# Infor VISUAL API Toolkit Core Class Library Reference

### Publication Information

Release: Infor VISUAL

Publication date: August 13, 2024

# About this guide

This guide describes the objects available for use in the Infor VISUAL API Core library.

**NOTE:** This class library exposes classes and methods that are not compatible with the VISUAL API Toolkit. Only the classes and methods specifically described in this document are compatible. The use of any class or method that is not described in this document is not supported.

# Intended audience

The intended audience of this guide is developers who are using the API Toolkit to extend the VISUAL solution.

# Contacting Support

If you have questions about Infor products, go to the Infor Customer Portal at https://customerportal.infor.com/csmcore/

If we update this document after the product release, we will post the new version on this Web site. We recommend that you check this Web site periodically for updated documentation.

If you have comments about Infor documentation, contact https://docs.infor.com/en-us.

# Supported languages

These languages are supported for use with the toolkit:

- Visual Basic
- C#

While it is possible to use any .NET-aware programming language with the toolkit, other languages are not officially supported.

# Support information

The API Toolkit will be updated regularly as more class members are added to each assembly, schema changes are made, and any reported issues are resolved. Infor Support cannot assist you with developing customized code using the API Toolkit. For assistance with customizations, contact Infor Consulting Services or your channel partner.

The functionality provided within the API Toolkit will not be extended beyond the standard functionality experienced in the VISUAL application itself. Enhancement requests with compelling business cases detailing how suggested alternatives are not viable will be evaluated and considered.

Infor is not responsible for data incorrectly entered to the database through the use of the API Toolkit. Customers must establish a full test environment to ensure that data created by APIs functions in the same manner as data created through the VISUAL interface.

# Lsa.Data Namespace

## Classes

| | Class | Description |
|---|---|---|
|  | Dbms | Static class containing major entry points to access data in all databases. Dbms acts more or less as the factory for all other object types in the system. |

# Dbms Class

Static class containing major entry points to access data in all databases. Dbms acts more or less as the factory for all other object types in the system.

## Inheritance Hierarchy

System.Object
  Lsa.Data.Dbms

**Namespace:** Lsa.Data
**Assembly:** LsaCore (in LsaCore.dll) Version: 8.1.0.0 (8.1.0.0)

## Syntax

**C#**

```
public sealed class Dbms
```

**VB**

```
Public NotInheritable Class Dbms
```

The **Dbms** type exposes the following members.

## Methods

| | Name | Description |
|---|---|---|
|  | Close | Closes the named database. |
|  | CloseAll | Closes all open databases. |
|  | CompareDataspaceVersions | Compare the existing dataspace version to the argument. Existing dataspace version is the left hand argument, and compareVersion is the right hand argument in the comparison. |

| | | |
|---|---|---|
|  | DatabaseName | Gets the database name of the named instance. For Visual Manufacturing databases, this is the second parameter in the data source. |
|  | DeleteNextNumber | Delete the next number control record for the specified column in the named database instance. |
|  | GetInstanceInfo | Returns instance information for the named instance. |
|  | GetNextNumber | Retrieve the next number based on the current control values. Use this when showing current values and what the next number might be. |
|  | GetNextNumberAndAdvance | Retrieve the next number based on the current control values. Write the next control value (next number) back, but do not commit the change. Use with business logic when you need a new number. |
|  | GetSetting | Get the value of the named setting in the named instance. Settings are like environment or registry values except they may contain arbitrary string data up to 2GB and are stored directly in the database. |
|  | InstanceIsOpen | Determines if a named database instance is currently open. |
|  | OpenDirect(String, String, String, String, String, String) | Open the database in client (local) mode using direct values rather than looking for connection values in the Database.Config file. An already open instance is not reopened unless the user is changing. |
|  | OpenDirect(String, String, String, String, String, String, String, String) | Open the database in client (local) mode using direct values rather than looking for connection values in the Database.Config file. An already open instance is not reopened unless the user is changing. |
|  | OpenLocal(String, String, String) | Opens a database in client (local) mode. This is the recommended method of opening a database. An already open instance is not reopened unless the user ID is changing. Connection information is obtained from the Database.Config file. |
|  | OpenLocal(String, String, String, String) | Opens a database in client (local) mode. This is the recommended method of opening a database. An already open instance is not |

| | | |
|---|---|---|
| | | reopened unless the user ID is changing. Connection information is obtained from the Database.Config file. |
|  | OwnerPassword | Return the owner password of the named instance. You must have code authority to call this method. |
|  | OwnerUserID | Return the owner user ID of the named instance. You must have code authority to call this method. |
|  | ServerName | Gets the server name of the named database instance. |
|  | SetNextNumber | Set the next number generation control values for the specified column in the named instance. Next number generation is performed by the core classes so it is uniform for all applications. |
|  | SetSetting | Set the value of the named setting in the named instance. Settings are like environment or registry values except they may contain arbitrary string data up to 2GB and are stored directly in the database. Be sure the setting can be down converted from a string. If the setting value is null or blank, the setting entry is deleted. |
|  | Settings | Collection of settings for the named instance. Settings can be large. You will typically use GetSetting() instead. |
|  | UserID | Gets the user ID that currently has the named database instance opened. |

## See Also

Lsa.Data Namespace

# Dbms.Dbms Methods

The Dbms type exposes the following members.

## Methods

| | Name | Description |
|---|---|---|
|  | Close | Closes the named database. |
|  | CloseAll | Closes all open databases. |
|  | CompareDataspaceVersions | Compare the existing dataspace version to the argument. Existing dataspace version is the left hand argument, and compareVersion is the right hand argument in the comparison. |
|  | DatabaseName | Gets the database name of the named instance. For Visual Manufacturing databases, this is the second parameter in the data source. |
|  | DeleteNextNumber | Delete the next number control record for the specified column in the named database instance. |
|  | GetInstanceInfo | Returns instance information for the named instance. |
|  | GetNextNumber | Retrieve the next number based on the current control values. Use this when showing current values and what the next number might be. |
|  | GetNextNumberAndAdvance | Retrieve the next number based on the current control values. Write the next control value (next number) back, but do not commit the change. Use with business logic when you need a new number. |
|  | GetSetting | Get the value of the named setting in the named instance. Settings are like environment or registry values except they may contain arbitrary string data up to 2GB and are stored directly in the database. |
|  | InstanceIsOpen | Determines if a named database instance is currently open. |

| | | |
|---|---|---|
|  | OpenDirect(String, String, String, String, String, String) | Open the database in client (local) mode using direct values rather than looking for connection values in the Database.Config file. An already open instance is not reopened unless the user is changing. |
|  | OpenDirect(String, String, String, String, String, String, String, String) | Open the database in client (local) mode using direct values rather than looking for connection values in the Database.Config file. An already open instance is not reopened unless the user is changing. |
|  | OpenLocal(String, String, String) | Opens a database in client (local) mode. This is the recommended method of opening a database. An already open instance is not reopened unless the user ID is changing. Connection information is obtained from the Database.Config file. |
|  | OpenLocal(String, String, String, String) | Opens a database in client (local) mode. This is the recommended method of opening a database. An already open instance is not reopened unless the user ID is changing. Connection information is obtained from the Database.Config file. |
|  | OwnerPassword | Return the owner password of the named instance. You must have code authority to call this method. |
|  | OwnerUserID | Return the owner user ID of the named instance. You must have code authority to call this method. |
|  | ServerName | Gets the server name of the named database instance. |
|  | SetNextNumber | Set the next number generation control values for the specified column in the named instance. Next number generation is performed by the core classes so it is uniform for all applications. |
|  | SetSetting | Set the value of the named setting in the named instance. Settings are like environment or registry values except they may contain arbitrary string data up to 2GB and are stored directly in the database. Be sure the setting can be down converted from a string. If the setting value is null or blank, the setting entry is deleted. |

| | | |
|---|---|---|
|  | Settings | Collection of settings for the named instance. Settings can be large. You will typically use GetSetting() instead. |
|  | UserID | Gets the user ID that currently has the named database instance opened. |

## See Also

- Dbms Class
- Lsa.Data Namespace

# Dbms.Close Method

Closes the named database.

**Namespace:** Lsa.Data
**Assembly:** LsaCore (in LsaCore.dll) Version: 8.1.0.0 (8.1.0.0)

## Syntax

**C#**

```
public static bool Close(
    string instanceName
)
```

**VB**

```
Public Shared Function Close (
    instanceName As String
) As Boolean
```

### Parameters

*instanceName*

Type: System.String

Name of the database instance to close.

### Return Value

Type: Boolean
Boolean success or failure

## See Also

Dbms Class

Lsa.Data Namespace

# Dbms.CloseAll Method

Closes all open databases.

**Namespace:** Lsa.Data
**Assembly:** LsaCore (in LsaCore.dll) Version: 8.1.0.0 (8.1.0.0)

## Syntax

**C#**

```
public static void CloseAll()
```

**VB**

```
Public Shared Sub CloseAll
```

## See Also

Dbms Class

Lsa.Data Namespace

# Dbms.CompareDataspaceVersions Method

Compare the existing dataspace version to the argument. Existing dataspace version is the left hand argument, and compareVersion is the right hand argument in the comparison.

**Namespace:** Lsa.Data
**Assembly:** LsaCore (in LsaCore.dll) Version: 8.1.0.0 (8.1.0.0)

## Syntax

**C#**

```
public static int CompareDataspaceVersions(
    string version1,
    string version2
)
```

**VB**

```
Public Shared Function CompareDataspaceVersions (
    version1 As String,
    version2 As String
) As Integer
```

### Parameters

*version1*

Type: System.String

First version to compare

*version2*

Type: System.String

Second version to compare

### Return Value

Type: Int32
Zero, less than zero, or greater than zero.

## Remarks

Zero is returned if version 1 and version 2 are the same version. Negative 1 is returned if version 1 precedes version 2. Positive 1 is returned if version 1 succeeds version 2.

## See Also

Dbms Class

Lsa.Data Namespace

# Dbms.DatabaseName Method

Gets the database name of the named instance. For Visual Manufacturing databases, this is the second parameter in the data source.

**Namespace:** Lsa.Data
**Assembly:** LsaCore (in LsaCore.dll) Version: 8.1.0.0 (8.1.0.0)

## Syntax

**C#**

```
public static string DatabaseName(
    string instanceName
)
```

**VB**

```
Public Shared Function DatabaseName (
    instanceName As String
) As String
```

### Parameters

*instanceName*

Type: System.String

Name of instance to test.

### Return Value

Type: String
Server Name/Database Name

## See Also

Dbms Class

Lsa.Data Namespace

# Dbms.DeleteNextNumber Method

Delete the next number control record for the specified column in the named database instance.

**Namespace:** Lsa.Data
**Assembly:** LsaCore (in LsaCore.dll) Version: 8.1.0.0 (8.1.0.0)

## Syntax

**C#**

```
public static void DeleteNextNumber(
    string instanceName,
    string dataspaceName,
    string tableName,
    string columnName,
    string context
)
```

**VB**

```
Public Shared Sub DeleteNextNumber (
    instanceName As String,
    dataspaceName As String,
    tableName As String,
    columnName As String,
    context As String
)
```

### Parameters

*instanceName*

Type: System.String

Instance name to process.

*dataspaceName*

Type: System.String

Dataspace containing table and column. Not used for VMFG databases.

*tableName*

Type: System.String

Table containing column.

*columnName*

Type: System.String

Column name. Typically a primary key.

*context*

Type: System.String

Context of numbering. Blank means GENERAL.

## See Also

Dbms Class

Lsa.Data Namespace

# Dbms.GetInstanceInfo Method

Returns instance information for the named instance.

**Namespace:** Lsa.Data
**Assembly:** LsaCore (in LsaCore.dll) Version: 8.1.0.0 (8.1.0.0)

## Syntax

**C#**

```
public static Instance GetInstanceInfo(
    string instanceName
)
```

**VB**

```
Public Shared Function GetInstanceInfo (
    instanceName As String
) As Instance
```

### Parameters

*instanceName*

Type: System.String

Name of the instance to return

### Return Value

Type: **Instance**
Instance information.

## Remarks

Using a blank name will return the first open instance.

## See Also

Dbms Class

Lsa.Data Namespace

# Dbms.GetNextNumber Method

Retrieve the next number based on the current control values. Use this when showing current values and what the next number might be.

**Namespace:** Lsa.Data
**Assembly:** LsaCore (in LsaCore.dll) Version: 8.1.0.0 (8.1.0.0)

## Syntax

**C#**

```
public static string GetNextNumber(
    string instanceName,
    string dataspaceName,
    string tableName,
    string columnName,
    string context,
    out string alphaPrefix,
    out string alphaSuffix,
    out int nextNumber,
    out bool leadingZeros,
    out short decimalPlaces
)
```

**VB**

```
Public Shared Function GetNextNumber (
    instanceName As String,
    dataspaceName As String,
    tableName As String,
    columnName As String,
    context As String,
    <OutAttribute> ByRef alphaPrefix As String,
    <OutAttribute> ByRef alphaSuffix As String,
    <OutAttribute> ByRef nextNumber As Integer,
    <OutAttribute> ByRef leadingZeros As Boolean,
    <OutAttribute> ByRef decimalPlaces As Short
) As String
```

### Parameters

*instanceName*

Type: System.String

Name of instance to process.

*dataspaceName*

Type: System.String

Dataspace name containing table and column. Not used for VMFG databases.

*tableName*

Type: System.String

Table name containing column.

*columnName*

Type: System.String

Column name. Typically a primary key.

*context*

Type: System.String

Context of numbering. Blank means GENERAL.

*alphaPrefix*

Type: System.String

Up to 4 character prefix on new numbers.

*alphaSuffix*

Type: System.String

Up to 4 character suffix on new numbers.

*nextNumber*

Type: System.Int32

Next available number. 4 to 9 digits long.

*leadingZeros*

Type: System.Boolean

True to request leading zeros on new numbers.

*decimalPlaces*

Type: System.Int16

Number of digits in new numbers. 4 to 9, inclusive.

### Return Value

Type: String
Next number

## See Also

Dbms Class

Lsa.Data Namespace

# Dbms.GetNextNumberAndAdvance Method

Retrieve the next number based on the current control values. Write the next control value (next number) back, but do not commit the change. Use with business logic when you need a new number.

**Namespace:** Lsa.Data
**Assembly:** LsaCore (in LsaCore.dll) Version: 8.1.0.0 (8.1.0.0)

## Syntax

**C#**

```
public static string GetNextNumberAndAdvance(
    string instanceName,
    string dataspaceName,
    string tableName,
    string columnName,
    string context,
    out string alphaPrefix,
    out string alphaSuffix,
    out int nextNumber,
    out bool leadingZeros,
    out short decimalPlaces
)
```

**VB**

```
Public Shared Function GetNextNumberAndAdvance (
    instanceName As String,
    dataspaceName As String,
    tableName As String,
    columnName As String,
    context As String,
    <OutAttribute> ByRef alphaPrefix As String,
    <OutAttribute> ByRef alphaSuffix As String,
    <OutAttribute> ByRef nextNumber As Integer,
    <OutAttribute> ByRef leadingZeros As Boolean,
    <OutAttribute> ByRef decimalPlaces As Short
) As String
```

### Parameters

*instanceName*

Type: System.String

Name of instance to process.

*dataspaceName*

Type: System.String

Dataspace name containing table and column. Not used for VMFG databases.

*tableName*

Type: System.String

Table name containing column.

*columnName*

Type: System.String

Column name. Typically a primary key.

*context*

Type: System.String

Context of numbering. Blank means GENERAL.

*alphaPrefix*

Type: System.String

Up to 4 character prefix on new numbers.

*alphaSuffix*

Type: System.String

Up to 4 character suffix on new numbers.

*nextNumber*

Type: System.Int32

Next available number. 4 to 9 digits long.

*leadingZeros*

Type: System.Boolean

True to request leading zeros on new numbers.

*decimalPlaces*

Type: System.Int16

Number of digits in new numbers. 4 to 9, inclusive.

## Return Value

Type: String
Next number

# Remarks

The publically accessible method Dbms.GetNextNumbertAndAdvance would not typically be called directly by an API toolkit developer.

To simplify the auto numbering process, the toolkit accepts any primary key ID value supplied with a character beginning with “<” and ending with “>” as an ID that should be auto numbered.

For example, the value “<AUTO>” would suffice.

The advantage of using this methodology is that the complexities of determining the scope of the numbering context, and what parameters to pass into Dbms.GetNextNumbertAndAdvance, are handled automatically, basically wrapping Dbms.GetNextNumbertAndAdvance.

A transaction is started when Save is executed on a toolkit object. When that save is completed, the transaction is committed. If the save fails, the transaction is rolled back.

Using the “<AUTO>” numbering scheme as outlined above, Dbms.GetNextNumbertAndAdvance is called during the save and thus is part of the transaction. This method holds a tight lock on the NEXT_NUMBER_GEN table in effect acting as a semaphore so no other caller can be issued a duplicate number. When the save is committed, the changes to the NEXT_NUMBER_GEN table are committed as well as any other tables that may have been updated as part of the transaction.

This example would cause an execution of Dbms.GetNextNumbertAndAdvance (internally)

```
Lsa.Vmfg.Sales.CustomerOrder CO = new
Lsa.Vmfg.Sales.CustomerOrder("VE900");

            Lsa.Data.DataRow drHeader = null;

            CO.Load("");

            drHeader = CO.NewOrderRow("<AUTO>");  // Triggers a call to
Dbms.GetNextNumbertAndAdvance when saved

            drHeader["CUSTOMER_ID"] = "NOCREDIT";

            drHeader["SITE_ID"] = "MMC";

CO.Save();
```

# See Also

Dbms Class

Lsa.Data Namespace

# Dbms.GetSetting Method

Get the value of the named setting in the named instance. Settings are like environment or registry values except they may contain arbitrary string data up to 2GB and are stored directly in the database.

**Namespace:** Lsa.Data
**Assembly:** LsaCore (in LsaCore.dll) Version: 8.1.0.0 (8.1.0.0)

## Syntax

**C#**

```
public static string GetSetting(
    string instanceName,
    string settingName
)
```

**VB**

```
Public Shared Function GetSetting (
    instanceName As String,
    settingName As String
) As String
```

### Parameters

*instanceName*

Type: System.String

Name of instance to read from.

*settingName*

Type: System.String

Name of setting

### Return Value

Type: String
Setting value as a string

# See Also

- Dbms Class
- Lsa.Data Namespace

# Dbms.InstanceIsOpen Method

Determines if a named database instance is currently open.

**Namespace:** Lsa.Data
**Assembly:** LsaCore (in LsaCore.dll) Version: 8.1.0.0 (8.1.0.0)

## Syntax

**C#**

```
public static bool InstanceIsOpen(
    string instanceName
)
```

**VB**

```
Public Shared Function InstanceIsOpen (
    instanceName As String
) As Boolean
```

### Parameters

*instanceName*

Type: System.String

Name of the database instance to check

### Return Value

Type: Boolean
Boolean true if instance is open, false if it is not.

### Return Value

Type: Boolean
True or false

## See Also

Dbms Class

Lsa.Data Namespace

# Dbms.OpenDirect Method

## Overload List

| | Name | Description |
|---|---|---|
|  | OpenDirect(String, String, String, String, String, String) | Open the database in client (local) mode using direct values rather than looking for connection values in the Database.Config file. An already open instance is not reopened unless the user is changing. |
|  | OpenDirect(String, String, String, String, String, String, String, String) | Open the database in client (local) mode using direct values rather than looking for connection values in the Database.Config file. An already open instance is not reopened unless the user is changing. |

## See Also

- Dbms Class
- Lsa.Data Namespace

# Dbms.OpenDirect Method (String, String, String, String, String, String)

Open the database in client (local) mode using direct values rather than looking for connection values in the Database.Config file. An already open instance is not reopened unless the user is changing.

**Namespace:** Lsa.Data
**Assembly:** LsaCore (in LsaCore.dll) Version: 8.1.0.0 (8.1.0.0)

## Syntax

**C#**

```
public static bool OpenDirect(
    string instanceName,
    string provider,
    string driver,
    string dataSource,
    string ownerUser,
    string ownerPassword
)
```

**VB**

```
Public Shared Function OpenDirect (
    instanceName As String,
    provider As String,
    driver As String,
    dataSource As String,
    ownerUser As String,
    ownerPassword As String
) As Boolean
```

### Parameters

*instanceName*

Type: System.String

The name of the database to be opened.

*provider*

Type: System.String

Internal name of provider. SQLSERVER and ORACLE supported.

*driver*

Type: System.String

Not used in this verison.

*dataSource*

Type: System.String

Identifies server/database combination.

*ownerUser*

Type: System.String

User ID of user that owns the database.

*ownerPassword*

Type: System.String

Password of user that owns the database.

### Return Value

Type: Boolean
Boolean success or failure

## See Also

Dbms Class

OpenDirect Overload

Lsa.Data Namespace

# Dbms.OpenDirect Method (String, String, String, String, String, String, String, String)

Open the database in client (local) mode using direct values rather than looking for connection values in the Database.Config file. An already open instance is not reopened unless the user is changing.

**Namespace:** Lsa.Data
**Assembly:** LsaCore (in LsaCore.dll) Version: 8.1.0.0 (8.1.0.0)

## Syntax

**C#**

```
public static void OpenDirect(
    string instanceName,
    string provider,
    string driver,
    string dataSource,
    string ownerUser,
    string ownerPassword,
    string loginUser,
    string loginPassword
)
```

**VB**

```
Public Shared Sub OpenDirect (
    instanceName As String,
    provider As String,
    driver As String,
    dataSource As String,
    ownerUser As String,
    ownerPassword As String,
    loginUser As String,
    loginPassword As String
)
```

### Parameters

*instanceName*

Type: System.String

The name of the database to be opened.

*provider*

Type: System.String

Internal name of provider. SQLSERVER and ORACLE supported.

*driver*

Type: System.String

Not used in this verison.

*dataSource*

Type: System.String

Identifies server/database combination.

*ownerUser*

Type: System.String

User ID of user that owns the database.User ID of user opening the database.

*ownerPassword*

Type: System.String

Password of user that owns the database.Password of user opening the database.

*loginUser*

Type: System.String

*loginPassword*

Type: System.String

### Return Value

Type:
Boolean success or failure

## See Also

Dbms Class

OpenDirect Overload

Lsa.Data Namespace

# Dbms.OpenLocal Method

## Overload List

| | Name | Description |
|---|---|---|
|  | OpenLocal(String, String, String) | Opens a database in client (local) mode. This is the recommended method of opening a database. An already open instance is not reopened unless the user ID is changing. Connection information is obtained from the Database.Config file. |
|  | OpenLocal(String, String, String, String) | Opens a database in client (local) mode. This is the recommended method of opening a database. An already open instance is not reopened unless the user ID is changing. Connection information is obtained from the Database.Config file. |

## See Also

- Dbms Class
- Lsa.Data Namespace

# Dbms.OpenLocal Method (String, String, String)

Opens a database in client (local) mode. This is the recommended method of opening a database. An already open instance is not reopened unless the user ID is changing. Connection information is obtained from the Database.Config file.

**Namespace:** Lsa.Data
**Assembly:** LsaCore (in LsaCore.dll) Version: 8.1.0.0 (8.1.0.0)

## Syntax

**C#**

```
public static bool OpenLocal(
    string instanceName,
    string loginUser,
    string loginPassword
)
```

**VB**

```
Public Shared Function OpenLocal (
    instanceName As String,
    loginUser As String,
    loginPassword As String
) As Boolean
```

### Parameters

*instanceName*

Type: System.String

Name of instance to be opened.

*loginUser*

Type: System.String

User ID opening the database.

*loginPassword*

Type: System.String

Password of user opening the database.

### Return Value

Type: Boolean
Boolean success or failure

## See Also

Dbms Class

OpenLocal Overload

Lsa.Data Namespace

# Dbms.OpenLocal Method (String, String, String, String)

Opens a database in client (local) mode. This is the recommended method of opening a database. An already open instance is not reopened unless the user ID is changing. Connection information is obtained from the Database.Config file.

**Namespace:** Lsa.Data
**Assembly:** LsaCore (in LsaCore.dll) Version: 8.1.0.0 (8.1.0.0)

## Syntax

**C#**

```
public static bool OpenLocal(
    string instanceName,
    string loginUser,
    string loginPassword,
    string databaseConfigPath
)
```

**VB**

```
Public Shared Function OpenLocal (
    instanceName As String,
    loginUser As String,
    loginPassword As String,
    databaseConfigPath As String
) As Boolean
```

### Parameters

*instanceName*

Type: System.String

Name of instance to be opened.

*loginUser*

Type: System.String

User ID opening the database.

*loginPassword*

Type: System.String

Password of user opening the database.

*databaseConfigPath*

Type: System.String

Full path to the Database.Config file.

### Return Value

Type: Boolean
Boolean success or failure

## See Also

Dbms Class

OpenLocal Overload

Lsa.Data Namespace

# Dbms.OwnerPassword Method

Return the owner password of the named instance. You must have code authority to call this method.

**Namespace:** Lsa.Data
**Assembly:** LsaCore (in LsaCore.dll) Version: 8.1.0.0 (8.1.0.0)

## Syntax

**C#**

```
public static string OwnerPassword(
    string instanceName
)
```

**VB**

```
Public Shared Function OwnerPassword (
    instanceName As String
) As String
```

### Parameters

*instanceName*

Type: System.String

Name of instance to test.

### Return Value

Type: String
Password

## Remarks

Note: This method is not supported for use with the Visual API Toolkit.

## See Also

Dbms Class

Lsa.Data Namespace

# Dbms.OwnerUserID Method

Return the owner user ID of the named instance. You must have code authority to call this method.

**Namespace:** Lsa.Data
**Assembly:** LsaCore (in LsaCore.dll) Version: 8.1.0.0 (8.1.0.0)

## Syntax

**C#**

```
public static string OwnerUserID(
    string instanceName
)
```

**VB**

```
Public Shared Function OwnerUserID (
    instanceName As String
) As String
```

### Parameters

*instanceName*

Type: System.String

Name of instance to test.

### Return Value

Type: String
User ID

## Remarks

Note: This method is not supported for use with the Visual API Toolkit.

## See Also

Dbms Class

Lsa.Data Namespace

# Dbms.ServerName Method

Gets the server name of the named database instance.

**Namespace:** Lsa.Data
**Assembly:** LsaCore (in LsaCore.dll) Version: 8.1.0.0 (8.1.0.0)

## Syntax

**C#**

```
public static string ServerName(
    string instanceName
)
```

**VB**

```
Public Shared Function ServerName (
    instanceName As String
) As String
```

### Parameters

*instanceName*

Type: System.String

Name of database instance to test.

### Return Value

Type: String
Server name

## See Also

Dbms Class

Lsa.Data Namespace

# Dbms.SetNextNumber Method

Set the next number generation control values for the specified column in the named instance. Next number generation is performed by the core classes so it is uniform for all applications.

**Namespace:** Lsa.Data
**Assembly:** LsaCore (in LsaCore.dll) Version: 8.1.0.0 (8.1.0.0)

## Syntax

**C#**

```
public static void SetNextNumber(
    string instanceName,
    string dataspaceName,
    string tableName,
    string columnName,
    string context,
    string alphaPrefix,
    string alphaSuffix,
    int nextNumber,
    bool leadingZeros,
    short decimalPlaces
)
```

**VB**

```
Public Shared Sub SetNextNumber (
    instanceName As String,
    dataspaceName As String,
    tableName As String,
    columnName As String,
    context As String,
    alphaPrefix As String,
    alphaSuffix As String,
    nextNumber As Integer,
    leadingZeros As Boolean,
    decimalPlaces As Short
)
```

### Parameters

*instanceName*

Type: System.String

Name of the database instance to write to.

*dataspaceName*

Type: System.String

Dataspace name containing table and column. Not used for VMFG databases

*tableName*

Type: System.String

Table name containing column.

*columnName*

Type: System.String

Column name. Typically a primary key.

*context*

Type: System.String

Context of numbering. Blank means GENERAL.

*alphaPrefix*

Type: System.String

Up to 4 character prefix on new numbers.

*alphaSuffix*

Type: System.String

Up to 4 character suffix on new numbers.

*nextNumber*

Type: System.Int32

Next available number. 4 to 9 digits long.

*leadingZeros*

Type: System.Boolean

True to request leading zeros on new numbers.

*decimalPlaces*

Type: System.Int16

Number of digits in new numbers. 4 to 9, inclusive.

# See Also

Dbms Class

Lsa.Data Namespace

# Dbms.SetSetting Method

Set the value of the named setting in the named instance. Settings are like environment or registry values except they may contain arbitrary string data up to 2GB and are stored directly in the database. Be sure the setting can be down converted from a string. If the setting value is null or blank, the setting entry is deleted.

**Namespace:** Lsa.Data
**Assembly:** LsaCore (in LsaCore.dll) Version: 8.1.0.0 (8.1.0.0)

## Syntax

**C#**

```
public static void SetSetting(
    string instanceName,
    string settingName,
    string settingValue
)
```

**VB**

```
Public Shared Sub SetSetting (
    instanceName As String,
    settingName As String,
    settingValue As String
)
```

### Parameters

*instanceName*

Type: System.String

Name of instance to write to.

*settingName*

Type: System.String

Name of setting.

*settingValue*

Type: System.String

Setting value converted to a string.

# See Also

- Dbms Class
- Lsa.Data Namespace

# Dbms.Settings Method

Collection of settings for the named instance. Settings can be large. You will typically use GetSetting() instead.

**Namespace:** Lsa.Data
**Assembly:** LsaCore (in LsaCore.dll) Version: 8.1.0.0 (8.1.0.0)

## Syntax

**C#**

```
public static Settings Settings(
    string instanceName
)
```

**VB**

```
Public Shared Function Settings (
    instanceName As String
) As Settings
```

### Parameters

*instanceName*

Type: System.String

Name of instance to return.

### Return Value

Type: **Settings**
Setting collection

## See Also

Dbms Class

Lsa.Data Namespace

# Dbms.UserID Method

Gets the user ID that currently has the named database instance opened.

**Namespace:** Lsa.Data
**Assembly:** LsaCore (in LsaCore.dll) Version: 8.1.0.0 (8.1.0.0)

## Syntax

**C#**

```
public static string UserID(
    string instanceName
)
```

**VB**

```
Public Shared Function UserID (
    instanceName As String
) As String
```

### Parameters

*instanceName*

Type: System.String

Name of database instance to test.

### Return Value

Type: String
User ID

## See Also

Dbms Class

Lsa.Data Namespace