object</code> (Overloaded)
<code>ReferenceEquals</code>	Inherited from <code>System.Object</code>

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleWebEventProvider Class](#)
- [OracleWebEventProvider Members](#)

## OracleWebEventProvider Public Properties

The OracleWebEventProvider public properties are listed in [Table 7–6](#).

**Table 7–6 OracleWebEventProvider Public Properties**

Public Properties	Description
BufferMode	Inherited from System.Web.Management.BufferedWebEventProvider
<a href="#">CommandTimeout</a>	Gets the number of seconds that the command is allowed to execute before it is terminated with an exception
Description	Inherited from System.Configuration.Provider.ProviderBase
Name	Inherited from System.Configuration.Provider.ProviderBase
UseBuffering	Inherited from System.Web.Management.BufferedWebEventProvider

### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleWebEventProvider Class](#)
- [OracleWebEventProvider Members](#)

## CommandTimeout

This property gets the number of seconds that the command is allowed to execute before it is terminated with an exception.

### Declaration

```
// C#
public int CommandTimeout {get;}
```

### Property Value

An int.

### Remarks

To customize a provider, ASP.NET developers can set an integer value for this property through the web.config file using the commandTimeout attribute.

The default value is 30 seconds. The attribute name in the configuration file is case-sensitive.

### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleWebEventProvider Class](#)
- [OracleWebEventProvider Members](#)

## OracleWebEventProvider Public Methods

The `OracleWebEventProvider` public methods are listed in [Table 7-7](#).

**Table 7-7 OracleWebEventProvider Public Methods**

Public Method	Description
<a href="#">Initialize</a>	Initializes the <code>OracleWebEventProvider</code> instance with the property values specified in the ASP.NET application configuration file
<a href="#">ProcessEvent</a>	Processes the event passed to it as an argument
<a href="#">ProcessEventFlush</a>	Flushes the information passed to it as an argument
<a href="#">Shutdown</a>	Releases all resources
<code>Flush</code>	Inherited from <code>System.BufferedWebEventProvider</code>
<code>Equals(Overloaded)</code>	Inherited from <code>System.Object</code>
<code>GetHashCode</code>	Inherited from <code>System.Object</code>
<code>GetType</code>	Inherited from <code>System.Object</code>
<code>ToString</code>	Inherited from <code>System.Object</code>

### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleWebEventProvider Class](#)
- [OracleWebEventProvider Members](#)

## Initialize

This method initializes the `OracleWebEventProvider` instance with the property values specified in the ASP.NET application configuration file (`web.config`).

### Declaration

```
// C#
public override void Initialize(string name, NameValueCollection config);
```

### Parameters

- *name*  
The name of the `OracleWebEventProvider` instance to initialize.
- *config*  
A `Systems.Collections.Specialized.NameValueCollection` object that contains the names and values of configuration options for the `OracleWebEventProvider`.

### Exceptions

`InvalidOperationException` - If the `OracleWebEventProvider` instance is already initialized.

`ProviderException` - One of the following conditions exists:

- The `connectionStringName` attribute in the configuration file is null or empty.

- The connection string corresponding to value of the `connectionStringName` attribute is null or empty.
- An unrecognized attribute is found in the configuration file.
- Another error occurs during initialization of the provider.

**Remarks**

The `Initialize` method is not intended to be called directly by the application.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleWebEventProvider Class](#)
- [OracleWebEventProvider Members](#)

**ProcessEvent**

This method processes the event passed to it as an argument.

**Declaration**

```
// C#  
public override void ProcessEvent(WebBaseEvent eventRaised);
```

**Parameters**

- *eventRaised*

The `WebBaseEvent` object to be processed.

**Remarks**

This method is called by ASP.NET applications to start event processing. If buffering is enabled, then the event is added to the buffer of events, otherwise, the event information is directly written into Oracle Database.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleWebEventProvider Class](#)
- [OracleWebEventProvider Members](#)

**ProcessEventFlush**

This method flushes the information passed to it as an argument.

**Declaration**

```
// C#  
public override void ProcessEventFlush(WebEventBufferFlushInfo flushEvent);
```

**Parameters**

- *flushEvent*

The `WebEventBufferFlushInfo` object that contains a collection of buffered Web events.

**Remarks**

This method is called by ASP.NET applications to flush all events into Oracle Database.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleWebEventProvider Class](#)
- [OracleWebEventProvider Members](#)

**Shutdown**

This method releases all resources.

**Declaration**

```
// C#  
public override void Shutdown();
```

**Remarks**

This method is called by ASP.NET applications when the provider is unloaded. All the buffered events are first flushed into Oracle Database before the provider proceeds with shutdown.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleWebEventProvider Class](#)
- [OracleWebEventProvider Members](#)

---

---

# OraclePersonalizationProvider

This chapter describes the OraclePersonalizationProvider class.

**See Also:**

- ASP.NET Web Parts  
<http://msdn.microsoft.com/en-us/library/e0s9t4ck.aspx>
- ASP.NET Web Parts Personalization  
<http://msdn.microsoft.com/en-us/library/ms178182.aspx>

This chapter contains the following topic:

- [OraclePersonalizationProvider Class](#)

---

## OraclePersonalizationProvider Class

The `OraclePersonalizationProvider` class enables ASP.NET developers to store Web parts personalization information in an Oracle database.

### Class Inheritance

`System.Object`

`System.Configuration.Provider.ProviderBase`

`System.Web.UI.WebControls.WebParts.PersonalizationProvider`

`Oracle.Web.Personalization.OraclePersonalizationProvider`

### Declaration

```
// C#
public class OraclePersonalizationProvider: PersonalizationProvider
```

### Thread Safety

All public static methods are thread-safe, although instance members are not guaranteed to be thread-safe.

### Remarks

This class allows ASP.NET applications to store and manage personalization information in an Oracle database.

### Example

The following is a `web.config` example for an ASP.NET application that uses an `OraclePersonalizationProvider` as the default provider. This configuration uses the connection string and default attribute values specified in the `machine.config` file.

```
<?xml version="1.0"?>
<configuration xmlns="http://schemas.microsoft.com/.NetConfiguration/v2.0">
  <system.web>
    <webParts>
      <personalization defaultProvider="OraclePersonalizationProvider"/>
    </webParts>
  </system.web>
</configuration>
```

The following is a `web.config` example for an ASP.NET application that uses an `OraclePersonalizationProvider` as the default provider, with customized settings and an application-specific connection string:

```
<?xml version="1.0"?>
<configuration xmlns=
  "http://schemas.microsoft.com/.NetConfiguration/v2.0">
  <connectionStrings>
    <add name="my_personalization_app_con_string" connectionString=
      "User Id=scott;Password=tiger;Data Source=Oracle"/>
  </connectionStrings>
  <system.web>
    <webParts>
      <!-- Enable and customize OraclePersonalizationProvider -->
      <personalization defaultProvider="CustomOraclePersonalizationProvider">
```



```
<providers>
  <add name="CustomOraclePersonalizationProvider"
        type="Oracle.Web.Personalization.OraclePersonalizationProvider,
              Oracle.Web, Version=2.112.2.0, Culture=neutral,
              PublicKeyToken=89b483f429c47342"
        connectionStringName="my_personalization_app_con_string"
        applicationName="my_personalization_app" />
</providers>
</personalization>
</webParts>
</system.web>
</configuration>
```

Note that the `applicationName` attribute should be set to a unique value for each ASP.NET application.

### Requirements

Namespace: `Oracle.Web.Personalization`

Assembly: `Oracle.Web.dll`

Oracle Providers for ASP.NET Version: Oracle Providers for ASP.NET 2.0 and Oracle Providers for ASP.NET 4

#### See Also:

- ["Oracle Providers for ASP.NET Assembly" on page 1-4](#)
- [OraclePersonalizationProvider Members](#)
- [OraclePersonalizationProvider Constructors](#)
- [OraclePersonalizationProvider Static Methods](#)
- [OraclePersonalizationProvider Public Properties](#)
- [OraclePersonalizationProvider Public Methods](#)

## OraclePersonalizationProvider Members

OraclePersonalizationProvider members are listed in the following tables.

### OraclePersonalizationProvider Constructors

The OraclePersonalizationProvider constructor is listed in [Table 8-1](#).

**Table 8-1 OraclePersonalizationProvider Constructor**

Constructor	Description
<a href="#">OraclePersonalizationProvider Constructors</a>	Instantiates a new instance of the OraclePersonalizationProvider class

### OraclePersonalizationProvider Static Methods

OraclePersonalizationProvider static methods are listed in [Table 8-2](#).

**Table 8-2 OraclePersonalizationProvider Static Methods**

Static Methods	Description
Equals	Inherited from System.Object
ReferenceEquals	Inherited from System.Object

### OraclePersonalizationProvider Public Properties

OraclePersonalizationProvider public properties are listed in [Table 8-3](#).

**Table 8-3 OraclePersonalizationProvider Public Properties**

Public Properties	Description
<a href="#">ApplicationName</a>	Gets or sets the name of the application that is used to specify personalization information specific to an application
<a href="#">CommandTimeout</a>	Gets the number of seconds that the command is allowed to execute before it is terminated with an exception
Description	Inherited from System.Configuration.Provider.Providerbase
Name	Inherited from System.Configuration.Provider.Providerbase

### OraclePersonalizationProvider Public Methods

OraclePersonalizationProvider public methods are listed in [Table 8-4](#).

**Table 8-4 OraclePersonalizationProvider Public Methods**

Public Methods	Description
<a href="#">FindState</a>	Returns a collection containing zero or more PersonalizationStateInfo-derived objects based on scope and specific query parameters
<a href="#">GetCountOfState</a>	Returns the number of rows in the underlying Oracle database that are within the specified scope
<a href="#">Initialize</a>	Initializes the Oracle personalization provider
Equals	Inherited from System.Object (Overloaded)
ResetPersonalizationState	Inherited from System.Web.UI.WebControls.WebParts.PersonalizationProvider

**Table 8–4 (Cont.) OraclePersonalizationProvider Public Methods**

Public Methods	Description
<a href="#">ResetState</a>	Deletes personalization state information from the underlying data source, based on the specified parameters
<a href="#">ResetUserState</a>	Deletes user personalization data from the underlying Oracle data source, based on the specified parameters
GetHashCode	Inherited from System.Object
GetType	Inherited from System.Object
ToString	Inherited from System.Object
SavePersonalizationState	Inherited from System.Web.UI.WebControls.WebParts.PersonalizationProvider
DetermineInitialScope	Inherited from System.Web.UI.WebControls.WebParts.PersonalizationProvider
DetermineUserCapabilities	Inherited from System.Web.UI.WebControls.WebParts.PersonalizationProvider

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OraclePersonalizationProvider Class](#)

## OraclePersonalizationProvider Constructors

OraclePersonalizationProvider constructors create instances of the OraclePersonalizationProvider class.

### Overload List:

- [OraclePersonalizationProvider\(\)](#)

This constructor creates an instance of the OraclePersonalizationProvider class.

#### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OraclePersonalizationProvider Class](#)
- [OraclePersonalizationProvider Members](#)

### OraclePersonalizationProvider()

This constructor creates an instance of the OraclePersonalizationProvider class.

#### Declaration

```
// C#  
public OraclePersonalizationProvider();
```

#### Remarks

ASP.NET applications call this constructor to create an instance of the OraclePersonalizationProvider class as specified in the application configuration file. Initialization values for the OraclePersonalizationProvider instance are passed through the Initialize method.

#### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OraclePersonalizationProvider Class](#)
- [OraclePersonalizationProvider Members](#)

## OraclePersonalizationProvider Static Methods

The OraclePersonalizationProvider static methods are listed in [Table 8-5](#).

**Table 8-5 OraclePersonalizationProvider Static Methods**

Static Methods	Description
Equals	Inherited from System.Object
ReferenceEquals	Inherited from System.Object

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OraclePersonalizationProvider Class](#)
- [OraclePersonalizationProvider Members](#)

## OraclePersonalizationProvider Public Properties

The OraclePersonalizationProvider public properties are listed in [Table 8–6](#).

**Table 8–6 OraclePersonalizationProvider Public Properties**

Public Properties	Description
<a href="#">ApplicationName</a>	Gets or sets the name of the application that is used to specify personalization information specific to an application
<a href="#">CommandTimeout</a>	Gets the number of seconds that the command is allowed to execute before it is terminated with an exception
Description	Inherited from System.Configuration.Provider.Providerbase
Name	Inherited from System.Configuration.Provider.Providerbase

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OraclePersonalizationProvider Class](#)
- [OraclePersonalizationProvider Members](#)

### ApplicationName

This property gets or sets the name of the application that the personalization information is specific to.

**Declaration**

```
// C#
public override string ApplicationName{get; set;}
```

**Property Value**

The name of the application. If the applicationName attribute is not specified in the application configuration file, or if the value is an empty string, then this property is set to the application virtual path.

**Exceptions**

HttpException - The caller does not have high trust for ASP.NET hosting.

ProviderException - The ApplicationName string is greater than 256 characters.

**Remarks**

The main purpose of the ApplicationName property is to scope the data managed by OraclePersonalizationProvider object. Applications that specify the same ApplicationName string when configuring the Web parts personalization service share personalization state, but applications that specify unique ApplicationName strings do not. The OraclePersonalizationProvider must associate the personalization state with application names so operations performed on personalization data sources can be scoped accordingly.

The following example shows typical code that the OraclePersonalizationProvider might use to retrieve the personalization state for a user named *Scott* and an application named *App*:

```
SELECT * FROM PersonalizationState
```

```
WHERE UserName='Scott' AND Path='~/Default.aspx'  
AND ApplicationName='App'
```

The final AND in the WHERE clause ensures that other applications that contain personalization state keyed by the same user name and path do not conflict with the *App* application.

If no value is specified for the `applicationName` attribute in the configuration file, then the default is the `ApplicationPath` property value for the current request. The attribute name in the configuration file is case-sensitive.

The `ApplicationName` property is not thread-safe. It is recommended that application code not allow users to set the `ApplicationName` property in Web applications.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OraclePersonalizationProvider Class](#)
- [OraclePersonalizationProvider Members](#)

## CommandTimeout

This property gets the number of seconds that the command is allowed to execute before it is terminated with an exception.

**Declaration**

```
// C#  
public int CommandTimeout {get;}
```

**Property Value**

An int.

**Remarks**

To customize a provider, ASP.NET developers can set an integer value for this property through the `web.config` file using the `commandTimeout` attribute.

The default value is 30 seconds. The attribute name in the configuration file is case-sensitive.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OraclePersonalizationProvider Class](#)
- [OraclePersonalizationProvider Members](#)

## OraclePersonalizationProvider Public Methods

The OraclePersonalizationProvider public methods are listed in [Table 8-7](#).

**Table 8-7 OraclePersonalizationProvider Public Methods**

Public Methods	Description
<a href="#">FindState</a>	Returns a collection containing zero or more PersonalizationStateInfo-derived objects based on scope and specific query parameters
<a href="#">GetCountOfState</a>	Returns the number of rows in the underlying Oracle database that are within the specified scope
<a href="#">Initialize</a>	Initializes the Oracle personalization provider
Equals	Inherited from System.Object (Overloaded)
ResetPersonalizationState	Inherited from System.Web.UI.WebControls.WebParts.PersonalizationProvider
<a href="#">ResetState</a>	Deletes personalization state information from the underlying data source, based on the specified parameters
<a href="#">ResetUserState</a>	Deletes user personalization data from the underlying Oracle data source, based on the specified parameters
GetHashCode	Inherited from System.Object
GetType	Inherited from System.Object
ToString	Inherited from System.Object
SavePersonalizationState	Inherited from System.Web.UI.WebControls.WebParts.PersonalizationProvider
DetermineInitialScope	Inherited from System.Web.UI.WebControls.WebParts.PersonalizationProvider
DetermineUserCapabilities	Inherited from System.Web.UI.WebControls.WebParts.PersonalizationProvider

### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OraclePersonalizationProvider Class](#)
- [OraclePersonalizationProvider Members](#)

### FindState

This method returns a collection containing zero or more PersonalizationStateInfo-derived objects based on scope and specific query parameters.

#### Declaration

```
// C#
public override PersonalizationStateInfoCollection FindState(PersonalizationScope
    scope, PersonalizationStateQuery query, int pageIndex, int pageSize,
```



```
out int totalRecords);
```

### Parameters

- *scope*  
The scope of query (User or Shared) for personalization information. This cannot be a null reference.
- *query*  
The query to be used for filtering personalization information. This can be a null reference.
- *pageIndex*  
The location where the query starts.
- *pageSize*  
The number of records to return.
- *totalRecords*  
The total number of records available.

### Return Value

A `PersonalizationStateInfoCollection` object containing zero or more `PersonalizationStateInfo`-derived objects.

### Exceptions

`ArgumentOutOfRangeException` - The scope contains a value other than `PersonalizationScope.User` or `PersonalizationScope.Shared`.

`OracleException` - An Oracle-related error has occurred.

`ArgumentException` - One of the following conditions exists:

- The value of the *pageSize* parameter is 0 or 1.
- The *pageIndex* or *pageSize* parameter is less than 0.
- $((pageIndex * pageSize + pageSize) - 1)$  is greater than `Int32.MaxValue - 1` accounts for zero-based indexing of records.

### Remarks

The `PersonalizationStateInfo`-derived objects should be returned in alphabetic order and sorted by a combination of their `Path` and `Username` property values, both in ascending order.

This method passes the query wildcard characters to the underlying Oracle database. The database performs a wildcard search on a partial path with the wildcard character appearing at the beginning, the end, or the middle of the search string text in the `PathToMatch` property of the *query* parameter. For example, setting the `PathToMatch` property to `~/appdir%` finds all paths that start with `~/appdir`.

Likewise, in a wildcard search on a partial user name, the wildcard character can appear at any point in the text string of the `UsernameToMatch` property of the *query* parameter. For example, to find all user names that start with `scott`, the `UsernameToMatch` parameter looks like `scott%`.

The following query rules must be followed:

- If only the *scope* parameter is provided, and the *query* parameter is null or all the properties on the *query* parameter return either a null reference or default values, then all records matching the indicated *scope* parameter are returned.
- If the *PathToMatch* property is not a null reference, then the returned records are also filtered based on paths that match the *PathToMatch* value.
- If the *UsernameToMatch* property is not a null reference, then the returned records are also filtered based on user names that match the *UsernameToMatch* property value.
- If the *UserInactiveSinceDate* property is not equal to the *MaxValue*, then the records returned are also filtered to return only those records associated with inactive users. The comparison includes records where the *LastActivityDate* property is less than or equal to the *UserInactiveSinceDate* property.

This method does not validate combinations of query parameters. For example, the application code can request a set of personalization state records associated with a specific user name in the shared scope. Because user names are not associated with shared information, the returned collection is empty.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OraclePersonalizationProvider Class](#)
- [OraclePersonalizationProvider Members](#)

## GetCountOfState

This method returns the number of rows in the underlying Oracle database that are within the specified scope.

**Declaration**

```
// C#
public override int GetCountOfState(PersonalizationScope scope,
    PersonalizationStateQuery query);
```

**Parameters**

- *scope*  
The scope of query (User or Shared) for personalization information. This cannot be a null reference.
- *query*  
The query to be used for filtering personalization information. This can be a null reference.

**Return Value**

The number of rows in the underlying data source that are within the specified scope parameter.

**Exceptions**

*ArgumentException* -The *PathToMatch* or the *UsernameToMatch* property of *query* is a non-null reference and is an empty string ("") after trimming.

`ArgumentOutOfRangeException` - The scope specified is not a valid value from the `PersonalizationScope` enumeration.

`OracleException` - An Oracle-related error has occurred.

### Remarks

This method passes the query wildcard characters to the underlying Oracle database. The database performs a wildcard search on a partial path with the wildcard character appearing at the beginning, the end, or the middle of the search string text in the `PathToMatch` property of the `query` parameter. For example, setting the `PathToMatch` property to `~/appdir%` finds all paths that start with `~/appdir`.

Likewise, in a wildcard search on a partial user name, the wildcard character can appear at any point in the text string of the `UsernameToMatch` property of the `query` parameter. For example, to find all user names that start with `scott`, the `UsernameToMatch` parameter looks like `scott%`

The following query constraints must be followed:

- If only the `scope` parameter is provided, and the `query` parameter is a null reference or all the properties on the `query` parameter return either a null reference or default values, then all records matching the indicated `scope` parameter are returned.
- If the `PathToMatch` property is not a null reference, then the records returned are also filtered based on paths that match the `PathToMatch` value.
- If the `UsernameToMatch` property is not a null reference, then the returned records are also filtered based on user names that match the `UsernameToMatch` property value.
- If the `UserInactiveSinceDate` property is not equal to the `MaxValue`, then the returned records are also filtered to return only those records associated with inactive users. The comparison includes records where the `LastActivityDate` property is less than or equal to the `UserInactiveSinceDate` property.

#### See Also:

- ["Oracle Providers for ASP.NET Assembly" on page 1-4](#)
- [OraclePersonalizationProvider Class](#)
- [OraclePersonalizationProvider Members](#)

## Initialize

This method initializes the `OraclePersonalizationProvider` with the property values specified in the ASP.NET application configuration file (`web.config`).

### Declaration

```
// C#
public override void Initialize(string name, NameValueCollection config);
```

### Parameters

- `name`  
The friendly name of the provider.
- `config`  
A collection of the name/value pairs configuration options for this provider.

**Exceptions**

`HttpException` - The current trust level is less than Low.

`InvalidOperationException` - An attempt is made to call the `Initialize` method on a provider that has already been initialized.

`ArgumentNullException` - The `config` parameter is a null reference.

`System.Configuration.Provider.ProviderException` - One of the following conditions exists in the application configuration file:

- The `connectionStringName` attribute is empty or does not exist in the application configuration file.
- The value of the connection string for the `connectionStringName` attribute value is empty, or the specified `connectionStringName` attribute does not exist in the application configuration file.
- The `applicationName` attribute value exceeds 256 characters.
- The application configuration file for this `OraclePersonalizationProvider` instance contains an unrecognized attribute.

**Remarks**

The `Initialize` method is not intended to be called directly by the application.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OraclePersonalizationProvider Class](#)
- [OraclePersonalizationProvider Members](#)

**ResetState**

This method deletes personalization state information from the underlying data source, based on the specified parameters.

**Declaration**

```
// C#
public override int ResetState(PersonalizationScope scope, string[] paths,
    string[] usernames);
```

**Parameters**

- *scope*  
A `PersonalizationScope` type indicating the personalization information to be queried. This value cannot be a null reference.
- *paths*  
The paths for personalization information in the shared *scope* parameter to be deleted.
- *usernames*  
The user names for personalization information in the user *scope* parameter to be deleted.

**Return Value**

The number of rows deleted.

**Exceptions**

`ArgumentOutOfRangeException` - The *scope* parameter specified is not a member of the `PersonalizationScope` enumeration value.

`OracleException` - An Oracle-related error has occurred.

`ArgumentException` - Either of the following conditions exists:

- The *paths* or *usernames* parameter is an empty array.
- Elements of the *paths* or *usernames* arrays do not meet the validation rules. Validation rules are discussed in the following Remarks section.

**Remarks**

This method performs its operations as a single, atomic transaction.

Any *paths* and *usernames* elements contained within the respective arrays must meet the following validation rules. If a validation rule fails for any member of the parameter arrays, then an `ArgumentException` exception is thrown. The validation rules are:

- Null reference values are not allowed.
- An empty string ("" ) is not allowed. Parameters should be trimmed prior to performing an empty string check.
- The *usernames* array cannot contain a comma (,).

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OraclePersonalizationProvider Class](#)
- [OraclePersonalizationProvider Members](#)

**ResetUserState**

This method deletes user personalization data from the underlying Oracle data source, based on the specified parameters.

**Declaration**

```
// C#
public override int ResetUserState(string path, DateTime userInactiveSinceDate);
```

**Parameters**

- *path*

The path of the personalization data to be deleted. This value can be a null reference but cannot be an empty string ("" ).
- *userInactiveSinceDate*

The date that indicates the last activity.

**Return Value**

The count of rows deleted from the underlying Oracle data source.

**Exceptions**

`ArgumentException` - The *path* parameter is an empty string.

`OracleException` - An Oracle-related error has occurred.

**Remarks**

The parameters of this method have the following restrictions:

- The *path* parameter cannot contain wildcard characters.
- If the *path* parameter is a non-null reference, then only per-user personalization records associated with the *path* parameter are deleted.
- Only per-user personalization records associated with users that are considered inactive since the date specified in the *userInactiveSinceDate* parameter are deleted. The exact comparison deletes records where the Last Activity Date property is less than or equal to the *userInactiveSinceDate* parameter.
- If both parameters are provided, then records that match both constraints are deleted.
- The *path* parameter can be a null reference.
- The *path* parameter cannot be an empty string after trimming.
- The *userInactiveSinceDate* parameter cannot be a null reference.

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OraclePersonalizationProvider Class](#)
- [OraclePersonalizationProvider Members](#)

---

---

# OracleCacheDependency Provider

This chapter describes the `OracleCacheDependency` class.

**See Also:** ASP.NET `CacheDependency` class

<http://msdn.microsoft.com/en-us/library/system.web.caching.cachedependency.aspx>

This chapter contains the following topic:

- [OracleCacheDependency Class](#)

## OracleCacheDependency Class

The `OracleCacheDependency` object enables ASP.NET applications to invalidate cached items based on changes made in an Oracle database.

### Class Inheritance

`System.Object`

`System.Web.Caching.CacheDependency`

`Oracle.Web.Caching.OracleCacheDependency`

### Declaration

```
// C#  
public sealed class OracleCacheDependency : CacheDependency
```

### Thread Safety

All public static methods are thread-safe, although instance members are not guaranteed to be thread-safe.

### Remarks

This class invalidates data that is cached by ASP.NET applications, based on changes in the Oracle database.

This feature uses the Oracle Database Change Notification feature.

The user must have the `CHANGE NOTIFICATION` privilege, which can be granted with the following SQL statement:

```
GRANT change notification TO username;
```

### Requirements

Namespace: `Oracle.Web.Caching`

Assembly: `Oracle.Web.dll`

Oracle Providers for ASP.NET Version: Oracle Providers for ASP.NET 2.0 and Oracle Providers for ASP.NET 4

#### See Also:

- ["Oracle Providers for ASP.NET Assembly" on page 1-4](#)
- [OracleCacheDependency Members](#)
- [OracleCacheDependency Constructors](#)
- [OracleCacheDependency Properties](#)
- [OracleCacheDependency Methods](#)



## OracleCacheDependency Members

OracleCacheDependency members are listed in the following tables.

### OracleCacheDependency Constructors

The OracleCacheDependency constructor is listed in [Table 9-1](#).

**Table 9-1 OracleCacheDependency Constructor**

Constructor	Description
<a href="#">OracleCacheDependency Constructors</a>	Instantiates a new instance of the OracleCacheDependency class

### OracleCacheDependency Properties

OracleCacheDependency properties are listed in [Table 9-2](#).

**Table 9-2 OracleCacheDependency Properties**

Properties	Description
HasChanged	Inherited from System.CacheDependency
UtcLastModified	Inherited from System.CacheDependency

### OracleCacheDependency Methods

OracleCacheDependency methods are listed in [Table 9-3](#).

**Table 9-3 OracleCacheDependency Methods**

Methods	Description
Dispose	Inherited from System.Object
Equals	Inherited from System.Object (Overloaded)
GetHashCode	Inherited from System.Object
GetType	Inherited from System.Object
<a href="#">GetUniqueID</a>	Returns a unique identifier for the OracleCacheDependency object
ReferenceEquals	Inherited from System.Object
ToString	Inherited from System.Object

#### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleCacheDependency Class](#)

## OracleCacheDependency Constructors

This constructor instantiates a new instance of the `OracleCacheDependency` class.

### Overload List:

- [OracleCacheDependency\(OracleCommand\)](#)

This constructor creates an instance of the `OracleCacheDependency` class.

### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleCacheDependency Class](#)
- [OracleCacheDependency Members](#)

## OracleCacheDependency(OracleCommand)

This constructor instantiates a new instance of the `OracleCacheDependency` class.

### Declaration

```
// C#  
public OracleCacheDependency(OracleCommand cmd);
```

### Parameters

- *cmd*

The `OracleCommand` object has the command text on which the change notification is based.

### Remarks

When this constructor is invoked, the `OracleCacheDependency` object is instantiated and the `OracleCommand` object is configured for change notification. When the supplied `OracleCommand` object is executed by the application, the change notification is registered and the `OracleCacheDependency` instance is notified when changes are detected on the server side.

### See Also:

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleCacheDependency Class](#)
- [OracleCacheDependency Members](#)

## OracleCacheDependency Properties

OracleCacheDependency properties are listed in [Table 9-4](#).

**Table 9-4 OracleCacheDependency Properties**

Properties	Description
HasChanged	Inherited from System.CacheDependency
UtcLastModified	Inherited from System.CacheDependency

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleCacheDependency Class](#)
- [OracleCacheDependency Members](#)

## OracleCacheDependency Methods

OracleCacheDependency methods are listed in [Table 9–5](#).

**Table 9–5 OracleCacheDependency Methods**

Methods	Description
Dispose	Inherited from System.Object
Equals	Inherited from System.Object (Overloaded)
GetHashCode	Inherited from System.Object
GetType	Inherited from System.Object
<a href="#">GetUniqueID</a>	Returns a unique identifier for the OracleCacheDependency object
ReferenceEquals	Inherited from System.Object
ToString	Inherited from System.Object

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleCacheDependency Class](#)
- [OracleCacheDependency Members](#)

### GetUniqueID

This method returns a string that uniquely identifies the OracleCacheDependency object.

**Declaration**

```
// C#  
public override string GetUniqueID()
```

**See Also:**

- ["Oracle Providers for ASP.NET Assembly"](#) on page 1-4
- [OracleCacheDependency Class](#)
- [OracleCacheDependency Members](#)

## A

---

ASP.NET provider model, 1-1

## C

---

client configuration, 1-7  
configuration, 1-7  
configuration file, 1-1  
configuration scripts, 1-6

## D

---

documentation, 1-9

## F

---

file locations, 1-9

## I

---

install scripts, 1-6  
installation, 1-5  
    machine.config, 1-7  
InstallOracleASPNETCommon.sql, 1-6  
Instant Client, 1-5

## M

---

Microsoft ASP.NET 2.0, 1-1

## O

---

object references, 1-10  
objects, 1-10  
Oracle Data Provider for .NET  
    system requirements, 1-4  
Oracle Database, 1-1  
Oracle Providers for ASP.NET Assembly, 1-4  
Oracle Universal Installer, 1-5  
OracleCacheDependency Class  
    class description, 9-2  
    constructors, 9-4  
    members, 9-3  
    methods, 9-6  
    properties, 9-5  
OracleMembershipProvider Class

    class description, 2-2  
    constructors, 2-7  
    members, 2-4  
    public methods, 2-19, 2-38  
    public properties, 2-9  
    static methods, 2-8

### OraclePersonalizationProvider Class

    class description, 8-2  
    constructors, 8-6  
    members, 8-4  
    public methods, 8-10  
    public properties, 8-8  
    static methods, 8-7

### OracleProfileProvider Class

    class description, 6-2  
    constructors, 6-6  
    members, 6-4  
    public methods, 6-10  
    public properties, 6-8  
    static methods, 6-7

### OracleRoleProvider Class, 3-2

    class description, 3-2  
    constructors, 3-6  
    members, 3-4  
    public methods, 3-10  
    public properties, 3-8  
    static methods, 3-7

### OracleRoleProvider Member

    OracleRoleProvider Constructor, 7-5, 8-4

### OracleRoleProvider Members, 3-4

### OracleRoleProvider Public Properties, 7-5, 8-4

### OracleRoleProvider Static Methods, 7-5, 8-4

### OracleSessionStateStore Class

    class description, 5-2  
    constructors, 5-6  
    members, 5-4  
    public methods, 5-8  
    public properties, 5-7

### OracleSiteMapProvider Class

    class description, 4-2  
    constructors, 4-6  
    members, 4-4  
    public methods, 4-10  
    public properties, 4-8  
    static methods, 4-7

Oracle.Web.dll, 1-4, 1-9

## OracleWebEventProvider Class

- class description, 7-2
- constructors, 7-7
- members, 7-5
- public methods, 7-10
- public properties, 7-9
- static methods, 7-8

OraProvCfg, 1-7, 1-9

## **P**

---

passwords in code examples, x

privileges

- granting, 1-6

## **R**

---

roles, 1-11

## **S**

---

schema objects, 1-10

- roles, 1-11
- stored procedures, 1-13
- synonyms, 1-18
- tables, 1-10
- views, 1-12

SQL scripts, 1-9

stored procedures, 1-13

synonyms, 1-18

system requirements

- Oracle Data Provider for .NET, 1-4

## **T**

---

tables, 1-10

## **V**

---

views, 1-12

## **X**

---

xcopy Instant Client, 1-5