## Contents

### Preface
- Audience xii
- Documentation Accessibility xii
- Related Documents xii
- Passwords in Code Examples xiii
- Conventions xiii

### Changes in This Release for Oracle Providers for ASP.NET Developer's Guide
- Changes in Oracle Providers for ASP.NET in ODAC 12c Release 4 xiv
- Changes in Oracle Providers for ASP.NET Release 11.2.0.2 xiv
- Changes in Oracle Providers for ASP.NET Release 11.2.0.1.2 xv

### 1 Introduction to Oracle Providers for ASP.NET

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.4 Connecting to Oracle Database Cloud Service</td>
<td>1-1</td>
</tr>
<tr>
<td>1.1 Overview of Oracle Providers for ASP.NET</td>
<td>1-1</td>
</tr>
<tr>
<td>1.2 Oracle Providers for ASP.NET Assembly</td>
<td>1-4</td>
</tr>
<tr>
<td>1.3 System Requirements</td>
<td>1-5</td>
</tr>
<tr>
<td>1.5 Oracle Providers for ASP.NET Installation</td>
<td>1-5</td>
</tr>
<tr>
<td>1.5.1 Database Server Setup</td>
<td>1-8</td>
</tr>
<tr>
<td>1.5.1.1 Database Privileges for Setup</td>
<td>1-8</td>
</tr>
<tr>
<td>1.5.1.2 Configuring All Oracle Providers for ASP.NET</td>
<td>1-9</td>
</tr>
<tr>
<td>1.5.1.3 Configuring Oracle Providers for ASP.NET Individually</td>
<td>1-9</td>
</tr>
<tr>
<td>1.5.1.4 General Setup Information</td>
<td>1-10</td>
</tr>
<tr>
<td>1.5.2 ASP.NET Client Setup</td>
<td>1-10</td>
</tr>
<tr>
<td>1.6 Upgrading Oracle Providers for ASP.NET</td>
<td>1-11</td>
</tr>
<tr>
<td>1.6.1 Coexistence of Multiple Oracle Providers for ASP.NET Versions</td>
<td>1-12</td>
</tr>
<tr>
<td>1.7 File Locations After Installation</td>
<td>1-12</td>
</tr>
<tr>
<td>1.8 Oracle Providers for ASP.NET Object References</td>
<td>1-13</td>
</tr>
<tr>
<td>1.8.1 Tables</td>
<td>1-13</td>
</tr>
<tr>
<td>1.8.2 Roles</td>
<td>1-14</td>
</tr>
</tbody>
</table>
1.8.3 Views
1.8.3.1 OracleMembershipProvider Views
1.8.3.2 OracleRoleProvider Views
1.8.3.3 OracleProfileProvider Views
1.8.3.4 OraclePersonalizationProvider Views
1.8.3.5 OracleSessionStateStore Views
1.8.4 Stored Procedures
1.8.4.1 OracleMembershipProvider Stored Procedures
1.8.4.2 OracleRoleProvider Stored Procedures
1.8.4.3 OracleProfileProvider Stored Procedures
1.8.4.4 OraclePersonalizationProvider Stored Procedures
1.8.4.5 OracleWebEventProvider Stored Procedures
1.8.4.6 OracleSiteMapProvider Stored Procedures
1.8.4.7 OracleSessionStateStore Provider Stored Procedures
1.8.5 Synonyms

2 OracleMembershipProvider
2.1 OracleMembershipProvider Class
2.1.1 OracleMembershipProvider Members
2.1.2 OracleMembershipProvider Constructors
2.1.2.1 OracleMembershipProvider()
2.1.3 OracleMembershipProvider Static Methods
2.1.4 OracleMembershipProvider Public Properties
2.1.4.1 ApplicationName
2.1.4.2 CommandTimeout
2.1.4.3 EnablePasswordReset
2.1.4.4 EnablePasswordRetrieval
2.1.4.5 MaxInvalidPasswordAttempts
2.1.4.6 MinRequiredNonAlphanumericCharacters
2.1.4.7 MinRequiredPasswordLength
2.1.4.8 PasswordAttemptWindow
2.1.4.9 PasswordCompatMode
2.1.4.10 PasswordFormat
2.1.4.11 PasswordStrengthRegularExpression
2.1.4.12 RequiresQuestionAndAnswer
2.1.4.13 RequiresUniqueEmail
2.1.5 OracleMembershipProvider Public Methods
2.1.5.1 ChangePassword
2.1.5.2 ChangePasswordQuestionAndAnswer
2.1.5.3 CreateUser
2.1.5.4 DeleteUser 2-23
2.1.5.5 FindUsersByEmail 2-24
2.1.5.6 FindUsersByName 2-25
2.1.5.7 GeneratePassword 2-27
2.1.5.8 GetAllUsers 2-27
2.1.5.9 GetNumberOfUsersOnline 2-28
2.1.5.10 GetPassword 2-29
2.1.5.11 GetUser 2-30
2.1.5.12 GetUser(Object, bool) 2-31
2.1.5.13 GetUser(string, bool) 2-32
2.1.5.14 GetUserByNameByEmail 2-33
2.1.5.15 Initialize 2-33
2.1.5.16 ResetPassword 2-35
2.1.5.17 UnlockUser 2-36
2.1.5.18 UpdateUser 2-37
2.1.5.19 ValidateUser 2-38
2.1.6 OracleMembershipProvider Public Events 2-39

3 OracleRoleProvider

3.1 OracleRoleProvider Class 3-1
3.1.1 OracleRoleProvider Members 3-3
3.1.2 OracleRoleProvider Constructors 3-4
3.1.2.1 OracleRoleProvider() 3-5
3.1.3 OracleRoleProvider Static Methods 3-5
3.1.4 OracleRoleProvider Public Properties 3-6
3.1.4.1 ApplicationName 3-6
3.1.4.2 CommandTimeout 3-7
3.1.5 OracleRoleProvider Public Methods 3-8
3.1.5.1 AddUsersToRoles 3-9
3.1.5.2 CreateRole 3-9
3.1.5.3 DeleteRole 3-10
3.1.5.4 FindUsersInRole 3-11
3.1.5.5 GetAllRoles 3-12
3.1.5.6 GetRolesForUser 3-13
3.1.5.7 GetUsersInRole 3-13
3.1.5.8 Initialize 3-14
3.1.5.9 IsUserInRole 3-15
3.1.5.10 RemoveUsersFromRoles 3-16
3.1.5.11 RoleExists 3-17
<table>
<thead>
<tr>
<th>4</th>
<th><strong>OracleSiteMapProvider</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>4.1</td>
<td>OracleSiteMapProvider Class 4-1</td>
</tr>
<tr>
<td>4.1.1</td>
<td>OracleSiteMapProvider Members 4-3</td>
</tr>
<tr>
<td>4.1.2</td>
<td>OracleSiteMapProvider Constructors 4-5</td>
</tr>
<tr>
<td>4.1.2.1</td>
<td>OracleSiteMapProvider() 4-5</td>
</tr>
<tr>
<td>4.1.3</td>
<td>OracleSiteMapProvider Static Methods 4-6</td>
</tr>
<tr>
<td>4.1.4</td>
<td>OracleSiteMapProvider Public Properties 4-6</td>
</tr>
<tr>
<td>4.1.4.1</td>
<td>ApplicationName 4-7</td>
</tr>
<tr>
<td>4.1.4.2</td>
<td>CommandTimeout 4-8</td>
</tr>
<tr>
<td>4.1.5</td>
<td>OracleSiteMapProvider Public Methods 4-8</td>
</tr>
<tr>
<td>4.1.5.1</td>
<td>BuildSiteMap 4-9</td>
</tr>
<tr>
<td>4.1.5.2</td>
<td>Dispose 4-10</td>
</tr>
<tr>
<td>4.1.5.3</td>
<td>Initialize 4-11</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>5</th>
<th><strong>OracleSessionStateStore</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>5.1</td>
<td>OracleSessionStateStore Class 5-1</td>
</tr>
<tr>
<td>5.1.1</td>
<td>OracleSessionStateStore Members 5-2</td>
</tr>
<tr>
<td>5.1.2</td>
<td>OracleSessionStateStore Constructors 5-4</td>
</tr>
<tr>
<td>5.1.2.1</td>
<td>OracleSessionStateStore() 5-4</td>
</tr>
<tr>
<td>5.1.3</td>
<td>OracleSessionStateStore Public Properties 5-5</td>
</tr>
<tr>
<td>5.1.3.1</td>
<td>CommandTimeout 5-5</td>
</tr>
<tr>
<td>5.1.4</td>
<td>OracleSessionStateStore Public Methods 5-6</td>
</tr>
<tr>
<td>5.1.4.1</td>
<td>CreateNewStoreData 5-7</td>
</tr>
<tr>
<td>5.1.4.2</td>
<td>CreateUninitializedItem 5-7</td>
</tr>
<tr>
<td>5.1.4.3</td>
<td>Dispose 5-8</td>
</tr>
<tr>
<td>5.1.4.4</td>
<td>EndRequest 5-9</td>
</tr>
<tr>
<td>5.1.4.5</td>
<td>GetItem 5-9</td>
</tr>
<tr>
<td>5.1.4.6</td>
<td>GetItemExclusive 5-11</td>
</tr>
<tr>
<td>5.1.4.7</td>
<td>Initialize 5-12</td>
</tr>
<tr>
<td>5.1.4.8</td>
<td>InitializeRequest 5-13</td>
</tr>
<tr>
<td>5.1.4.9</td>
<td>ReleaseItemExclusive 5-13</td>
</tr>
<tr>
<td>5.1.4.10</td>
<td>RemoveItem 5-14</td>
</tr>
<tr>
<td>5.1.4.11</td>
<td>ResetItemTimeout 5-15</td>
</tr>
<tr>
<td>5.1.4.12</td>
<td>SetAndReleaseItemExclusive 5-16</td>
</tr>
<tr>
<td>5.1.4.13</td>
<td>SetItemExpireCallback 5-17</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>6</th>
<th><strong>OracleProfileProvider</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>6.1</td>
<td>OracleProfileProvider Class 6-1</td>
</tr>
<tr>
<td>6.1.1</td>
<td>OracleProfileProvider Members 6-3</td>
</tr>
</tbody>
</table>
6.1.2 OracleProfileProvider Constructors
   6.1.2.1 OracleProfileProvider() 6-5

6.1.3 OracleProfileProvider Static Methods 6-6

6.1.4 OracleProfileProvider Public Properties
   6.1.4.1 ApplicationName 6-7
   6.1.4.2 CommandTimeout 6-7

6.1.5 OracleProfileProvider Public Methods
   6.1.5.1 DeleteInactiveProfiles 6-9
   6.1.5.2 DeleteProfiles 6-10
   6.1.5.3 DeleteProfiles(ProfileInfoCollection) 6-10
   6.1.5.4 DeleteProfiles(string[]) 6-11
   6.1.5.5 FindInactiveProfilesByUserName 6-12
   6.1.5.6 FindProfilesByUserName 6-14
   6.1.5.7 GetAllInactiveProfiles 6-15
   6.1.5.8 GetAllProfiles 6-16
   6.1.5.9 GetNumberOfInactiveProfiles 6-17
   6.1.5.10 GetPropertyValues 6-18
   6.1.5.11 Initialize 6-19
   6.1.5.12 SetPropertyValues 6-20

7 OracleWebEventProvider
   7.1 OracleWebEventProvider Class 7-1
   7.1.1 OracleWebEventProvider Members 7-3
   7.1.2 OracleWebEventProvider Constructors
   7.1.2.1 OracleWebEventProvider() 7-5
   7.1.3 OracleWebEventProvider Static Methods 7-6
   7.1.4 OracleWebEventProvider Public Properties
   7.1.4.1 CommandTimeout 7-7
   7.1.5 OracleWebEventProvider Public Methods
   7.1.5.1 Initialize 7-8
   7.1.5.2 ProcessEvent 7-9
   7.1.5.3 ProcessEventFlush 7-9
   7.1.5.4 Shutdown 7-10

8 OraclePersonalizationProvider
   8.1 OraclePersonalizationProvider Class 8-1
   8.1.1 OraclePersonalizationProvider Members 8-3
   8.1.2 OraclePersonalizationProvider Constructors
   8.1.2.1 OraclePersonalizationProvider() 8-5
8.1.3 OraclePersonalizationProvider Static Methods 8-5
8.1.4 OraclePersonalizationProvider Public Properties 8-6
  8.1.4.1 ApplicationName 8-6
  8.1.4.2 CommandTimeout 8-8
8.1.5 OraclePersonalizationProvider Public Methods 8-8
  8.1.5.1 FindState 8-9
  8.1.5.2 GetCountOfState 8-11
  8.1.5.3 Initialize 8-12
  8.1.5.4 ResetState 8-13
  8.1.5.5 ResetUserState 8-14

9 OracleCacheDependency Provider
  9.1 OracleCacheDependency Class 9-1
    9.1.1 OracleCacheDependency Members 9-2
    9.1.2 OracleCacheDependency Constructors 9-3
      9.1.2.1 OracleCacheDependency(OracleCommand) 9-3
    9.1.3 OracleCacheDependency Properties 9-4
    9.1.4 OracleCacheDependency Methods 9-4
      9.1.4.1 GetUniqueID 9-5

Index
List of Tables

1-1 Oracle Providers for ASP.NET Namespaces and Providers 1-4
1-2 Install Scripts 1-9
1-3 Provider Tables 1-13
1-4 Roles and Privileges 1-14
1-5 OracleMembershipProvider 1-15
1-6 OracleRoleProvider 1-15
1-7 OracleProfileProvider 1-16
1-8 OraclePersonalizationProvider 1-16
1-9 OracleSessionStateStore 1-16
1-10 ora_aspnet_Mem_BasicAccess Role 1-17
1-11 ora_aspnet_Mem_ReportAccess Role 1-17
1-12 ora_aspnet_Mem_FullAccess Role 1-17
1-13 ora_aspnet_Roles_BasicAccess Role 1-18
1-14 ora_aspnet_Roles_ReportAccess Role 1-18
1-15 ora_aspnet_Roles_FullAccess Role 1-19
1-16 ora_aspnet_Prof_BasicAccess Role 1-19
1-17 ora_aspnet_Prof_ReportAccess Role 1-19
1-18 ora_aspnet_Prof_FullAccess Role 1-20
1-19 ora_aspnet_Pers_BasicAccess Role 1-20
1-20 ora_aspnet_Pers_ReportAccess Role 1-20
1-21 ora_aspnet_Pers_FullAccess Role 1-20
1-22 ora_aspnet_Wevnt_FullAccess Role 1-21
1-23 ora_aspnet_Smap_FullAccess Role 1-21
1-24 ora_aspnet_Sessn_FullAccess Role 1-21
2-1 OracleMembershipProvider Constructor 2-3
2-2 OracleMembershipProvider Static Methods 2-3
2-3 OracleMembershipProvider Public Properties 2-3
2-4 OracleMembershipProvider Public Methods 2-4
2-5 OracleMembershipProvider Public Event 2-5
2-6 OracleMembershipProvider Static Methods 2-7
2-7 OracleMembershipProvider Public Properties 2-7
2-8 OracleMembershipProvider Public Methods 2-18
2-9 MembershipCreateStatus Enumeration Values 2-23
2-10 OracleMembershipProvider Public Events 2-39
3-1 OracleRoleProvider Constructor 3-3
<table>
<thead>
<tr>
<th>Page</th>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>8-6</td>
<td>OraclePersonalizationProvider Public Properties</td>
<td>8-6</td>
</tr>
<tr>
<td>8-7</td>
<td>OraclePersonalizationProvider Public Methods</td>
<td>8-8</td>
</tr>
<tr>
<td>9-1</td>
<td>OracleCacheDependency Constructor</td>
<td>9-2</td>
</tr>
<tr>
<td>9-2</td>
<td>OracleCacheDependency Properties</td>
<td>9-2</td>
</tr>
<tr>
<td>9-3</td>
<td>OracleCacheDependency Methods</td>
<td>9-3</td>
</tr>
<tr>
<td>9-4</td>
<td>OracleCacheDependency Properties</td>
<td>9-4</td>
</tr>
<tr>
<td>9-5</td>
<td>OracleCacheDependency Methods</td>
<td>9-5</td>
</tr>
</tbody>
</table>
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


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](http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info) or visit [http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs](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*
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:

<table>
<thead>
<tr>
<th>Convention</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>boldface</strong></td>
<td>Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.</td>
</tr>
<tr>
<td><em>italic</em></td>
<td>Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.</td>
</tr>
<tr>
<td>monospace</td>
<td>Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.</td>
</tr>
</tbody>
</table>
Changes in This Release for Oracle Providers for ASP.NET Developer's Guide

This preface contains:

- Changes in Oracle Providers for ASP.NET in ODAC 12c Release 4
- 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 in ODAC 12c Release 4

The following are changes in Oracle Providers for ASP.NET for ODAC 12c Release 4.

New Features

The following features are new in this release:

- NuGet
  Oracle Providers for ASP.NET are available in a NuGet package. This capability simplifies distributing these providers to developers and end users.

- Windows Installer
  Oracle Providers for ASP.NET are now available as part of an ODAC Microsoft Windows Installer (MSI) package.

- ODP.NET, Managed Driver Support
  Oracle Providers for ASP.NET for .NET Framework 4 and higher depend on ODP.NET, Managed Driver (Oracle.ManagedDataAccess.dll), rather than ODP.NET, Unmanaged Driver (Oracle.DataAccess.dll) as in previous releases. The providers’ DLL version is 4.121.2.1, which distinguishes it from its predecessor version 4.121.2.0, which in ODAC 12c Release 3 depended on unmanaged ODP.NET. In addition, Oracle Providers for ASP.NET for .NET Framework 4 and higher is now a platform independent assembly, enabling platform-independent applications to simply use the same .NET assembly, regardless of the platform.

Oracle Providers for ASP.NET for .NET Framework 2.0 is platform dependent and depends on ODP.NET, Unmanaged Driver. Its version number remains the same as 2.121.2.0.

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."

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.
1

Introduction to Oracle Providers for ASP.NET

Oracle Providers for ASP.NET integrates directly with Microsoft ASP.NET controls and services to provide state management capabilities for web sites.

The following topics introduce Oracle Providers for ASP.NET.

- Overview of Oracle Providers for ASP.NET
- Oracle Providers for ASP.NET Assembly
- System Requirements
- Connecting to Oracle Database Cloud Service
- Oracle Providers for ASP.NET Installation
- Upgrading Oracle Providers for ASP.NET
- File Locations After Installation
- Oracle Providers for ASP.NET Object References

1.4 Connecting to Oracle Database Cloud Service

Oracle Providers for ASP.NET support connectivity with Oracle Database Cloud Services, including Oracle Autonomous Database.

Set-up Instructions

Oracle recommends using the latest Oracle Providers for ASP.NET version available when connecting to Oracle Database Cloud Services. You will find instructions about how to download, install, and configure Oracle Providers for ASP.NET below.

Oracle Autonomous Database: https://www.oracle.com/technetwork/topics/dotnet/tech-info/default-5032178.html

Oracle Database as a Service: https://www.oracle.com/technetwork/topics/dotnet/dotnetdbaas-3208838.html

1.1 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 lo-
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.

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.

- **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.

- **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.

- **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.

- **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.

- **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.

- **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.

- **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

The Microsoft Developer Network for more information about:

- ASP.NET Web Parts [http://msdn.microsoft.com/en-us/library/e0s9t4ck.aspx](http://msdn.microsoft.com/en-us/library/e0s9t4ck.aspx)

### 1.2 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**

<table>
<thead>
<tr>
<th>Provider Type</th>
<th>Class Name</th>
<th>Namespace</th>
</tr>
</thead>
<tbody>
<tr>
<td>Membership</td>
<td>OracleMembershipProviderClass</td>
<td>Oracle.Web.Security</td>
</tr>
<tr>
<td>Role</td>
<td>OracleRoleProviderClass</td>
<td>Oracle.Web.Security</td>
</tr>
<tr>
<td>Site Map</td>
<td>OracleSiteMapProviderClass</td>
<td>Oracle.Web.SiteMap</td>
</tr>
<tr>
<td>Session State</td>
<td>OracleSessionStateStoreClass</td>
<td>Oracle.Web.SessionState</td>
</tr>
<tr>
<td>Profile</td>
<td>OracleProfileProviderClass</td>
<td>Oracle.Web.Profile</td>
</tr>
</tbody>
</table>
Table 1-1 (Cont.) Oracle Providers for ASP.NET Namespaces and Providers

<table>
<thead>
<tr>
<th>Provider Type</th>
<th>Class Name</th>
<th>Namespace</th>
</tr>
</thead>
<tbody>
<tr>
<td>Personalization</td>
<td>OraclePersonalizationProvider Class</td>
<td>Oracle.Web.Personalization</td>
</tr>
<tr>
<td>Cache Dependency</td>
<td>OracleCacheDependency Class</td>
<td>Oracle.Web.Caching</td>
</tr>
</tbody>
</table>

1.3 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 4.5.2 and higher

- Windows operating system
  - x64: Windows 8.1 (Pro and Enterprise Editions), Windows 10 x64 (Pro, Enterprise, and Education Editions), Windows Server 2012 R2 x64 (Standard, Datacenter, Essentials, and Foundation Editions), Windows Server 2016 x64 (Standard, Datacenter, and Essentials Editions), or Windows Server 2019 x64 (Standard, Datacenter, and Essentials 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 11g Release 2 or later.

- Either one of the following:
  - Oracle Data Provider for .NET, Managed Driver
  - Oracle Data Provider for .NET, Unmanaged Driver and Oracle Client

Both drivers are installed with Oracle Providers for ASP.NET software. You must use the same versions of ODP.NET with Oracle Providers for ASP.NET. For example, if you use Oracle Providers for ASP.NET version 19, you should use ODP.NET and Oracle Client versions 19 as well.

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

1.5 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.

- NuGet
NuGet is software packaging method for Microsoft .NET. It provides an automated install and setup method popular with developers and administrators that now can be used to install Oracle Providers for ASP.NET.

- **Microsoft Windows Installer**
  
  Microsoft Windows Installer (MSI) is the popular, standard installation method for Windows-based software. It can be used to deploy Oracle Providers for ASP.NET.

- **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.

### Machine-Wide Configuration

Machine-wide configuration was once available as an install option in Oracle Universal Installer and XCopy. It is no longer available in current ODAC and Oracle Client installations. If machine-wide Oracle Providers for ASP.NET configuration was chosen in a previous version, that particular ASP.NET providers installation had:

- updated the `machine.config` to use an older `Oracle.Web.dll` version,
- placed that older `Oracle.Web.dll` into the `GAC_MSIL`,
- placed policy DLLs into the `GAC_MSIL` that redirects AnyCPU application to use the older `Oracle.Web.dll`.

Thus, existing applications will continue to use these older Oracle ASP.NET providers unless these changes are reversed through deinstalling the old providers or manual configuration steps, such as what is described in the next section.

Machine-wide configuration changes occurred for both Oracle Providers for ASP.NET 2.0 and Oracle Providers for ASP.NET 4.

### Non-Machine-Wide Configuration

Non-machine-wide Oracle Providers for ASP.NET configuration is the only install option now available. This install option will:

- **not update the** `machine.config` **to use** `Oracle.Web.dll` **version,**
- **not place** `Oracle.Web.dll` **into the** `GAC_MSIL`,
- **not place policy DLLs into the** `GAC_MSIL` **that redirects AnyCPU application to use the new** `Oracle.Web.dll`.

Unless some manual changes are made to existing applications, they will continue to use their current Oracle Providers for ASP.NET assembly version if configured machine-wide, not the newly installed version.

The following manual steps are required in order for your application to use the newly installed `Oracle.Web.dll` if machine-wide configuration is not chosen.

The `Oracle.Web.dll` and either `Oracle.ManagedDataAccess.dll` or `Oracle.DataAccess.dll` will need to be copied to your application directory for your application to use
the newer version of Oracle.Web.dll since it was chosen not to be placed into the GAC.

Proper configuration is needed in the web.config to refer to the new Oracle.Web.dll. Here’s an example of what the web.config will look if it’s desired to have all the providers configured to use Oracle.Web.dll version 4.122.19.1:

```xml
<configuration>
<system.web>
<membership>
<providers>
  <add name="OracleMembershipProvider" type="Oracle.Web.Security.OracleMembershipProvider, Oracle.Web, Version=4.122.19.1, Culture=neutral, PublicKeyToken=89b483f429c47342" connectionStringName="OraAspNetConString" applicationName="" enablePasswordRetrieval="false" enablePasswordReset="true" requiresQuestionAndAnswer="true" requiresUniqueEmail="false" passwordFormat="Hashed" maxInvalidPasswordAttempts="5" minRequiredPasswordLength="7" minRequiredNonalphanumericCharacters="1" passwordAttemptWindow="10" passwordStrengthRegularExpression="" />
</providers>
</membership>
<profile>
<providers>
  <add name="OracleProfileProvider" type="Oracle.Web.Profile.OracleProfileProvider, Oracle.Web, Version=4.122.19.1, Culture=neutral, PublicKeyToken=89b483f429c47342" connectionStringName="OraAspNetConString" applicationName="" />
</providers>
</profile>
<roleManager>
<providers>
  <add name="OracleRoleProvider" type="Oracle.Web.Security.OracleRoleProvider, Oracle.Web, Version=4.122.19.1, Culture=neutral, PublicKeyToken=89b483f429c47342" connectionStringName="OraAspNetConString" applicationName="" />
</providers>
</roleManager>
<siteMap>
<providers>
  <add name="OracleSiteMapProvider" type="Oracle.Web.SiteMap.OracleSiteMapProvider, Oracle.Web, Version=4.122.19.1, Culture=neutral, PublicKeyToken=89b483f429c47342" connectionStringName="OraAspNetConString" applicationName="" securityTrimmingEnabled="true" />
</providers>
</siteMap>
</system.web>
</configuration>
```
</providers>
</healthMonitoring>
</system.web>
</configuration>

Please note that the application can be updated to use the newer Oracle.Web.dll version and the web.config can be configured automatically by installing the NuGet package for Oracle Providers for ASP.NET.

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:
- "ASP.NET Client Setup" for more details
- Oracle Database Installation Guide for Microsoft Windows for installation instructions

1.5.1 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

1.5.1.1 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.

1.5.1.2 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.

1.5.1.3 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

<table>
<thead>
<tr>
<th>Provider</th>
<th>Required Installation Script</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle Membership Provider</td>
<td>InstallOracleMembership.sql</td>
</tr>
<tr>
<td>Oracle Personalization Provider</td>
<td>InstallOraclePersonalization.sql</td>
</tr>
</tbody>
</table>
### Table 1-2 (Cont.) Install Scripts

<table>
<thead>
<tr>
<th>Provider</th>
<th>Required Installation Script</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oracle Profile Provider</td>
<td>InstallOracleProfile.sql</td>
</tr>
<tr>
<td>Oracle Role Provider</td>
<td>InstallOracleRoles.sql</td>
</tr>
<tr>
<td>Oracle Session State Provider</td>
<td>InstallOracleSessionState.sql</td>
</tr>
<tr>
<td>Note: This provider requires the execution of only the InstallOracleSessionState.sql script. It does not require the execution of InstallOracleASPNETCommon.sql.</td>
<td></td>
</tr>
<tr>
<td>Oracle Site Map Provider</td>
<td>InstallOracleSiteMap.sql</td>
</tr>
<tr>
<td>Oracle Web Events Provider</td>
<td>InstallOracleWebEvents.sql</td>
</tr>
<tr>
<td>Oracle Cache Dependency Provider</td>
<td>No script execution needed</td>
</tr>
</tbody>
</table>

#### 1.5.1.4 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.

#### 1.5.2 ASP.NET Client Setup

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.

```xml
<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 machine-wide configuration 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`. 
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 machine-wide configuration 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.

### 1.6 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. The database version itself does not need to be upgraded, only the ASP.NET provider schema does. 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 no longer automatically updated by default so that Web appli-
cations are directed to use the recently installed version. If there are applications
that use the newly installed Oracle ASP.NET providers, then administrators must
update the appropriate .NET configuration 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 pre‐
  served when performing the upgrade. Encrypted data remains encrypted and usable
  after the upgrade.

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

If there are multiple ASP.NET applications using a single Web server or a single Ora‐
cle 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 Provid‐
ers 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 ver‐
sion 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 pro‐
vider 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 data‐
base, administrators can create separate schemas for each Oracle Providers for
ASP.NET version.

### 1.7 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.

- Configuration utility OraProvCfg.exe (For .NET Framework 4)
  \ORACLE_BASE\ORACLE_HOME\ASP.NET\Bin\4 directory.

- Configuration (SQL) scripts
  \ORACLE_BASE\\ORACLE_HOME\ASP.NET\SQL directory.

- Dynamic Help file
  \ORACLE_BASE\\ORACLE_HOME\ASP.NET\Help directory
1.8 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 tabulated 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

### 1.8.1 Tables

Table 1-3 lists the tables that are used by each provider.

<table>
<thead>
<tr>
<th>Oracle Provider</th>
<th>Table</th>
</tr>
</thead>
<tbody>
<tr>
<td>Membership</td>
<td>ora_aspnet_Membership</td>
</tr>
<tr>
<td></td>
<td>ora_aspnet_Applications</td>
</tr>
<tr>
<td></td>
<td>ora_aspnet_Users</td>
</tr>
<tr>
<td>Role</td>
<td>ora_aspnet_Roles</td>
</tr>
<tr>
<td></td>
<td>ora_aspnet_UsersInRoles</td>
</tr>
<tr>
<td></td>
<td>ora_aspnet_Applications</td>
</tr>
<tr>
<td></td>
<td>ora_aspnet_Users</td>
</tr>
<tr>
<td>Profile</td>
<td>ora_aspnet_Profile</td>
</tr>
<tr>
<td></td>
<td>ora_aspnet_Applications</td>
</tr>
<tr>
<td></td>
<td>ora_aspnet_Users</td>
</tr>
</tbody>
</table>
Table 1-3  (Cont.) Provider Tables

<table>
<thead>
<tr>
<th>Oracle Provider</th>
<th>Table</th>
</tr>
</thead>
<tbody>
<tr>
<td>Personalization</td>
<td>ora_aspnet_Paths</td>
</tr>
<tr>
<td></td>
<td>ora_aspnet_PersonaliznAllUsers</td>
</tr>
<tr>
<td></td>
<td>ora_aspnet_PersonaliznPerUser</td>
</tr>
<tr>
<td></td>
<td>ora_aspnet_Applications</td>
</tr>
<tr>
<td></td>
<td>ora_aspnet_Users</td>
</tr>
<tr>
<td>Web Events</td>
<td>ora_aspnet_WebEvents</td>
</tr>
<tr>
<td>Site Map</td>
<td>ora_aspnet_SiteMap</td>
</tr>
<tr>
<td></td>
<td>ora_aspnet_Applications</td>
</tr>
<tr>
<td>Session State</td>
<td>ora_aspnet_SessionApplications</td>
</tr>
<tr>
<td></td>
<td>ora_aspnet_Sessions</td>
</tr>
</tbody>
</table>

1.8.2 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

<table>
<thead>
<tr>
<th>Oracle Provider</th>
<th>Oracle Database Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Membership</td>
<td>ora_aspnet_Mem_BasicAccess</td>
</tr>
<tr>
<td></td>
<td>ora_aspnet_Mem_ReportAccess</td>
</tr>
<tr>
<td></td>
<td>ora_aspnet_Mem_FullAccess</td>
</tr>
<tr>
<td>Role</td>
<td>ora_aspnet_Roles_BasicAccess</td>
</tr>
<tr>
<td></td>
<td>ora_aspnet_Roles_ReportAccess</td>
</tr>
<tr>
<td></td>
<td>ora_aspnet_Roles_FullAccess</td>
</tr>
<tr>
<td>Profile</td>
<td>ora_aspnet_Prof_BasicAccess</td>
</tr>
<tr>
<td></td>
<td>ora_aspnet_Prof_ReportAccess</td>
</tr>
<tr>
<td></td>
<td>ora_aspnet_Prof_FullAccess</td>
</tr>
<tr>
<td>Personalization</td>
<td>ora_aspnet_Pers_BasicAccess</td>
</tr>
<tr>
<td></td>
<td>ora_aspnet_Pers_ReportAccess</td>
</tr>
<tr>
<td></td>
<td>ora_aspnet_Pers_FullAccess</td>
</tr>
<tr>
<td>Web Events</td>
<td>ora_aspnet_Wevnt_FullAccess</td>
</tr>
<tr>
<td>Site Map</td>
<td>ora_aspnet_Smap_FullAccess</td>
</tr>
</tbody>
</table>
1.8.3 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.

1.8.3.1 OracleMembershipProvider Views

Table 1-5 lists the roles and the view access that the roles provide.

Table 1-5  OracleMembershipProvider

<table>
<thead>
<tr>
<th>Role</th>
<th>View</th>
</tr>
</thead>
<tbody>
<tr>
<td>ora_aspnet_Mem_BasicAccess</td>
<td>(none)</td>
</tr>
<tr>
<td>ora_aspnet_Mem_ReportAccess</td>
<td>ora_vw_aspnet_Applications</td>
</tr>
<tr>
<td></td>
<td>ora_vw_aspnet_Users</td>
</tr>
<tr>
<td></td>
<td>ora_vw_aspnet_MemUsers</td>
</tr>
<tr>
<td>ora_aspnet_Mem_FullAccess</td>
<td>ora_vw_aspnet_Applications</td>
</tr>
<tr>
<td></td>
<td>ora_vw_aspnet_Users</td>
</tr>
<tr>
<td></td>
<td>ora_vw_aspnet_MemUsers</td>
</tr>
</tbody>
</table>

1.8.3.2 OracleRoleProvider Views

Table 1-6 lists the roles and the view access that the roles provide.

Table 1-6  OracleRoleProvider

<table>
<thead>
<tr>
<th>Role</th>
<th>View</th>
</tr>
</thead>
<tbody>
<tr>
<td>ora_aspnet_Roles_BasicAccess</td>
<td>(none)</td>
</tr>
<tr>
<td>ora_aspnet_Roles_ReportAccess</td>
<td>ora_vw_aspnet_Applications</td>
</tr>
<tr>
<td></td>
<td>ora_vw_aspnet_Users</td>
</tr>
<tr>
<td></td>
<td>ora_vw_aspnet_Roles</td>
</tr>
<tr>
<td></td>
<td>ora_vw_aspnet_UIR</td>
</tr>
<tr>
<td>ora_aspnet_Roles_FullAccess</td>
<td>ora_vw_aspnet_Applications</td>
</tr>
<tr>
<td></td>
<td>ora_vw_aspnet_Users</td>
</tr>
<tr>
<td></td>
<td>ora_vw_aspnet_Roles</td>
</tr>
<tr>
<td></td>
<td>ora_vw_aspnet_UIR</td>
</tr>
</tbody>
</table>
1.8.3.3 OracleProfileProvider Views

Table 1-7 lists the roles and the view access that the roles provide.

<table>
<thead>
<tr>
<th>Role</th>
<th>View</th>
</tr>
</thead>
<tbody>
<tr>
<td>ora_aspnet_Prof_BasicAccess</td>
<td>(none)</td>
</tr>
<tr>
<td>ora_aspnet_Prof_ReportAccess</td>
<td>ora_vw_aspnet_Applications</td>
</tr>
<tr>
<td></td>
<td>ora_vw_aspnet_Users</td>
</tr>
<tr>
<td></td>
<td>ora_vw_aspnet_Profiles</td>
</tr>
<tr>
<td>ora_aspnet_Prof_FullAccess</td>
<td>ora_vw_aspnet_Applications</td>
</tr>
<tr>
<td></td>
<td>ora_vw_aspnet_Users</td>
</tr>
<tr>
<td></td>
<td>ora_vw_aspnet_Profiles</td>
</tr>
</tbody>
</table>

1.8.3.4 OraclePersonalizationProvider Views

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

<table>
<thead>
<tr>
<th>Role</th>
<th>View</th>
</tr>
</thead>
<tbody>
<tr>
<td>ora_aspnet_Pers_BasicAccess</td>
<td>(none)</td>
</tr>
<tr>
<td>ora_aspnet_Pers_ReportAccess</td>
<td>ora_vw_aspnet_Applications</td>
</tr>
<tr>
<td></td>
<td>ora_vw_aspnet_Users</td>
</tr>
<tr>
<td></td>
<td>ora_vw_aspnet_Users</td>
</tr>
<tr>
<td>ora_aspnet_Pers_FullAccess</td>
<td>ora_vw_aspnet_Applications</td>
</tr>
<tr>
<td></td>
<td>ora_vw_aspnet_Users</td>
</tr>
</tbody>
</table>

1.8.3.5 OracleSessionStateStore Views

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

<table>
<thead>
<tr>
<th>Role</th>
<th>View</th>
</tr>
</thead>
<tbody>
<tr>
<td>ora_aspnet_Sessn_FullAccess</td>
<td>ora_vw_aspnet_sessions</td>
</tr>
</tbody>
</table>

1.8.4 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.
1.8.4.1 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

<table>
<thead>
<tr>
<th>Service Method</th>
<th>Stored Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetNumberOfUsersOnline</td>
<td>ora_aspnet_Mem_GetNumOfUsersOn</td>
</tr>
<tr>
<td>GetPassword</td>
<td>ora_aspnet_Mem_GetPassword</td>
</tr>
<tr>
<td>GetUser</td>
<td>ora_aspnet_Mem_GetUserByUid</td>
</tr>
<tr>
<td></td>
<td>ora_aspnet_Mem_GetUserByName</td>
</tr>
<tr>
<td>GetUserByEmail</td>
<td>ora_aspnet_Mem_GetUserByEml</td>
</tr>
<tr>
<td>UpdateUser</td>
<td>ora_aspnet_Mem_UpdateUser</td>
</tr>
<tr>
<td>ValidateUser</td>
<td>ora_aspnet_Mem_GetPwdWithFmt</td>
</tr>
<tr>
<td></td>
<td>ora_aspnet_Mem_UpdateUserInfo</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Service Method</th>
<th>Stored Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>FindUsersByEmail</td>
<td>ora_aspnet_Mem_FindUsersByEml</td>
</tr>
<tr>
<td>FindUsersByName</td>
<td>ora_aspnet_Mem_FindUsersByName</td>
</tr>
<tr>
<td>GetAllUsers</td>
<td>ora_aspnet_Mem_GetAllUsers</td>
</tr>
<tr>
<td>GetNumberOfUsersOnline</td>
<td>ora_aspnet_Mem_GetNumOfUsersOn</td>
</tr>
<tr>
<td>GetUser</td>
<td>ora_aspnet_Mem_GetUserByUid</td>
</tr>
<tr>
<td></td>
<td>ora_aspnet_Mem_GetUserByName</td>
</tr>
<tr>
<td>GetUserByEmail</td>
<td>ora_aspnet_Mem_GetUserByEml</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Service Method</th>
<th>Stored Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>All Membership methods</td>
<td>ora_aspnet_Mem_ChangePwdQAndA</td>
</tr>
<tr>
<td>All Membership methods</td>
<td>ora_aspnet_Mem_CreateUser</td>
</tr>
<tr>
<td>All Membership methods</td>
<td>ora_aspnet_Mem_FindUsersByEml</td>
</tr>
<tr>
<td>All Membership methods</td>
<td>ora_aspnet_Mem_FindUsersByName</td>
</tr>
<tr>
<td>All Membership methods</td>
<td>ora_aspnet_Mem_GetAllUsers</td>
</tr>
<tr>
<td>All Membership methods</td>
<td>ora_aspnet_Mem_GetNumOfUsersOn</td>
</tr>
</tbody>
</table>
### Table 1-12 (Cont.) ora_aspnet_Mem_FullAccess Role

<table>
<thead>
<tr>
<th>Service Method</th>
<th>Stored Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>All Membership methods</td>
<td>ora_aspnet_Mem_GetPassword</td>
</tr>
<tr>
<td>All Membership methods</td>
<td>ora_aspnet_Mem_GetPwdWithFmt</td>
</tr>
<tr>
<td>All Membership methods</td>
<td>ora_aspnet_Mem_GetUserByEml</td>
</tr>
<tr>
<td>All Membership methods</td>
<td>ora_aspnet_Mem_GetUserByName</td>
</tr>
<tr>
<td>All Membership methods</td>
<td>ora_aspnet_Mem_GetUserByUid</td>
</tr>
<tr>
<td>All Membership methods</td>
<td>ora_aspnet_Mem_ResetPassword</td>
</tr>
<tr>
<td>All Membership methods</td>
<td>ora_aspnet_Mem_SetPassword</td>
</tr>
<tr>
<td>All Membership methods</td>
<td>ora_aspnet_Mem_UnlockUser</td>
</tr>
<tr>
<td>All Membership methods</td>
<td>ora_aspnet_Mem_UpdateUser</td>
</tr>
<tr>
<td>All Membership methods</td>
<td>ora_aspnet_Mem_UpdateUserInfo</td>
</tr>
<tr>
<td>All Membership methods</td>
<td>ora_aspnet_Users_DeleteUser</td>
</tr>
</tbody>
</table>

#### 1.8.4.2 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

<table>
<thead>
<tr>
<th>Service Method</th>
<th>Stored Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetRolesForUser</td>
<td>ora_aspnet_UIR_GetRolesForUser</td>
</tr>
<tr>
<td>IsUserInRole</td>
<td>ora_aspnet_UIR_IsUserInRole</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Service Method</th>
<th>Stored Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>FindUsersInRole</td>
<td>ora_aspnet_UIR_FindUsersInRole</td>
</tr>
<tr>
<td>GetAllRoles</td>
<td>ora_aspnet_Roles_GetAllRoles</td>
</tr>
<tr>
<td>GetRolesForUser</td>
<td>ora_aspnet_UIR_GetRolesForUser</td>
</tr>
<tr>
<td>GetUsersInRole</td>
<td>ora_aspnet_UIR_GetUsersInRole</td>
</tr>
<tr>
<td>IsUserInRole</td>
<td>ora_aspnet_UIR_IsUserInRole</td>
</tr>
<tr>
<td>RoleExists</td>
<td>ora_aspnet_Roles_RoleExists</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Service Method</th>
<th>Stored Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>All Role Manager methods</td>
<td>ora_aspnet_Roles_CreateRole</td>
</tr>
<tr>
<td>All Role Manager methods</td>
<td>ora_aspnet_Roles_DeleteRole</td>
</tr>
<tr>
<td>All Role Manager methods</td>
<td>ora_aspnet_Roles_GetAllRoles</td>
</tr>
<tr>
<td>All Role Manager methods</td>
<td>ora_aspnet_Roles_RoleExists</td>
</tr>
<tr>
<td>All Role Manager methods</td>
<td>ora_aspnet_UIR_AddUsersToRoles</td>
</tr>
<tr>
<td>All Role Manager methods</td>
<td>ora_aspnet_UIR_FindUsersInRole</td>
</tr>
<tr>
<td>All Role Manager methods</td>
<td>ora_aspnet_UIR_GetRolesForUser</td>
</tr>
<tr>
<td>All Role Manager methods</td>
<td>ora_aspnet_UIR_GetUsersInRoles</td>
</tr>
<tr>
<td>All Role Manager methods</td>
<td>ora_aspnet_UIR_IsUserInRole</td>
</tr>
<tr>
<td>All Role Manager methods</td>
<td>ora_aspnet_UIR_RemUsersFmRoles</td>
</tr>
</tbody>
</table>

### 1.8.4.3 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

<table>
<thead>
<tr>
<th>Service Method</th>
<th>Stored Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetPropertyValues</td>
<td>ora_aspnet_Prof_GetProperties</td>
</tr>
<tr>
<td>SetPropertyValues</td>
<td>ora_aspnet_Prof_SetProperties</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Service Method</th>
<th>Stored Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetAllProfiles</td>
<td>ora_aspnet_Prof_GetProfiles</td>
</tr>
<tr>
<td>GetAllInactiveProfiles</td>
<td>ora_aspnet_Prof_GetProfiles</td>
</tr>
<tr>
<td>GetNumberOfInactiveProfiles</td>
<td>ora_aspnet_Prof_GetNumOfInactPf</td>
</tr>
<tr>
<td>FindProfilesByUserName</td>
<td>ora_aspnet_Prof_GetProfiles</td>
</tr>
<tr>
<td>FindInactiveProfilesByUserName</td>
<td>ora_aspnet_Prof_GetProfiles</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Service Method</th>
<th>Stored Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>All Profile methods</td>
<td>ora_aspnet_Prof_DeleteInactPf</td>
</tr>
<tr>
<td>All Profile methods</td>
<td>ora_aspnet_Prof_DeleteProfiles</td>
</tr>
<tr>
<td>All Profile methods</td>
<td>ora_aspnet_Prof_GetNumOfInactPf</td>
</tr>
<tr>
<td>All Profile methods</td>
<td>ora_aspnet_Prof_GetProfiles</td>
</tr>
<tr>
<td>All Profile methods</td>
<td>ora_aspnet_Prof_GetProperties</td>
</tr>
<tr>
<td>All Profile methods</td>
<td>ora_aspnet_Prof_SetProperties</td>
</tr>
</tbody>
</table>

1.8.4.4 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

<table>
<thead>
<tr>
<th>Service Method</th>
<th>Stored Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>LoadPersonalizationState</td>
<td>ora_aspnet_PPU_GetPgSettings</td>
</tr>
<tr>
<td></td>
<td>ora_aspnet_PAU_GetPgSettings</td>
</tr>
<tr>
<td>ResetPersonalizationState</td>
<td>ora_aspnet_PPU_ResetPgSettings</td>
</tr>
<tr>
<td></td>
<td>ora_aspnet_PAU_ResetPgSettings</td>
</tr>
<tr>
<td>SavePersonalizationState</td>
<td>ora_aspnet_PPU_SetPgSettings</td>
</tr>
<tr>
<td></td>
<td>ora_aspnet_PAU_SetPgSettings</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Service Method</th>
<th>Stored Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>FindState</td>
<td>ora_aspnet_PA_FindState</td>
</tr>
<tr>
<td>GetCountOfState</td>
<td>ora_aspnet_PA_GetCountOfState</td>
</tr>
</tbody>
</table>

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

<table>
<thead>
<tr>
<th>Service Method</th>
<th>Stored Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>All Personalization methods</td>
<td>ora_aspnet_PA_FindState</td>
</tr>
<tr>
<td>All Personalization methods</td>
<td>ora_aspnet_PA_GetCountOfState</td>
</tr>
<tr>
<td>All Personalization methods</td>
<td>ora_aspnet_PA_ResetSharedState</td>
</tr>
</tbody>
</table>
### Table 1-21  (Cont.) ora_aspnet_Pers_FullAccess Role

<table>
<thead>
<tr>
<th>Service Method</th>
<th>Stored Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>All Personalization methods</td>
<td>ora_aspnet_PA_ResetUserState</td>
</tr>
<tr>
<td>All Personalization methods</td>
<td>ora_aspnet_PAU_GetPgSettings</td>
</tr>
<tr>
<td>All Personalization methods</td>
<td>ora_aspnet_PAU_ResetPgSettings</td>
</tr>
<tr>
<td>All Personalization methods</td>
<td>ora_aspnet_PAU_SetPgSettings</td>
</tr>
<tr>
<td>All Personalization methods</td>
<td>ora_aspnet_PPU_GetPgSettings</td>
</tr>
<tr>
<td>All Personalization methods</td>
<td>ora_aspnet_PPU_ResetPgSettings</td>
</tr>
<tr>
<td>All Personalization methods</td>
<td>ora_aspnet_PPU_SetPgSettings</td>
</tr>
</tbody>
</table>

#### 1.8.4.5 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

<table>
<thead>
<tr>
<th>Service Method</th>
<th>Stored Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>All Web Event methods</td>
<td>ora_aspnet_LogWebEvents</td>
</tr>
</tbody>
</table>

#### 1.8.4.6 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

<table>
<thead>
<tr>
<th>Service Method</th>
<th>Stored Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>All Site Map methods</td>
<td>ora_aspnet_GetSiteMapData</td>
</tr>
</tbody>
</table>

#### 1.8.4.7 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

<table>
<thead>
<tr>
<th>Service Method</th>
<th>Stored Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>All Session State methods</td>
<td>ora_aspnet_SessnApp_SetAppID</td>
</tr>
<tr>
<td>All Session State methods</td>
<td>ora_aspnet_Sessn_InsUninitItem</td>
</tr>
<tr>
<td>All Session State methods</td>
<td>ora_aspnet_Sessn_RelStateItemEx</td>
</tr>
<tr>
<td>All Session State methods</td>
<td>ora_aspnet_Sessn_RmStateItem</td>
</tr>
</tbody>
</table>
Table 1-24  (Cont.) *ora_aspnet_Sessn_FullAccess* Role

<table>
<thead>
<tr>
<th>Service Method</th>
<th>Stored Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>All Session State methods</td>
<td><em>ora_aspnet_Sessn_ResetTimeout</em></td>
</tr>
<tr>
<td>All Session State methods</td>
<td><em>ora_aspnet_Sessn_UpdStateItem</em></td>
</tr>
<tr>
<td>All Session State methods</td>
<td><em>ora_aspnet_Sessn_InStateItem</em></td>
</tr>
<tr>
<td>All Session State methods</td>
<td><em>ora_aspnet_Sessn_GetStateItem</em></td>
</tr>
<tr>
<td>All Session State methods</td>
<td><em>ora_aspnet_Sessn_GetStateItmEx</em></td>
</tr>
</tbody>
</table>

1.8.5 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.
This chapter describes the OracleMembershipProvider class.

See Also:


This chapter contains the following topic:

• OracleMembershipProvider Class

2.1 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


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
<configuration version="1.0">
  <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
<configuration version="1.0">
  <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
2.1.1 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

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OracleMembershipProvider Constructors</td>
<td>Instantiates a new instance of the OracleMembershipProvider class</td>
</tr>
</tbody>
</table>

OracleMembershipProvider Static Methods

OracleMembershipProvider static methods are listed in Table 2-2.

Table 2-2  OracleMembershipProvider Static Methods

<table>
<thead>
<tr>
<th>Static Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equals</td>
<td>Inherited from System.Object (Overloaded)</td>
</tr>
<tr>
<td>ReferenceEquals</td>
<td>Inherited from System.Object</td>
</tr>
</tbody>
</table>

OracleMembershipProvider Public Properties

OracleMembershipProvider public properties are listed in Table 2-3.

Table 2-3  OracleMembershipProvider Public Properties

<table>
<thead>
<tr>
<th>Public Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ApplicationName</td>
<td>Gets or sets the name of the application that is used to group user information</td>
</tr>
</tbody>
</table>
Table 2-3  (Cont.) OracleMembershipProvider Public Properties

<table>
<thead>
<tr>
<th>Public Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CommandTimeout</td>
<td>Gets the number of seconds that the command is allowed to execute before it is terminated with an exception</td>
</tr>
<tr>
<td>Description</td>
<td>Inherited from System.Configuration.Provider.Providerbase</td>
</tr>
<tr>
<td>EnablePasswordReset</td>
<td>Indicates whether the membership provider is configured to allow users to reset their passwords</td>
</tr>
<tr>
<td>EnablePasswordRetrieval</td>
<td>Indicates whether the membership provider is configured to allow users to retrieve their passwords</td>
</tr>
<tr>
<td>MaxInvalidPasswordAttempts</td>
<td>Gets the number of invalid password or password-answer attempts allowed before the user is locked out</td>
</tr>
<tr>
<td>MinRequiredNonAlphanumericCharacters</td>
<td>Gets the minimum number of special characters that must be present in a valid password</td>
</tr>
<tr>
<td>MinRequiredPasswordLength</td>
<td>Gets the minimum length required for a password</td>
</tr>
<tr>
<td>Name</td>
<td>Inherited from System.Configuration.Provider.Providerbase</td>
</tr>
<tr>
<td>PasswordAttemptWindow</td>
<td>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</td>
</tr>
<tr>
<td>PasswordCompatMode</td>
<td>Gets the password compatibility mode.</td>
</tr>
<tr>
<td>PasswordFormat</td>
<td>Gets a value indicating the format for storing passwords in the membership data source</td>
</tr>
<tr>
<td>PasswordStrengthRegularExpression</td>
<td>Gets the regular expression used to evaluate a password</td>
</tr>
<tr>
<td>RequiresQuestionAndAnswer</td>
<td>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</td>
</tr>
<tr>
<td>RequiresUniqueEmail</td>
<td>Gets a value indicating whether the membership provider is configured to require a unique e-mail address for each user name</td>
</tr>
</tbody>
</table>

OracleMembershipProvider Public Methods

OracleMembershipProvider public methods are listed in Table 2-4.

Table 2-4  OracleMembershipProvider Public Methods

<table>
<thead>
<tr>
<th>Public Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ChangePassword</td>
<td>Updates the password for a user</td>
</tr>
</tbody>
</table>
Table 2-4  (Cont.) OracleMembershipProvider Public Methods

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

OracleMembershipProvider Public Events

OracleMembershipProvider public event is listed in Table 2-5.

Table 2-5  OracleMembershipProvider Public Event

<table>
<thead>
<tr>
<th>Public Event</th>
<th>Description</th>
</tr>
</thead>
</table>
2.1.2 OracleMembershipProvider Constructors

This constructor instantiates a new instance of the OracleMembershipProvider class.

Overload List:

• OracleMembershipProvider()
  This constructor creates an instance of the OracleMembershipProvider class.

2.1.2.1 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.
2.1.3 OracleMembershipProvider Static Methods

OracleMembershipProvider static methods are listed in Table 2-6.

<table>
<thead>
<tr>
<th>Static Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equals</td>
<td>Inherited from System.Object (Overloaded)</td>
</tr>
<tr>
<td>ReferenceEquals</td>
<td>Inherited from System.Object</td>
</tr>
</tbody>
</table>

See Also:
- “Oracle Providers for ASP.NET Assembly”
- OracleMembershipProvider Class
- OracleMembershipProvider Members

2.1.4 OracleMembershipProvider Public Properties

OracleMembershipProvider public properties are listed in Table 2-7.

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

<table>
<thead>
<tr>
<th>Public Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Inherited from System.Configuration.Provider.Providerbase</td>
</tr>
<tr>
<td>PasswordAttemptWindow</td>
<td>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</td>
</tr>
<tr>
<td>PasswordCompatMode</td>
<td>Gets the password compatibility mode.</td>
</tr>
<tr>
<td>PasswordFormat</td>
<td>Gets a value indicating the format for storing passwords in the membership data source</td>
</tr>
<tr>
<td>PasswordStrengthRegularExpression</td>
<td>Gets the regular expression used to evaluate a password</td>
</tr>
<tr>
<td>RequiresQuestionAndAnswer</td>
<td>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</td>
</tr>
<tr>
<td>RequiresUniqueEmail</td>
<td>Gets a value indicating whether the membership provider is configured to require a unique e-mail address for each user name</td>
</tr>
</tbody>
</table>

See Also:
- "Oracle Providers for ASP.NET Assembly"
- OracleMembershipProvider Class
- OracleMembershipProvider Members

### 2.1.4.1 ApplicationName

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

**Declaration**

```csharp
// 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”
- OracleMembershipProvider Class
- OracleMembershipProvider Members

2.1.4.2 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”
- OracleMembershipProvider Class
- OracleMembershipProvider Members
2.1.4.3 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"
- OracleMembershipProvider Class
- OracleMembershipProvider Members

2.1.4.4 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”
- `OracleMembershipProvider Class`
- `OracleMembershipProvider Members`

### 2.1.4.5 MaxInvalidPasswordAttempts

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

Declaration

```csharp
// 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.
2.1.4.6 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.

2.1.4.7 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 \texttt{minRequiredPasswordLength} attribute. The attribute name in the configuration file is case-sensitive.

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

\textbf{See Also:}

- “Oracle Providers for ASP.NET Assembly”
- OracleMembershipProvider Class
- OracleMembershipProvider Members

2.1.4.8 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.

\textbf{Declaration}

\begin{verbatim}
// C#
public override int PasswordAttemptWindow{get;}
\end{verbatim}

\textbf{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.

\textbf{Remarks}

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

The \texttt{PasswordAttemptWindow} property works in conjunction with the \texttt{MaxInvalidPasswordAttempts} property. If the number of invalid passwords or password question entries is greater than or equal to the \texttt{MaxInvalidPasswordAttempts} property value within the \texttt{PasswordAttemptWindow} property value (in minutes), then the user is locked out until the user is unlocked by the \texttt{UnlockUser} method. If a valid password or password answer is supplied before the \texttt{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 \texttt{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 \texttt{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"
- `OracleMembershipProvider` Class
- `OracleMembershipProviderMembers`

---

### 2.1.4.9 PasswordCompatMode

This property gets the password compatibility mode.

**Declaration**

```csharp
// 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:

```xml
<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`, `HMAC-SHA256`, `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="HMAC-SHA256">
  <providers>
    <add name="MyOracleMembershipProvider"
            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"
- OracleMembershipProvider Class
- OracleMembershipProvider Members

2.1.4.10 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"
• OracleMembershipProvider Class
• OracleMembershipProvider Members

2.1.4.11 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="(?=.*\d){1,}(?=.*[a-zA-Z]){1,}(?=.*[!@\#\$%^&*()]{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"
- `OracleMembershipProvider Class`
- `OracleMembershipProvider Members`

### 2.1.4.12 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**

```csharp
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"
- `OracleMembershipProvider Class`
- `OracleMembershipProvider Members`
2.1.4.13 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"
- OracleMembershipProvider Class
- OracleMembershipProvider Members

2.1.5 OracleMembershipProvider Public Methods

OracleMembershipProvider public methods are listed in Table 2-8.

Table 2-8  OracleMembershipProvider Public Methods

<table>
<thead>
<tr>
<th>Public Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ChangePassword</td>
<td>Updates the password for a user</td>
</tr>
<tr>
<td>ChangePasswordQuestionAndAnswer</td>
<td>Updates the password question and answer for a user</td>
</tr>
<tr>
<td>CreateUser</td>
<td>Adds a new user to the database</td>
</tr>
<tr>
<td>DeleteUser</td>
<td>Removes a user from the database</td>
</tr>
<tr>
<td>Equals</td>
<td>Inherited from System.Object (Overloaded)</td>
</tr>
<tr>
<td>FindUsersByEmail</td>
<td>Returns a collection of users whose e-mail addresses match the specified e-mail address</td>
</tr>
<tr>
<td>FindUsersByName</td>
<td>Returns a collection of users that match the specified user name</td>
</tr>
<tr>
<td>GeneratePassword</td>
<td>Generates a random password that is at least 14 characters in length</td>
</tr>
</tbody>
</table>
### Table 2-8  (Cont.) OracleMembershipProvider Public Methods

<table>
<thead>
<tr>
<th>Public Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetAllUsers</td>
<td>Returns a collection of all the users in the database</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Inherited from System.Object</td>
</tr>
<tr>
<td>GetNumberOfUsersOnline</td>
<td>Returns the number of users that are currently accessing the application</td>
</tr>
<tr>
<td>GetPassword</td>
<td>Returns the password for the specified user name from the database</td>
</tr>
<tr>
<td>GetType</td>
<td>Inherited from System.Object</td>
</tr>
<tr>
<td>GetUser</td>
<td>Returns user information from the database based on the unique identifier for the user (Overloaded)</td>
</tr>
<tr>
<td>GetUserNameByEmail</td>
<td>Returns the user name associated with the specified e-mail address</td>
</tr>
<tr>
<td>Initialize</td>
<td>Initializes the OracleMembership provider with the property values specified in the ASP.NET application configuration file (web.config)</td>
</tr>
<tr>
<td>ResetPassword</td>
<td>Resets a user's password and returns a new automatically generated password</td>
</tr>
<tr>
<td>ToString</td>
<td>Inherited from System.Object</td>
</tr>
<tr>
<td>UnlockUser</td>
<td>Unlocks a user so that the user can be validated</td>
</tr>
<tr>
<td>UpdateUser</td>
<td>Updates information about a user in the database</td>
</tr>
<tr>
<td>ValidateUser</td>
<td>Validates the user</td>
</tr>
</tbody>
</table>

See Also:

- "Oracle Providers for ASP.NET Assembly"
- OracleMembershipProvider Class
- OracleMembershipProvider Members

#### 2.1.5.1 ChangePassword

This method updates the password for a user.

**Declaration**

```csharp
// 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.

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"
• OracleMembershipProvider Class
• OracleMembershipProvider Members

2.1.5.2 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”
• OracleMembershipProvider Class
• OracleMembershipProvider Members
2.1.5.3 CreateUser

This method adds a new user to the database.

Declaration

```csharp
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

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

See Also:

- "Oracle Providers for ASP.NET Assembly"
- OracleMembershipProvider Class
- OracleMembershipProvider Members

2.1.5.4 DeleteUser

This method removes a user from the database.

Declaration

```csharp
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"
• OracleMembershipProvider Class
• OracleMembershipProvider Members

2.1.5.5 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"
- OracleMembershipProvider Class
- OracleMembershipProvider Members

2.1.5.6 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 \((pageIndex \times pageSize)\) and the page upper bound is \((pageIndex \times pageSize) + (pageSize - 1)\).

- `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"
- `OracleMembershipProvider Class`
- `OracleMembershipProvider Members`
2.1.5.7 GeneratePassword

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

Declaration

```csharp
// 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”
- `OracleMembershipProvider` Class
- `OracleMembershipProvider` Members

2.1.5.8 GetAllUsers

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

Declaration

```csharp
// 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`.

- **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"
- `OracleMembershipProvider Class`
- `OracleMembershipProvider Members`

### 2.1.5.9 GetNumberOfUsersOnline

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

#### Declaration

```csharp
// 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 application Name.

---

See Also:
- “Oracle Providers for ASP.NET Assembly”
- `OracleMembershipProvider Class`
- `OracleMembershipProvider Members`

---

2.1.5.10 GetPassword

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

Declaration

```csharp
// 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”
- OracleMembershipProvider Class
- OracleMembershipProvider Members

2.1.5.11 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.
2.1.5.12 GetUser(Object, bool)

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

Declaration

```csharp
// 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.
2.1.5.13 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.
2.1.5.14 GetUserNameByEmail

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

Declaration

```csharp
// 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

- ArgumentNullException - 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”
- OracleMembershipProvider Class
- OracleMembershipProvider Members

2.1.5.15 Initialize

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

```csharp
// 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.
2.1.5.16 ResetPassword

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

Declaration

```csharp
// 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"
- **OracleMembershipProvider Class**
- **OracleMembershipProvider Members**

### 2.1.5.17 UnlockUser

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

Declaration

```csharp
// 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”
• OracleMembershipProvider Class
• OracleMembershipProvider Members

2.1.5.18 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.

ArgumentNullException - 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"
• OracleMembershipProvider Class
• OracleMembershipProvider Members

2.1.5.19 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”
- OracleMembershipProvider Class
- OracleMembershipProvider Members

2.1.6 OracleMembershipProvider Public Events

OracleMembershipProvider public event is listed in Table 2-10.

Table 2-10 OracleMembershipProvider Public Events

<table>
<thead>
<tr>
<th>Public Event</th>
<th>Description</th>
</tr>
</thead>
</table>

See Also:

- “Oracle Providers for ASP.NET Assembly”
- OracleMembershipProvider Class
- OracleMembershipProvider Members
This chapter describes the OracleRoleProvider class.

See Also:
ASP.NET role management and role providers [http://msdn.microsoft.com/en-us/library/9ab2fxh0.aspx](http://msdn.microsoft.com/en-us/library/9ab2fxh0.aspx)

This chapter contains the following topic:

- OracleRoleProvider Class

### 3.1 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


Oracle.Web.Security

#### Declaration

```csharp
// 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
<?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
<?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"
             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
3.1.1 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

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OracleRoleProvider Constructors</td>
<td>Instantiates a new instance of the OracleRoleProvider class</td>
</tr>
</tbody>
</table>

**OracleRoleProvider Static Methods**

OracleRoleProvider static methods are listed in Table 3-2.

Table 3-2  OracleRoleProvider Static Methods

<table>
<thead>
<tr>
<th>Static Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equals</td>
<td>Inherited from System.Object (Overloaded)</td>
</tr>
<tr>
<td>ReferenceEquals</td>
<td>Inherited from System.Object</td>
</tr>
</tbody>
</table>

**OracleRoleProvider Public Properties**

OracleRoleProvider public properties are listed in Table 3-3.

Table 3-3  OracleRoleProvider Public Properties

<table>
<thead>
<tr>
<th>Public Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ApplicationName</td>
<td>Gets or sets the name of the application that stores the role provider information</td>
</tr>
<tr>
<td>CommandTimeout</td>
<td>Gets the number of seconds that the command is allowed to execute before it is terminated with an exception</td>
</tr>
</tbody>
</table>
Table 3-3  (Cont.) OracleRoleProvider Public Properties

<table>
<thead>
<tr>
<th>Public Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Inherited from <code>System.Configuration.Provider.Providerbase</code></td>
</tr>
<tr>
<td>Name</td>
<td>Inherited from <code>System.Configuration.Provider.Providerbase</code></td>
</tr>
</tbody>
</table>

OracleRoleProvider Public Methods

OracleRoleProvider public methods are listed in Table 3-4.

Table 3-4  OracleRoleProvider Public Methods

<table>
<thead>
<tr>
<th>Public Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AddUsersToRoles</td>
<td>Adds the specified users to the specified roles</td>
</tr>
<tr>
<td>CreateRole</td>
<td>Adds a new role to the database</td>
</tr>
<tr>
<td>DeleteRole</td>
<td>Deletes a role in the database</td>
</tr>
<tr>
<td>Equals</td>
<td>Inherited from <code>System.Object</code> (Overloaded)</td>
</tr>
<tr>
<td>FindUsersInRole</td>
<td>Returns an array of user names that match the specified user name</td>
</tr>
<tr>
<td>GetAllRoles</td>
<td>Returns an array of all the roles for an application</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Inherited from <code>System.Object</code></td>
</tr>
<tr>
<td>GetRolesForUser</td>
<td>Returns an array of role names for the specified user</td>
</tr>
<tr>
<td>GetType</td>
<td>Inherited from <code>System.Object</code></td>
</tr>
<tr>
<td>GetUsersInRole</td>
<td>Returns an array of users in the specified role name</td>
</tr>
<tr>
<td>Initialize</td>
<td>Initializes OracleRoleProvider with the property values specified in the ASP.NET application configuration file</td>
</tr>
<tr>
<td>IsUserInRole</td>
<td>Indicates whether the specified user is in the specified role</td>
</tr>
<tr>
<td>RemoveUsersFromRoles</td>
<td>Removes the specified array of users from the specified array of role names</td>
</tr>
<tr>
<td>RoleExists</td>
<td>Indicates whether the specified role name exists in the database</td>
</tr>
<tr>
<td>ToString</td>
<td>Inherited from <code>System.Object</code></td>
</tr>
</tbody>
</table>

See Also:

- "Oracle Providers for ASP.NET Assembly"
- OracleRoleProvider Class

3.1.2 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”
- `OracleRoleProvider` Class
- `OracleRoleProvider` Members

### 3.1.2.1 OracleRoleProvider()

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

**Declaration**

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

**Remarks**

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

See Also:

- “Oracle Providers for ASP.NET Assembly”
- `OracleRoleProvider` Class
- `OracleRoleProvider` Members

### 3.1.3 OracleRoleProvider Static Methods

The `OracleRoleProvider` static methods are listed in Table 3-5.

#### Table 3-5  OracleRoleProvider Static Methods

<table>
<thead>
<tr>
<th>Static Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equals(Overloaded)</td>
<td>Inherited from System.Object</td>
</tr>
<tr>
<td>ReferenceEquals</td>
<td>Inherited from System.Object</td>
</tr>
</tbody>
</table>
3.1.4 OracleRoleProvider Public Properties

The OracleRoleProvider public properties are listed in Table 3-6.

Table 3-6  OracleRoleProvider Public Properties

<table>
<thead>
<tr>
<th>Public Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ApplicationName</td>
<td>Gets or sets the name of the application that stores the role provider information.</td>
</tr>
<tr>
<td>CommandTimeout</td>
<td>Gets the number of seconds that the command is allowed to execute before it is terminated with an exception.</td>
</tr>
<tr>
<td>Name</td>
<td>Inherited from System.Configuration.Provider.Providerbase.</td>
</tr>
</tbody>
</table>

3.1.4.1 ApplicationName

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

Declaration

```csharp
// 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"
• OracleRoleProvider Class
• OracleRoleProvider Members

3.1.4.2 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"
- OracleRoleProvider Class
- OracleRoleProvider Members

3.1.5 OracleRoleProvider Public Methods

The OracleRoleProvider public methods are listed in Table 3-7.

Table 3-7  OracleRoleProvider Public Methods

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

See Also:

- "Oracle Providers for ASP.NET Assembly"
- OracleRoleProvider Class
- OracleRoleProvider Members
3.1.5.1 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”
• OracleRoleProvider Class
• OracleRoleProvider Members

3.1.5.2 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"
• OracleRoleProvider Class
• OracleRoleProvider Members

3.1.5.3 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”
- OracleRoleProvider Class
- OracleRoleProvider Members

3.1.5.4 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"
- OracleRoleProvider Class
- OracleRoleProvider Members

3.1.5.5 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.
3.1.5.6 GetRolesForUser

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

Declaration

```csharp
// 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”
- `OracleRoleProvider Class`
- `OracleRoleProvider Members`

3.1.5.7 GetUsersInRole

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

Declaration

```csharp
// 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"
- OracleRoleProvider Class
- OracleRoleProvider Members

### 3.1.5.8 Initialize

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

Declaration

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

Parameters

- **name**
  The name of the OracleRoleProvider instance to initialize.
- **config**
  A `System.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”
• OracleRoleProvider Class
• OracleRoleProvider Members

3.1.5.9 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.
### 3.1.5.10 RemoveUsersFromRoles

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

**Declaration**

```csharp
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.
3.1.5.11 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”
• OracleRoleProvider Class
• OracleRoleProvider Members
This chapter describes the `OracleSiteMapProvider` class.

### See Also:


This chapter contains the following topic:

- **OracleSiteMapProvider Class**

### 4.1 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

```csharp
// 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 Oracle-SiteMapProvider as the default provider. This configuration uses the connection string and default attribute values specified in the machine.config file.

```xml
<?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 Oracle-SiteMapProvider as the default provider, with customized settings and an application-specific connection string:

```xml
<?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.
4.1.1 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

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OracleSiteMapProvider Constructors</td>
<td>Instantiates a new instance of the OracleSiteMapProvider class</td>
</tr>
</tbody>
</table>

OracleSiteMapProvider Static Methods

OracleSiteMapProvider static methods are listed in Table 4-2.

Table 4-2  OracleSiteMapProvider Static Methods

<table>
<thead>
<tr>
<th>Static Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equals</td>
<td>Inherited from System.Object (Overloaded)</td>
</tr>
<tr>
<td>ReferenceEquals</td>
<td>Inherited from System.Object</td>
</tr>
</tbody>
</table>

OracleSiteMapProvider Public Properties

OracleSiteMapProvider public properties are listed in Table 4-3.

Table 4-3  OracleSiteMapProvider Public Properties

<table>
<thead>
<tr>
<th>Public Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ApplicationName</td>
<td>Gets or sets the name of the application that differentiates site map data for different applications</td>
</tr>
<tr>
<td>CommandTimeout</td>
<td>Gets the number of seconds that the command is allowed to execute before it is terminated with an exception</td>
</tr>
<tr>
<td>CurrentNode</td>
<td>Inherited from System.Web.SiteMapProvider (Read-Only)</td>
</tr>
</tbody>
</table>
Table 4-3  (Cont.) OracleSiteMapProvider Public Properties

<table>
<thead>
<tr>
<th>Public Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Inherited from System.Configuration.Provider.Providerbase</td>
</tr>
<tr>
<td>EnableLocalization</td>
<td>Inherited from System.Web.SiteMapProvider</td>
</tr>
<tr>
<td>Name</td>
<td>Inherited from System.Configuration.Provider.Providerbase</td>
</tr>
<tr>
<td>ParentProvider</td>
<td>Inherited from System.Web.SiteMapProvider</td>
</tr>
<tr>
<td>ResourceKey</td>
<td>Inherited from System.Web.SiteMapProvider</td>
</tr>
<tr>
<td>RootNode</td>
<td>Inherited from System.Web.SiteMapProvider (Read-Only)</td>
</tr>
<tr>
<td>RootProvider</td>
<td>Inherited from System.Web.SiteMapProvider</td>
</tr>
<tr>
<td>SecurityTrimmingEnabled</td>
<td>Inherited from System.Web.SiteMapProvider</td>
</tr>
</tbody>
</table>

OracleSiteMapProvider Public Methods

OracleSiteMapProvider public methods are listed in Table 4-4.

Table 4-4  OracleSiteMapProvider Public Methods

<table>
<thead>
<tr>
<th>Public Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BuildSiteMap</td>
<td>Builds a SiteMap tree of the SiteMapNode objects by loading site map data from an Oracle database</td>
</tr>
<tr>
<td>Dispose</td>
<td>Releases all the resources for this instance</td>
</tr>
<tr>
<td>FindSiteMapNode</td>
<td>Inherited from System.Web.StaticSiteMapProvider</td>
</tr>
<tr>
<td>FindSiteMapNodeFromKey</td>
<td>Inherited from System.Web.StaticSiteMapProvider</td>
</tr>
<tr>
<td>GetChildNodes</td>
<td>Inherited from System.Web.StaticSiteMapProvider</td>
</tr>
<tr>
<td>GetCurrentNodeAndHintAncestorNodes</td>
<td>Inherited from System.Web.SiteMapProvider</td>
</tr>
<tr>
<td>GetCurrentNodeAndHintNeighborhoodNodes</td>
<td>Inherited from System.Web.SiteMapProvider</td>
</tr>
<tr>
<td>GetParentNode</td>
<td>Inherited from System.Web.StaticSiteMapProvider</td>
</tr>
<tr>
<td>GetParentNodeRelativeToCurrentNodeAndHintDownFromParent</td>
<td>Inherited from System.Web.SiteMapProvider</td>
</tr>
<tr>
<td>GetParentNodeRelativeToNodeAndHintDownFromParent</td>
<td>Inherited from System.Web.SiteMapProvider</td>
</tr>
<tr>
<td>HintAncestorNodes</td>
<td>Inherited from System.Web.SiteMapProvider</td>
</tr>
<tr>
<td>IsAccessibleToUser</td>
<td>Inherited from System.Web.SiteMapProvider</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Inherited from System.Object</td>
</tr>
<tr>
<td>GetType</td>
<td>Inherited from System.Object</td>
</tr>
</tbody>
</table>
Table 4-4  (Cont.) OracleSiteMapProvider Public Methods

<table>
<thead>
<tr>
<th>Public Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initialize</td>
<td>Initializes the OracleSiteMapProvider instance with the property values specified in the ASP.NET application configuration file</td>
</tr>
<tr>
<td>ToString</td>
<td>Inherited from System.Object</td>
</tr>
</tbody>
</table>

See Also:
- “Oracle Providers for ASP.NET Assembly”
- OracleSiteMapProvider Class

4.1.2 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

4.1.2.1 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.
4.1.3 OracleSiteMapProvider Static Methods

OracleSiteMapProvider static methods are listed in Table 4-5.

Table 4-5  OracleSiteMapProvider Static Methods

<table>
<thead>
<tr>
<th>Static Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equals</td>
<td>Inherited from System.Object (Overloaded)</td>
</tr>
<tr>
<td>ReferenceEquals</td>
<td>Inherited from System.Object</td>
</tr>
</tbody>
</table>

4.1.4 OracleSiteMapProvider Public Properties

OracleSiteMapProvider public properties are listed in Table 4-6.

Table 4-6  OracleSiteMapProvider Public Properties

<table>
<thead>
<tr>
<th>Public Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ApplicationName</td>
<td>Gets or sets the name of the application that differentiates site map data for different applications</td>
</tr>
<tr>
<td>CommandTimeout</td>
<td>Gets the number of seconds that the command is allowed to execute before it is terminated with an exception</td>
</tr>
<tr>
<td>CurrentNode</td>
<td>Inherited from System.Web.SiteMapProvider</td>
</tr>
<tr>
<td>Description</td>
<td>Inherited from System.Configuration.Providerbase</td>
</tr>
<tr>
<td>EnableLocalization</td>
<td>Inherited from System.Web.SiteMapProvider</td>
</tr>
<tr>
<td>Name</td>
<td>Inherited from System.Configuration.Providerbase</td>
</tr>
<tr>
<td>ParentProvider</td>
<td>Inherited from System.Web.SiteMapProvider</td>
</tr>
<tr>
<td>ResourceKey</td>
<td>Inherited from System.Web.SiteMapProvider</td>
</tr>
<tr>
<td>RootNode</td>
<td>Inherited from System.Web.SiteMapProvider</td>
</tr>
<tr>
<td>RootProvider</td>
<td>Inherited from System.Web.SiteMapProvider</td>
</tr>
<tr>
<td>SecurityTrimmingEnabled</td>
<td>Inherited from System.Web.SiteMapProvider</td>
</tr>
</tbody>
</table>
4.1.4.1 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.


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.
4.1.4.2 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.

4.1.5 OracleSiteMapProvider Public Methods

OracleSiteMapProvider public methods are listed in Table 4-7.

Table 4-7  OracleSiteMapProvider Public Methods

<table>
<thead>
<tr>
<th>Public Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BuildSiteMap</td>
<td>Builds a SiteMap tree of the SiteMapNode objects by loading site map data from an Oracle database</td>
</tr>
<tr>
<td>Dispose</td>
<td>Releases all the resources for this instance</td>
</tr>
<tr>
<td>FindSiteMapNode</td>
<td>Inherited from System.Web.StaticSiteMapProvider</td>
</tr>
</tbody>
</table>
Table 4-7  (Cont.) OracleSiteMapProvider Public Methods

<table>
<thead>
<tr>
<th>Public Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FindSiteMapNodeFromKey</td>
<td>Inherited from System.Web.StaticSiteMapProvider</td>
</tr>
<tr>
<td>GetChildNodes</td>
<td>Inherited from System.Web.StaticSiteMapProvider</td>
</tr>
<tr>
<td>GetCurrentNodeAndHintAncestorNodes</td>
<td>Inherited from System.Web.SiteMapProvider</td>
</tr>
<tr>
<td>GetCurrentNodeAndHintNeighborNodes</td>
<td>Inherited from System.Web.SiteMapProvider</td>
</tr>
<tr>
<td>GetParentNode</td>
<td>Inherited from System.Web.StaticSiteMapProvider</td>
</tr>
<tr>
<td>GetParentNodeRelativeToCurrentNodeAndHintDownFromPar-</td>
<td>Inherited from System.Web.SiteMapProvider</td>
</tr>
<tr>
<td>enet</td>
<td>Inherited from System.Web.SiteMapProvider</td>
</tr>
<tr>
<td>GetParentNodeRelativeToNodeAndHintDownFromParent</td>
<td>Inherited from System.Web.SiteMapProvider</td>
</tr>
<tr>
<td>HintAncestorNodes</td>
<td>Inherited from System.Web.SiteMapProvider</td>
</tr>
<tr>
<td>IsAccessibleToUser</td>
<td>Inherited from System.Web.SiteMapProvider</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Inherited from System.Object</td>
</tr>
<tr>
<td>GetType</td>
<td>Inherited from System.Object</td>
</tr>
<tr>
<td><strong>Initialize</strong></td>
<td>Initializes the OracleSiteMapProvider instance with the property values sp-</td>
</tr>
<tr>
<td></td>
<td>eified in the ASP.NET application's configuration file</td>
</tr>
<tr>
<td><strong>ToString</strong></td>
<td>Inherited from System.Object</td>
</tr>
</tbody>
</table>

See Also:

- “Oracle Providers for ASP.NET Assembly”
- OracleSiteMapProvider Class
- OracleSiteMapProvider Members

4.1.5.1 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.

ConfigurationException - 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"
- OracleSiteMapProvider Class
- OracleSiteMapProvider Members
- Oracle Data Provider for .NET Developer’s Guide for Microsoft Windows for more information about continuous query notification support

4.1.5.2 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.
4.1.5.3 Initialize

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

Declaration

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

Parameters

- `name`
  The name of the `OracleSiteMapProvider` instance to initialize.

- `config`
  A `System.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"
- OracleSiteMapProvider Class
- OracleSiteMapProvider Members
5

OracleSessionStateStore

This chapter describes the OracleSessionStateStore class.

See Also:

This chapter contains the following topic:

• OracleSessionStateStore Class

5.1 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
<?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"
- OracleSessionStateStore Members
- OracleSessionStateStore Constructors
- OracleSessionStateStore Public Methods
- OracleSessionStateStore Public Properties

5.1.1 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

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OracleSessionStateStore Constructor</td>
<td>Instantiates a new instance of the OracleSessionStateStore class</td>
</tr>
</tbody>
</table>

OracleSessionStateStore Public Properties

OracleSessionStateStore public properties are listed in Table 5-2.

Table 5-2  OracleSessionStateStore Public Properties

<table>
<thead>
<tr>
<th>Public Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CommandTimeout</td>
<td>Gets the number of seconds that the command is allowed to execute before it is terminated with an exception</td>
</tr>
<tr>
<td>Description</td>
<td>Inherited from System.Configuration.Provider.ProviderBase</td>
</tr>
<tr>
<td>Name</td>
<td>Inherited from System.Configuration.Provider.ProviderBase</td>
</tr>
</tbody>
</table>

OracleSessionStateStore Public Methods

The OracleSessionStateStore public methods are listed in Table 5-3.

Table 5-3  OracleSessionStateStore Public Methods

<table>
<thead>
<tr>
<th>Public Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CreateNewStoreData</td>
<td>Creates a new SessionStateStoreData object for the current request</td>
</tr>
<tr>
<td>CreateUninitializedItem</td>
<td>Adds a new session state item to the database</td>
</tr>
<tr>
<td>Dispose</td>
<td>Releases all the resources for this instance</td>
</tr>
<tr>
<td>EndRequest</td>
<td>Allows the OracleSessionStateStore object to perform any cleanup that may be required for the current request</td>
</tr>
<tr>
<td>GetItem</td>
<td>Returns a read-only session item from the database</td>
</tr>
<tr>
<td>GetItemExclusive</td>
<td>Locks and returns a session item from the database</td>
</tr>
<tr>
<td>Initialize</td>
<td>Initializes the provider with the property values specified in the ASP.NET application configuration file</td>
</tr>
<tr>
<td>InitializeRequest</td>
<td>Performs any per-request initializations that the OracleSessionStateStore provider requires</td>
</tr>
<tr>
<td>ReleaseItemExclusive</td>
<td>Releases the lock on a session item in the database, if multiple attempts to retrieve the session item fail</td>
</tr>
<tr>
<td>RemoveItem</td>
<td>Removes the specified session item from the database</td>
</tr>
<tr>
<td>ResetItemTimeout</td>
<td>Resets the expiration date and timeout for a session item in the database</td>
</tr>
<tr>
<td>SetAndReleaseItemExclusive</td>
<td>Updates the session time information in the database with the specified session item and releases the lock</td>
</tr>
</tbody>
</table>
Table 5-3 (Cont.) OracleSessionStateStore Public Methods

<table>
<thead>
<tr>
<th>Public Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SetItemExpireCallback</td>
<td>Returns a false value to indicate that callbacks for expired sessions are not supported</td>
</tr>
</tbody>
</table>

See Also:

- "Oracle Providers for ASP.NET Assembly"
- OracleSessionStateStore Class

5.1.2 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"
- OracleSessionStateStore Class
- OracleSessionStateStore Members

5.1.2.1 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.
5.1.3 OracleSessionStateStore Public Properties

The OracleSessionStateStore public properties are listed in Table 5-4.

Table 5-4  OracleSessionStateStore Public Properties

<table>
<thead>
<tr>
<th>Public Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CommandTimeout</td>
<td>Gets the number of seconds that the command is allowed to execute before it is terminated with an exception</td>
</tr>
<tr>
<td>Description</td>
<td>Inherited from System.Configuration.Provider.Providerbase</td>
</tr>
<tr>
<td>Name</td>
<td>Inherited from System.Configuration.Provider.Providerbase</td>
</tr>
</tbody>
</table>

5.1.3.1 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"
- OracleSessionStateStore Class
- OracleSessionStateStore Members

5.1.4 OracleSessionStateStore Public Methods

The OracleSessionStateStore public methods are listed in Table 5-5.

Table 5-5  OracleSessionStateStore Public Methods

<table>
<thead>
<tr>
<th>Public Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CreateNewStoreData</td>
<td>Creates a new SessionStateStoreData object for the current request</td>
</tr>
<tr>
<td>CreateUninitializedItem</td>
<td>Adds a new session state item to the database</td>
</tr>
<tr>
<td>Dispose</td>
<td>Releases all the resources for this instance</td>
</tr>
<tr>
<td>EndRequest</td>
<td>Allows the OracleSessionStateStore object to perform any cleanup that may be required for the current request</td>
</tr>
<tr>
<td>GetItem</td>
<td>Returns a read-only session item from the database</td>
</tr>
<tr>
<td>GetItemExclusive</td>
<td>Locks and returns a session item from the database</td>
</tr>
<tr>
<td>Initialize</td>
<td>Initializes the provider with the property values specified in the ASP.NET application configuration file</td>
</tr>
<tr>
<td>InitializeRequest</td>
<td>Performs any per-request initializations that the OracleSessionStateStore provider requires</td>
</tr>
<tr>
<td>ReleaseItemExclusive</td>
<td>Releases the lock on a session item in the database, if multiple attempts to retrieve the session item fail</td>
</tr>
<tr>
<td>RemoveItem</td>
<td>Removes the specified session item from the database</td>
</tr>
<tr>
<td>ResetItemTimeout</td>
<td>Resets the expiration date and timeout for a session item in the database</td>
</tr>
<tr>
<td>SetAndReleaseItemExclu‐</td>
<td>Updates the session time information in the database with the specified session item and releases the lock</td>
</tr>
<tr>
<td>sive</td>
<td>SetItemExpireCallback</td>
</tr>
</tbody>
</table>
5.1.4.1 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.

5.1.4.2 CreateUninitializedItem

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

```csharp
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"
- OracleSessionStateStore Class
- OracleSessionStateStore Members

5.1.4.3 Dispose

This method releases all the resources for this instance.

Declaration

```csharp
public override void Dispose();
```

Remarks

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

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

**Declaration**

```csharp
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”
- OracleSessionStateStore Class
- OracleSessionStateStore Members

---

### 5.1.4.5 GetItem

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

**Declaration**

```csharp
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 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**

- ArgumentException - 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.
5.1.4.6 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 new session 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"
- OracleSessionStateStore Class
- OracleSessionStateStore Members

5.1.4.7 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

ArgumentException - 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”
- OracleSessionStateStore Class
- OracleSessionStateStore Members

5.1.4.8 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”
- OracleSessionStateStore Class
- OracleSessionStateStore Members

5.1.4.9 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 HttpRuntimeSection.ExecutionTimeout property value.

See Also:

• “Oracle Providers for ASP.NET Assembly”
• OracleSessionStateStore Class
• OracleSessionStateStore Members

5.1.4.10 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”
- `OracleSessionStateStore Class`
- `OracleSessionStateStore Members`

### 5.1.4.11 ResetItemTimeout

This method resets the expiration date and timeout for a session item in the database.

**Declaration**

```csharp
// 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.
5.1.4.12 SetAndReleaseItemExclusive

This method updates the session time information in the database with the specified session item, and releases the lock.

**Declaration**

```csharp
// 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.
5.1.4.13 SetItemExpireCallback

This method returns a false value to indicate that callbacks for expired sessions are not supported.

Declaration

```csharp
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.
This chapter describes the **OracleProfileProvider** class.

### See Also:


This chapter contains the following topic:

- **OracleProfileProvider Class**

## 6.1 OracleProfileProvider Class

The **OracleProfileProvider** class 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
<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
<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.
6.1.1 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

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OracleProfileProvider Constructors</td>
<td>Instantiates a new instance of the OracleProfileProvider class</td>
</tr>
</tbody>
</table>

OracleProfileProvider Static Methods

OracleProfileProvider static methods are listed in Table 6-2.

Table 6-2  OracleProfileProvider Static Methods

<table>
<thead>
<tr>
<th>Static Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equals</td>
<td>Inherited from System.Object (Overloaded)</td>
</tr>
<tr>
<td>ReferenceEquals</td>
<td>Inherited from System.Object</td>
</tr>
</tbody>
</table>

OracleProfileProvider Public Properties

OracleProfileProvider public properties are listed in Table 6-3.
Table 6-3  OracleProfileProvider Public Properties

<table>
<thead>
<tr>
<th>Public Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ApplicationName</td>
<td>Gets or sets the name of the application that groups the profile information</td>
</tr>
<tr>
<td>CommandTimeout</td>
<td>Gets the number of seconds that the command is allowed to execute before it is terminated with an exception</td>
</tr>
<tr>
<td>Description</td>
<td>Inherited from System.Configuration.Provider.Providerbase</td>
</tr>
<tr>
<td>Name</td>
<td>Inherited from System.Configuration.Provider.Providerbase</td>
</tr>
</tbody>
</table>

OracleProfileProvider Public Methods

OracleProfileProvider public methods are listed in Table 6-4.

Table 6-4  OracleProfileProvider Public Methods

<table>
<thead>
<tr>
<th>Public Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeleteInactiveProfiles</td>
<td>Deletes user profile data that has its last activity date on or before the specified date and time</td>
</tr>
<tr>
<td>DeleteProfiles</td>
<td>Deletes profile properties and information from the data source for the supplied profile collection or list of user names (Overloaded)</td>
</tr>
<tr>
<td>Equals</td>
<td>Inherited from System.Object (Overloaded)</td>
</tr>
<tr>
<td>FindInactiveProfilesByUserName</td>
<td>Retrieves inactive profile information for the specified user name</td>
</tr>
<tr>
<td>FindProfilesByUserName</td>
<td>Retrieves profile information for the specified user name</td>
</tr>
<tr>
<td>GetAllInactiveProfiles</td>
<td>Retrieves all profile information for profiles with the last activity date on or before the specified date and time</td>
</tr>
<tr>
<td>GetAllProfiles</td>
<td>Retrieves all profile information from the data source</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Inherited from System.Object</td>
</tr>
<tr>
<td>GetNumberOfInactiveProfiles</td>
<td>Returns the count of profiles where the last activity date is on or before the specified date and time</td>
</tr>
<tr>
<td>GetPropertyValues</td>
<td>Retrieves profile properties and values from the Oracle profile database</td>
</tr>
<tr>
<td>GetType</td>
<td>Inherited from System.Object</td>
</tr>
<tr>
<td>Initialize</td>
<td>Initializes the OracleProfileProvider instance with the property values specified in the ASP.NET application configuration file (web.config)</td>
</tr>
<tr>
<td>SetPropertyValues</td>
<td>Updates the Oracle profile database with the specified profile property values</td>
</tr>
<tr>
<td>ToString</td>
<td>Inherited from System.Object</td>
</tr>
</tbody>
</table>
6.1.2 OracleProfileProvider Constructors

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

**Overload List:**

- `OracleProfileProvider()`

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

6.1.2.1 OracleProfileProvider()

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

**Declaration**

```csharp
// 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.
6.1.3 OracleProfileProvider Static Methods

OracleProfileProvider static methods are listed in Table 6-5.

Table 6-5  OracleProfileProvider Static Methods

<table>
<thead>
<tr>
<th>Static Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equals</td>
<td>Inherited from System.Object (Overloaded)</td>
</tr>
<tr>
<td>ReferenceEquals</td>
<td>Inherited from System.Object</td>
</tr>
</tbody>
</table>

See Also:

- "Oracle Providers for ASP.NET Assembly"
- OracleProfileProvider Class
- OracleProfileProvider Members

6.1.4 OracleProfileProvider Public Properties

OracleProfileProvider public properties are listed in Table 6-6.

Table 6-6  OracleProfileProvider Public Properties

<table>
<thead>
<tr>
<th>Public Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ApplicationName</td>
<td>Gets or sets the name of the application that groups the profile information</td>
</tr>
<tr>
<td>CommandTimeout</td>
<td>Gets the number of seconds that the command is allowed to execute before it is terminated with an exception</td>
</tr>
<tr>
<td>Description</td>
<td>Inherited from System.Configuration.Provider.Providerbase</td>
</tr>
<tr>
<td>Name</td>
<td>Inherited from System.Configuration.Provider.Providerbase</td>
</tr>
</tbody>
</table>

See Also:

- "Oracle Providers for ASP.NET Assembly"
- OracleProfileProvider Class
- OracleProfileProvider Members
6.1.4.1 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.


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"
- OracleProfileProvider Class
- OracleProfileProvider Members

6.1.4.2 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"
- OracleProfileProvider Class
- OracleProfileProvider Members

6.1.5 OracleProfileProvider Public Methods

OracleProfileProvider public methods are listed in Table 6-7.

Table 6-7  OracleProfileProvider Public Methods

<table>
<thead>
<tr>
<th>Public Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeleteInactiveProfiles</td>
<td>Deletes user profile data that has its last activity date on or before the specified date and time</td>
</tr>
<tr>
<td>DeleteProfiles</td>
<td>Deletes profile properties and information from the data source for the supplied profile collection or list of user names (Overloaded)</td>
</tr>
<tr>
<td>Equals</td>
<td>Inherited from System.Object (Overloaded)</td>
</tr>
<tr>
<td>FindInactiveProfilesByUserName</td>
<td>Retrieves inactive profile information for the specified user name</td>
</tr>
<tr>
<td>FindProfilesByUserName</td>
<td>Retrieves profile information for the specified user name</td>
</tr>
<tr>
<td>GetAllInactiveProfiles</td>
<td>Retrieves all profile information for profiles with the last activity date on or before the specified date and time</td>
</tr>
<tr>
<td>GetAllProfiles</td>
<td>Retrieves all profile information from the data source</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Inherited from System.Object</td>
</tr>
<tr>
<td>GetNumberOfInactiveProfiles</td>
<td>Returns the count of profiles where the last activity date is on or before the specified date and time</td>
</tr>
</tbody>
</table>
Table 6-7  (Cont.) OracleProfileProvider Public Methods

<table>
<thead>
<tr>
<th>Public Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetPropertyValues</td>
<td>Retrieves profile properties and values from the Oracle profile database</td>
</tr>
<tr>
<td>GetType</td>
<td>Inherited from System.Object</td>
</tr>
<tr>
<td>Initialize</td>
<td>Initializes the OracleProfileProvider instance with the property values specified in the ASP.NET application configuration file (web.config)</td>
</tr>
<tr>
<td>SetPropertyValues</td>
<td>Updates the Oracle profile database with the specified profile property values</td>
</tr>
<tr>
<td>ToString</td>
<td>Inherited from System.Object</td>
</tr>
</tbody>
</table>

See Also:

- “Oracle Providers for ASP.NET Assembly”
- OracleProfileProvider Class
- OracleProfileProvider Members

6.1.5.1 DeleteInactiveProfiles

This method deletes user profile data that has its last activity date on or before the specified date and time.

Declaration

```csharp
// 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 `profileAu-
the `AuthenticationOption` 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.

### 6.1.5.2 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"
- OracleProfileProvider Class
- OracleProfileProvider Members

### 6.1.5.3 DeleteProfiles(ProfileInfoCollection)

This method deletes profile properties and information from the data source for the supplied profile collection.

**Declaration**

```csharp
// 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”
- `OracleProfileProvider Class`
- `OracleProfileProvider Members`

6.1.5.4 `DeleteProfiles(string[])`

This method deletes profile properties and information from the data source for the supplied list of user names.
Declaration

```csharp
// 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"
- OracleProfileProvider Class
- OracleProfileProvider Members

6.1.5.5 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 speci-

fied by the applicationName attribute in the configuration file. The profileAuthenti-
cationOption 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 param-
eter 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 `inactiveSince-DateTime` parameters.

### See Also:

- "Oracle Providers for ASP.NET Assembly"
- `OracleProfileProvider` Class
- `OracleProfileProvider` Members

### 6.1.5.6 FindProfilesByUserName

This method retrieves profile information for the specified user name.

#### Declaration

```csharp
// 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”
• `OracleProfileProvider` Class
• `OracleProfileProvider` Members

6.1.5.7 GetAllInactiveProfiles

This method retrieves all profile information for profiles with the last activity date on or before the specified date and time.

Declaration

```csharp
// 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 in 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"
- `OracleProfileProvider Class`
- `OracleProfileProvider Members`

### 6.1.5.8 GetAllProfiles

This method retrieves all profile information from the data source.

**Declaration**

```csharp
// 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”
- OracleProfileProvider Class
- OracleProfileProvider Members

6.1.5.9 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 profileAu-
thenticationOption 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"
• OracleProfileProvider Class
• OracleProfileProvider Members

6.1.5.10 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”
- OracleProfileProvider Class
- OracleProfileProvider Members

### 6.1.5.11 Initialize

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

**Declaration**

```csharp
// 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"
• `OracleProfileProvider` Class
• `OracleProfileProvider` Members

### 6.1.5.12 SetPropertyValues

This method updates the Oracle profile database with the specified profile property values.

Declaration

```csharp
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”
- OracleProfileProvider Class
- OracleProfileProvider Members
This chapter describes the OracleWebEventProvider class.

See Also:


This chapter contains the following topic:

- OracleWebEventProvider Class

### 7.1 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


#### Declaration

```csharp
// 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
<?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
<?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" 
          connectionStringName="my_webevent_app_con_string" 
          bufferMode="CustomBufferMode">
```
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”
- `OracleWebEventProvider` Members
- `OracleWebEventProvider` Constructors
- `OracleWebEventProvider` Static Methods
- `OracleWebEventProvider` Public Properties
- `OracleWebEventProvider` Public Methods

---

### 7.1.1 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

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OracleWebEventProvider Constructors</td>
<td>Instantiates a new instance of the OracleWebEventProvider class</td>
</tr>
</tbody>
</table>

OracleWebEventProvider Static Methods

OracleWebEventProvider static methods are listed in Table 7-2.

Table 7-2  OracleWebEventProvider Static Methods

<table>
<thead>
<tr>
<th>Static Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equals</td>
<td>Inherited from System.Object (Overloaded)</td>
</tr>
<tr>
<td>ReferenceEquals</td>
<td>Inherited from System.Object</td>
</tr>
</tbody>
</table>

OracleWebEventProvider Public Properties

OracleWebEventProvider public properties are listed in Table 7-3.

Table 7-3  OracleWebEventProvider Public Properties

<table>
<thead>
<tr>
<th>Public Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BufferMode</td>
<td>Inherited from System.Web.Management.BufferedWebEventProvider</td>
</tr>
<tr>
<td>CommandTimeout</td>
<td>Gets the number of seconds that the command is allowed to execute before it is terminated with an exception</td>
</tr>
<tr>
<td>Description</td>
<td>Inherited from System.Configuration.Provider.ProviderBase</td>
</tr>
<tr>
<td>Name</td>
<td>Inherited from System.Configuration.Provider.ProviderBase</td>
</tr>
</tbody>
</table>

OracleWebEventProvider Public Methods

OracleWebEventProvider public methods are listed in Table 7-4.

Table 7-4  OracleWebEventProvider Public Methods

<table>
<thead>
<tr>
<th>Public Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initialize</td>
<td>Initializes the OracleWebEventProvider instance with the property values specified in the ASP.NET application configuration file</td>
</tr>
<tr>
<td>ProcessEvent</td>
<td>Processes the event passed to it as an argument</td>
</tr>
<tr>
<td>ProcessEventFlush</td>
<td>Flushes the information passed to it as an argument</td>
</tr>
</tbody>
</table>
Table 7-4  (Cont.) OracleWebEventProvider Public Methods

<table>
<thead>
<tr>
<th>Public Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shutdown</td>
<td>Releases all resources</td>
</tr>
<tr>
<td>Flush</td>
<td>Inherited from System.BufferedWebEventProvider</td>
</tr>
<tr>
<td>Equals(Overloaded)</td>
<td>Inherited from System.Object</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Inherited from System.Object</td>
</tr>
<tr>
<td>GetType</td>
<td>Inherited from System.Object</td>
</tr>
<tr>
<td>ToString</td>
<td>Inherited from System.Object</td>
</tr>
</tbody>
</table>

See Also:

- “Oracle Providers for ASP.NET Assembly”
- OracleWebEventProvider Class

7.1.2 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”
- OracleWebEventProvider Class
- OracleWebEventProvider Members

7.1.2.1 OracleWebEventProvider()

This constructor creates an instance of the OracleWebEventProvider class.

Declaration

```csharp
// C#
public OracleWebEventProvider();
```

Remarks

This constructor creates a new instance of the OracleWebEventProvider class.
7.1.3 OracleWebEventProvider Static Methods

The `OracleWebEventProvider` static methods are listed in Table 7-5.

<table>
<thead>
<tr>
<th>Static Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equals</td>
<td>Inherited from <code>System.Object</code> (Overloaded)</td>
</tr>
<tr>
<td>ReferenceEquals</td>
<td>Inherited from <code>System.Object</code></td>
</tr>
</tbody>
</table>

7.1.4 OracleWebEventProvider Public Properties

The `OracleWebEventProvider` public properties are listed in Table 7-6.

<table>
<thead>
<tr>
<th>Public Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BufferMode</td>
<td>Inherited from <code>System.Web.Management.BufferedWebEventProvider</code></td>
</tr>
<tr>
<td>CommandTimeout</td>
<td>Gets the number of seconds that the command is allowed to execute before it</td>
</tr>
<tr>
<td></td>
<td>is terminated with an exception</td>
</tr>
<tr>
<td>Description</td>
<td>Inherited from <code>System.Configuration.Provider.ProviderBase</code></td>
</tr>
<tr>
<td>Name</td>
<td>Inherited from <code>System.Configuration.Provider.ProviderBase</code></td>
</tr>
</tbody>
</table>
7.1.4.1 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.

7.1.5 OracleWebEventProvider Public Methods

The OracleWebEventProvider public methods are listed in Table 7-7.

Table 7-7  OracleWebEventProvider Public Methods

<table>
<thead>
<tr>
<th>Public Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initialize</td>
<td>Initializes the OracleWebEventProvider instance with the property values specified in the ASP.NET application configuration file</td>
</tr>
<tr>
<td>ProcessEvent</td>
<td>Processes the event passed to it as an argument</td>
</tr>
</tbody>
</table>
Table 7-7  (Cont.) OracleWebEventProvider Public Methods

<table>
<thead>
<tr>
<th>Public Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ProcessEventFlush</td>
<td>Flushes the information passed to it as an argument</td>
</tr>
<tr>
<td>Shutdown</td>
<td>Releases all resources</td>
</tr>
<tr>
<td>Flush</td>
<td>Inherited from System.BufferedWebEventProvider</td>
</tr>
<tr>
<td>Equals(Overloaded)</td>
<td>Inherited from System.Object</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Inherited from System.Object</td>
</tr>
<tr>
<td>GetType</td>
<td>Inherited from System.Object</td>
</tr>
<tr>
<td>ToString</td>
<td>Inherited from System.Object</td>
</tr>
</tbody>
</table>

See Also:
- "Oracle Providers for ASP.NET Assembly"
- OracleWebEventProvider Class
- OracleWebEventProvider Members

7.1.5.1 Initialize

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

Declaration

```csharp
// 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”
• OracleWebEventProvider Class
• OracleWebEventProvider Members

7.1.5.2 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”
• OracleWebEventProvider Class
• OracleWebEventProvider Members

7.1.5.3 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"
- OracleWebEventProvider Class
- OracleWebEventProvider Members

7.1.5.4 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"
- OracleWebEventProvider Class
- OracleWebEventProvider Members
This chapter describes the OraclePersonalizationProvider class.

See Also:

- ASP.NET Web Parts [http://msdn.microsoft.com/en-us/library/e0s9t4ck.aspx]

This chapter contains the following topic:

- OraclePersonalizationProvider Class

8.1 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


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
<?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
<?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
8.1.1 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

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OraclePersonalizationProvider Constructors</td>
<td>Instantiates a new instance of the OraclePersonalizationProvider class</td>
</tr>
</tbody>
</table>

OraclePersonalizationProvider Static Methods

OraclePersonalizationProvider static methods are listed in Table 8-2.

Table 8-2  OraclePersonalizationProvider Static Methods

<table>
<thead>
<tr>
<th>Static Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equals</td>
<td>Inherited from System.Object</td>
</tr>
<tr>
<td>ReferenceEquals</td>
<td>Inherited from System.Object</td>
</tr>
</tbody>
</table>

OraclePersonalizationProvider Public Properties

OraclePersonalizationProvider public properties are listed in Table 8-3.

Table 8-3  OraclePersonalizationProvider Public Properties

<table>
<thead>
<tr>
<th>Public Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ApplicationName</td>
<td>Gets or sets the name of the application that is used to specify personalization information specific to an application</td>
</tr>
<tr>
<td>CommandTimeout</td>
<td>Gets the number of seconds that the command is allowed to execute before it is terminated with an exception</td>
</tr>
</tbody>
</table>
Table 8-3  (Cont.) OraclePersonalizationProvider Public Properties

<table>
<thead>
<tr>
<th>Public Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>Inherited from System.Configuration.Provider.Providerbase</td>
</tr>
<tr>
<td>Name</td>
<td>Inherited from System.Configuration.Provider.Providerbase</td>
</tr>
</tbody>
</table>

OraclePersonalizationProvider Public Methods

OraclePersonalizationProvider public methods are listed in Table 8-4.

Table 8-4  OraclePersonalizationProvider Public Methods

<table>
<thead>
<tr>
<th>Public Methods</th>
<th>Description</th>
</tr>
</thead>
</table>
| FindState                      | Returns a collection containing zero or more Persona-  
|                                | lizationStateInfo-derived objects based on scope and specific query parameters |
| GetCountOfState                | Returns the number of rows in the underlying Oracle database that are within the specified scope |
| Initialize                     | Initializes the Oracle personalization provider                                |
| Equals                         | Inherited from System.Object (Overloaded)                                      |
| ResetState                     | Deletes personalization state information from the underlying data source, based on the specified parameters |
| ResetUserState                 | 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"
- OraclePersonalizationProvider Class
8.1.2 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”
- OraclePersonalizationProvider Class
- OraclePersonalizationProvider Members

8.1.2.1 OraclePersonalizationProvider()

This constructor creates an instance of the OraclePersonalizationProvider class.

Declaration

```csharp
// 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”
- OraclePersonalizationProvider Class
- OraclePersonalizationProvider Members

8.1.3 OraclePersonalizationProvider Static Methods

The OraclePersonalizationProvider static methods are listed in Table 8-5.
Table 8-5 OraclePersonalizationProvider Static Methods

<table>
<thead>
<tr>
<th>Static Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equals</td>
<td>Inherited from System.Object</td>
</tr>
<tr>
<td>ReferenceEquals</td>
<td>Inherited from System.Object</td>
</tr>
</tbody>
</table>

See Also:
- "Oracle Providers for ASP.NET Assembly"
- OraclePersonalizationProvider Class
- OraclePersonalizationProvider Members

8.1.4 OraclePersonalizationProvider Public Properties

The OraclePersonalizationProvider public properties are listed in Table 8-6.

Table 8-6 OraclePersonalizationProvider Public Properties

<table>
<thead>
<tr>
<th>Public Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ApplicationName</td>
<td>Gets or sets the name of the application that is used to specify personalization information specific to an application</td>
</tr>
<tr>
<td>CommandTimeout</td>
<td>Gets the number of seconds that the command is allowed to execute before it is terminated with an exception</td>
</tr>
<tr>
<td>Description</td>
<td>Inherited from System.Configuration.Provider.Providerbase</td>
</tr>
<tr>
<td>Name</td>
<td>Inherited from System.Configuration.Provider.Providerbase</td>
</tr>
</tbody>
</table>

See Also:
- "Oracle Providers for ASP.NET Assembly"
- OraclePersonalizationProvider Class
- OraclePersonalizationProvider Members

8.1.4.1 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”
- `OraclePersonalizationProvider` Class
- `OraclePersonalizationProvider` Members
8.1.4.2 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"
- OraclePersonalizationProvider Class
- OraclePersonalizationProvider Members

8.1.5 OraclePersonalizationProvider Public Methods

The OraclePersonalizationProvider public methods are listed in Table 8-7.

Table 8-7  OraclePersonalizationProvider Public Methods

<table>
<thead>
<tr>
<th>Public Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FindState</td>
<td>Returns a collection containing zero or more PersonalizationStateInfo-derived objects based on scope and specific query parameters</td>
</tr>
<tr>
<td>GetCountOfState</td>
<td>Returns the number of rows in the underlying Oracle database that are within the specified scope</td>
</tr>
<tr>
<td>Initialize</td>
<td>Initializes the Oracle personalization provider</td>
</tr>
<tr>
<td>Equals</td>
<td>Inherited from System.Object (Overloaded)</td>
</tr>
<tr>
<td>ResetState</td>
<td>Deletes personalization state information from the underlying data source, based on the specified parameters</td>
</tr>
</tbody>
</table>
Table 8-7  (Cont.) OraclePersonalizationProvider Public Methods

<table>
<thead>
<tr>
<th>Public Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ResetUserState</td>
<td>Deletes user personalization data from the underlying Oracle data source,</td>
</tr>
<tr>
<td></td>
<td>based on the specified parameters</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Inherited from System.Object</td>
</tr>
<tr>
<td>GetType</td>
<td>Inherited from System.Object</td>
</tr>
<tr>
<td>ToString</td>
<td>Inherited from System.Object</td>
</tr>
<tr>
<td>SavePersonalizationState</td>
<td>Inherited from System.Web.UI.WebControls.WebParts.PersonalizationProvider</td>
</tr>
<tr>
<td>DetermineInitialScope</td>
<td>Inherited from System.Web.UI.WebControls.WebParts.PersonalizationProvider</td>
</tr>
<tr>
<td>DetermineUserCapabilities</td>
<td>Inherited from System.Web.UI.WebControls.WebParts.PersonalizationProvider</td>
</tr>
</tbody>
</table>

See Also:

- “Oracle Providers for ASP.NET Assembly”
- OraclePersonalizationProvider Class
- OraclePersonalizationProvider Members

8.1.5.1 FindState

This method returns a collection containing zero or more `PersonalizationStateInfo`-derived objects based on scope and specific query parameters.

Declaration

```csharp
// 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.
  - \((\text{pageIndex} \times \text{pageSize} + \text{pageSize}) - 1\) 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 `Last ActivityDate` property is less than or equal to the `User Inactive Since Date` 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”
- OraclePersonalizationProvider Class
- OraclePersonalizationProvider Members

8.1.5.2 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 Path-
ToMatch 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"
- OraclePersonalizationProvider Class
- OraclePersonalizationProvider Members

8.1.5.3 Initialize

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

Declaration

```csharp
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”
- OraclePersonalizationProvider Class
- OraclePersonalizationProvider Members

8.1.5.4 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"
- OraclePersonalizationProvider Class
- OraclePersonalizationProvider Members

### 8.1.5.5 ResetUserState

This method deletes user personalization data from the underlying Oracle data source, based on the specified parameters.

**Declaration**

```csharp
// 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”
- `OraclePersonalizationProvider` Class
- `OraclePersonalizationProvider Members`
This chapter describes the OracleCacheDependency class.

See Also:


This chapter contains the following topic:

• OracleCacheDependency Class

9.1 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


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"
- OracleCacheDependency Members
- OracleCacheDependency Constructors
- OracleCacheDependency Properties
- OracleCacheDependency Methods

9.1.1 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

<table>
<thead>
<tr>
<th>Constructor</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OracleCacheDependency Constructors</td>
<td>Instantiates a new instance of the OracleCacheDependency class</td>
</tr>
</tbody>
</table>

OracleCacheDependency Properties

OracleCacheDependency properties are listed in Table 9-2.

Table 9-2 OracleCacheDependency Properties

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HasChanged</td>
<td>Inherited from System.CacheDependency</td>
</tr>
<tr>
<td>UtcLastModified</td>
<td>Inherited from System.CacheDependency</td>
</tr>
</tbody>
</table>

OracleCacheDependency Methods

OracleCacheDependency methods are listed in Table 9-3.
Table 9-3  OracleCacheDependency Methods

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dispose</td>
<td>Inherited from System.Object</td>
</tr>
<tr>
<td>Equals</td>
<td>Inherited from System.Object (Overloaded)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Inherited from System.Object</td>
</tr>
<tr>
<td>GetType</td>
<td>Inherited from System.Object</td>
</tr>
<tr>
<td>GetUniqueID</td>
<td>Returns a unique identifier for the OracleCacheDependency object</td>
</tr>
<tr>
<td>ReferenceEquals</td>
<td>Inherited from System.Object</td>
</tr>
<tr>
<td>ToString</td>
<td>Inherited from System.Object</td>
</tr>
</tbody>
</table>

See Also:

- “Oracle Providers for ASP.NET Assembly”
- OracleCacheDependency Class

9.1.2 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”
- OracleCacheDependency Class
- OracleCacheDependency Members

9.1.2.1 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"
- OracleCacheDependency Class
- OracleCacheDependency Members

9.1.3 OracleCacheDependency Properties

OracleCacheDependency properties are listed in Table 9-4.

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HasChanged</td>
<td>Inherited from System.CacheDependency</td>
</tr>
<tr>
<td>UtcLastModified</td>
<td>Inherited from System.CacheDependency</td>
</tr>
</tbody>
</table>

See Also:

- "Oracle Providers for ASP.NET Assembly"
- OracleCacheDependency Class
- OracleCacheDependency Members

9.1.4 OracleCacheDependency Methods

OracleCacheDependency methods are listed in Table 9-5.
### Table 9-5  OracleCacheDependency Methods

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dispose</td>
<td>Inherited from System.Object</td>
</tr>
<tr>
<td>Equals</td>
<td>Inherited from System.Object (Overloaded)</td>
</tr>
<tr>
<td>GetHashCode</td>
<td>Inherited from System.Object</td>
</tr>
<tr>
<td>GetType</td>
<td>Inherited from System.Object</td>
</tr>
<tr>
<td>GetUniqueID</td>
<td>Returns a unique identifier for the OracleCacheDependency object</td>
</tr>
<tr>
<td>ReferenceEquals</td>
<td>Inherited from System.Object</td>
</tr>
<tr>
<td>ToString</td>
<td>Inherited from System.Object</td>
</tr>
</tbody>
</table>

#### See Also:
- “Oracle Providers for ASP.NET Assembly”
- OracleCacheDependency Class
- OracleCacheDependency Members

---

#### 9.1.4.1 GetUniqueID

This method returns a string that uniquely identifies the OracleCacheDependency object.

**Declaration**

```csharp
// C#
public override string GetUniqueID()
```

#### See Also:
- “Oracle Providers for ASP.NET Assembly”
- OracleCacheDependency Class
- OracleCacheDependency Members
Index

A
ASP.NET provider model, 1-1

C
client configuration, 1-10
configuration, 1-10
configuration file, 1-1
configuration scripts, 1-8

D
documentation, 1-12

F
file locations, 1-12

I
install scripts, 1-8
installation, 1-5
machine.config, 1-10
InstallOracleASPNETCommon.sql, 1-8
Instant Client, 1-5

M
Microsoft ASP.NET 2.0, 1-1

O
object references, 1-13
objects, 1-13
Oracle Data Provider for .NET
system requirements, 1-5
Oracle Database, 1-1
Oracle Providers for ASP.NET Assembly, 1-4
Oracle Universal Installer, 1-5
Oracle.Web.dll, 1-4, 1-12
OracleCacheDependency Class
class description, 9-1
constructors, 9-3

OracleCacheDependency Class (continued)
members, 9-2
methods, 9-4
properties, 9-4

OracleMembershipProvider Class
class description, 2-1
constructors, 2-6
members, 2-3
public methods, 2-18, 2-39
public properties, 2-7
static methods, 2-7

OraclePersonalizationProvider Class
class description, 8-1
constructors, 8-5
members, 8-3
public methods, 8-8
public properties, 8-6
static methods, 8-5

OracleProfileProvider Class
class description, 6-1
constructors, 6-5
members, 6-3
public methods, 6-8
public properties, 6-6
static methods, 6-6

OracleRoleProvider Class, 3-1
class description, 3-1
constructors, 3-4
members, 3-3
public methods, 3-8
public properties, 3-6
static methods, 3-5

OracleRoleProvider Member
OracleRoleProvider Constructor, 7-3, 8-3
OracleRoleProvider Members, 3-3
OracleRoleProvider Public Properties, 7-3, 8-3
OracleRoleProvider Static Methods, 7-3, 8-3

OracleSessionStateStore Class
class description, 5-1
constructors, 5-4
members, 5-2
public methods, 5-6
public properties, 5-5

OracleSiteMapProvider Class
class description, 4-1
OracleSiteMapProvider Class (continued)
constructors, 4-5
members, 4-3
public methods, 4-8
public properties, 4-6
static methods, 4-6
OracleWebEventProvider Class
class description, 7-1
constructors, 7-5
members, 7-3
public methods, 7-7
public properties, 7-6
static methods, 7-6
OraProvCfg, 1-10, 1-12

P
passwords in code examples, xiii
privileges
  granting, 1-8

R
roles, 1-14

S
schema objects, 1-13
  roles, 1-14
  stored procedures, 1-16
  synonyms, 1-22
  tables, 1-13
  views, 1-15
SQL scripts, 1-12
stored procedures, 1-16
synonyms, 1-22
system requirements
  Oracle Data Provider for .NET, 1-5

T
tables, 1-13

V
views, 1-15

X
xcopy Instant Client, 1-5