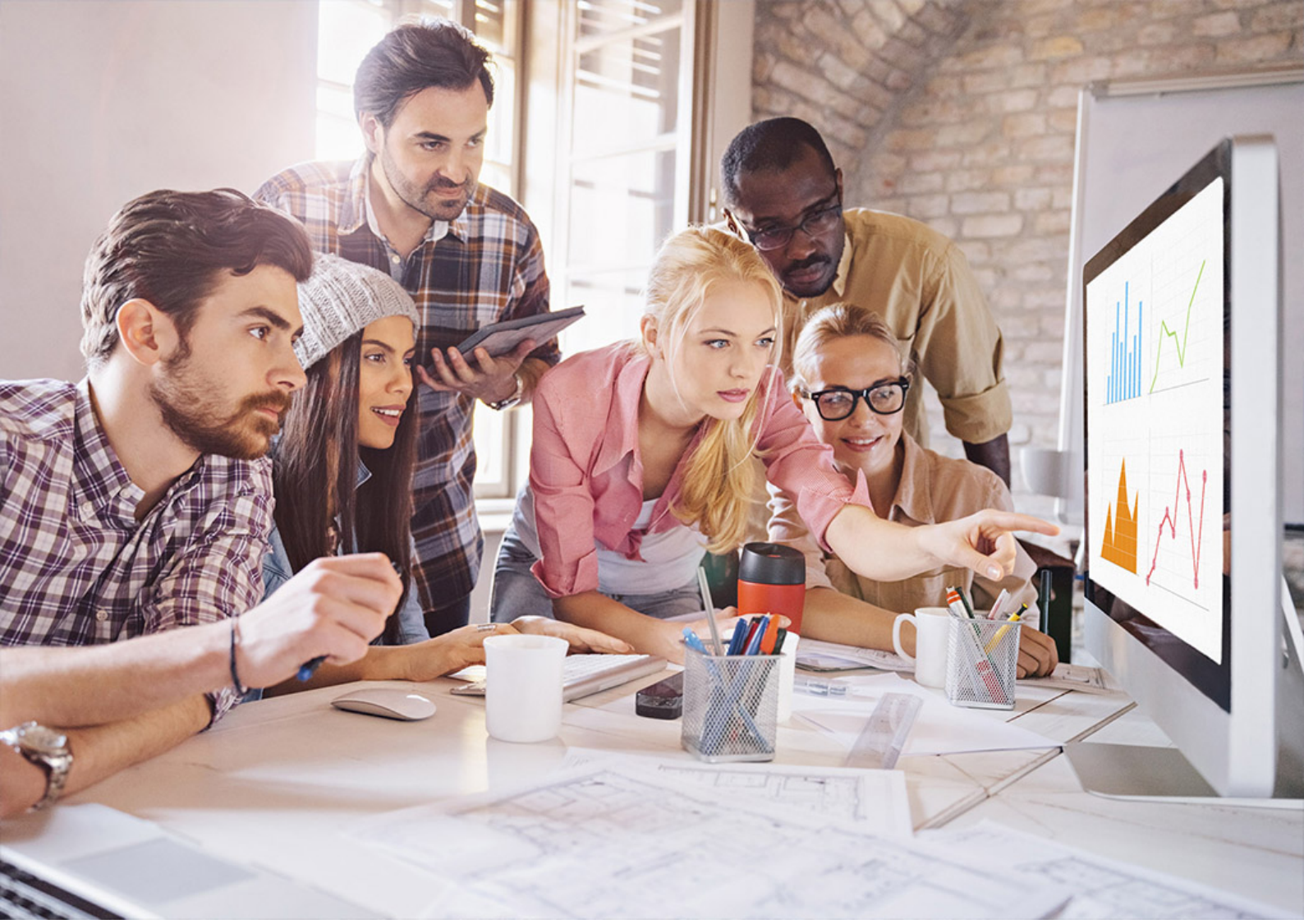

PUBLIC
SQL Anywhere Server
Document Version: 17.01.0 – 2021-10-15

SQL Anywhere - .NET API Reference

THE BEST RUN SAP

Content

1 SQL Anywhere .NET API Reference

Use SQL Anywhere with .NET, including the API for the SQL Anywhere .NET Data Provider.

Namespace

- `Sap.Data.SQLAnywhere`
- `Sap.SQLAnywhere.Server`

In this section:

Represents a set of commands and a database connection used to fill a System.Data.DataSet and to update a database.

SADataReader Class [page 205]
A read-only, forward-only result set from a query or stored procedure.

SADataSourceEnumerator Class [page 242]
Provides a mechanism for enumerating all available instances of database servers within the local network.

SADefault Class [page 244]
Represents a parameter with a default value.

SAError Class [page 244]
Collects information relevant to a warning or error returned by the data source.

SAErrorCollection Class [page 247]
Collects all errors generated by the .NET Data Provider.

SAException Class [page 251]
The exception that is thrown when the database server returns a warning or error.

SAFactory Class [page 254]
Represents a set of methods for creating instances of the Sap.Data.SQLAnywhere provider's implementation of the data source classes.

SAInfoMessageEventArgs Class [page 262]
Provides data for the InfoMessage event.

SAMetaDataCollectionNames Class [page 266]
Provides a list of constants for use with the SAConnection.GetSchema(string) method to retrieve metadata collections.

SAParameter Class [page 268]
Represents a parameter to an SACommand, and optionally, its mapping to a DataSet column.

SAParameterCollection Class [page 284]
Represents all parameters to an SACommand object and, optionally, their mapping to a DataSet column.

SAPermission Class [page 313]
Enables the .NET Data Provider to ensure that a user has a security level adequate to access a data source.

SAPermissionAttribute Class [page 315]
Associates a security action with a custom security attribute.

SARowsCopiedEventArgs Class [page 317]
Represents the set of arguments passed to the SARowsCopiedEventHandler.

SARowUpdatedEventArgs Class [page 319]
Provides data for the RowUpdated event.

SARowUpdatingEventArgs Class [page 321]
Provides data for the RowUpdating event.

SAServerSideConnection Class [page 323]
Represents a server-side connection to a database.

SATcpOptionsBuilder Class [page 325]

Provides a simple way to create and manage the TCP options portion of connection strings used by the SAConnection object.

SATransaction Class [page 335]
Represents a SQL transaction.

SAInfoMessageEventHandler(object, SAInfoMessageEventArgs) Delegate [page 342]
Represents the method that handles the SAConnection.InfoMessage event of an SAConnection object.

SARowsCopiedEventHandler(object, SARowsCopiedEventArgs) Delegate [page 343]
Represents the method that handles the SABulkCopy.SARowsCopied event of an SABulkCopy.

SARowUpdatedEventHandler(object, SARowUpdatedEventArgs) Delegate [page 343]
Represents the method that handles the RowUpdated event of an SADataAdapter.

SARowUpdatingEventHandler(object, SARowUpdatingEventArgs) Delegate [page 344]
Represents the method that handles the RowUpdating event of an SADataAdapter.

SABulkCopyOptions Enumeration [page 344]
A bitwise flag that specifies one or more options to use with an instance of SABulkCopy.

SADbType Enumeration [page 346]
Enumerates the database server .NET database data types.

SAIsolationLevel Enumeration [page 350]
Specifies database server isolation levels.

SAMessageType Enumeration [page 352]
Identifies the type of message.

1.1 SABulkCopy Class

Efficiently bulk load a database table with data from another source.

Syntax

Visual Basic

```
Public NotInheritable Class SABulkCopy Implements System.IDisposable
```

C#

```
public sealed class SABulkCopy : System.IDisposable
```

Members

All members of SABulkCopy, including inherited members.

Constructors

Modifier and Type	Constructor	Description
public	SABulkCopy [page 16]	Initializes an SABulkCopy object.

Methods

Modifier and Type	Method	Description
public void	Close() [page 20]	Closes the SABulkCopy instance.
public void	Dispose() [page 20]	Disposes of the SABulkCopy instance.
public void	WriteToServer [page 20]	Copies all rows in the supplied array of System.Data.DataRow objects to a destination table specified by the DestinationTableName property of the SABulkCopy object.
public Task	WriteToServerAsync [page 24]	Copies all rows from the supplied System.Data.DataRow array to a destination table specified by the SABulkCopy.DestinationTableName property of the SABulkCopy object.

Properties

Modifier and Type	Property	Description
public int	BatchSize [page 33]	Gets or sets the number of rows in each batch.
public int	BulkCopyTimeout [page 33]	Gets or sets the number of seconds for the operation to complete before it times out.
public SABulkCopyColumnMappingCollection	ColumnMappings [page 34]	Returns a collection of SABulkCopyColumnMapping items.
public string	DestinationTableName [page 34]	Gets or sets the name of the destination table on the database server.
public int	NotifyAfter [page 35]	Gets or sets the number of rows to be processed before generating a notification event.

Events

Modifier and Type	Event	Description
public SARowsCopiedEventHandler	SARowsCopied [page 36]	This event occurs every time the number of rows specified by the NotifyAfter property have been processed.

Remarks

Implements: System.IDisposable

In this section:

1.1.1 SABulkCopy Constructor

Initializes an SABulkCopy object.

Overload list

Modifier and Type	Overload name	Description
public	SABulkCopy(SAConnection) [page 17]	Initializes an SABulkCopy object.
public	SABulkCopy(SAConnection, SABulkCopyOptions, SATransaction) [page 18]	Initializes an SABulkCopy object.
public	SABulkCopy(string) [page 18]	Initializes an SABulkCopy object.

Modifier and Type	Overload name	Description
public	SABulkCopy(string, SABulkCopyOptions) [page 19]	Initializes an SABulkCopy object.

In this section:

SABulkCopy(SAConnection) Constructor [page 17]
Initializes an SABulkCopy object.

SABulkCopy(SAConnection, SABulkCopyOptions, SATransaction) Constructor [page 18]
Initializes an SABulkCopy object.

SABulkCopy(string) Constructor [page 18]
Initializes an SABulkCopy object.

SABulkCopy(string, SABulkCopyOptions) Constructor [page 19]
Initializes an SABulkCopy object.

1.1.1.1 SABulkCopy(SAConnection) Constructor

Initializes an SABulkCopy object.

Syntax

Visual Basic

```
Public Sub SABulkCopy (ByVal connection As SAConnection)
```

C#

```
public SABulkCopy (SAConnection connection)
```

Parameters

connection An SAConnection object that is used to perform the bulk-copy operation. If the connection is not open, then an exception is thrown in WriteToServer.

1.1.1.2 SABulkCopy(SAConnection, SABulkCopyOptions, SATransaction) Constructor

Initializes an SABulkCopy object.

Syntax

Visual Basic

```
Public Sub SABulkCopy (
    ByVal connection As SAConnection,
    ByVal copyOptions As SABulkCopyOptions,
    ByVal externalTransaction As SATransaction
)
```

C#

```
public SABulkCopy (
    SAConnection connection,
    SABulkCopyOptions copyOptions,
    SATransaction externalTransaction
)
```

Parameters

connection An SAConnection object that is used to perform the bulk-copy operation. If the connection is not open, then an exception is thrown in WriteToServer.

copyOptions A combination of values from the SABulkCopyOptions enumeration that determines which data source rows are copied to the destination table.

externalTransaction An existing SATransaction instance under which the bulk copy will occur. If externalTransaction is not NULL, then the bulk-copy operation is done within it. It is an error to specify both an external transaction and the UseInternalTransaction option.

1.1.1.3 SABulkCopy(string) Constructor

Initializes an SABulkCopy object.

Syntax

Visual Basic

```
Public Sub SABulkCopy (ByVal connectionString As String)
```

C#

```
public SABulkCopy (string connectionString)
```

Parameters

connectionString The string defining the connection that is opened for use by the SABulkCopy instance. A connection string is a semicolon-separated list of keyword=value pairs.

Remarks

This syntax opens a connection during WriteToServer using connectionString. The connection is closed at the end of WriteToServer.

1.1.1.4 SABulkCopy(string, SABulkCopyOptions) Constructor

Initializes an SABulkCopy object.

Syntax

Visual Basic

```
Public Sub SABulkCopy (
    ByVal connectionString As String,
    ByVal copyOptions As SABulkCopyOptions
)
```

C#

```
public SABulkCopy (
    string connectionString,
    SABulkCopyOptions copyOptions
)
```

Parameters

connectionString The string defining the connection that is opened for use by the SABulkCopy instance. A connection string is a semicolon-separated list of keyword=value pairs.

copyOptions A combination of values from the SABulkCopyOptions enumeration that determines which data source rows are copied to the destination table.

Remarks

This syntax opens a connection during WriteToServer using connectionString. The connection is closed at the end of WriteToServer. The copyOptions parameter has the effects described above.

1.1.2 Close() Method

Closes the SABulkCopy instance.

Syntax

Visual Basic

```
Public Sub Close ()
```

C#

```
public void Close ()
```

1.1.3 Dispose() Method

Disposes of the SABulkCopy instance.

Syntax

Visual Basic

```
Public Sub Dispose ()
```

C#

```
public void Dispose ()
```

1.1.4 WriteToServer Method

Copies all rows in the supplied array of System.Data.DataRow objects to a destination table specified by the DestinationTableName property of the SABulkCopy object.

Overload list

Modifier and Type	Overload name	Description
public void	WriteToServer(DataRow[]) [page 21]	Copies all rows in the supplied array of System.Data.DataRow objects to a destination table specified by the DestinationTableName property of the SABulkCopy object.

Modifier and Type	Overload name	Description
public void	WriteToServer(DataTable) [page 22]	Copies all rows in the supplied System.Data.DataTable to a destination table specified by the DestinationTableName property of the SABulkCopy object.
public void	WriteToServer(DataTable, DataRowState) [page 23]	Copies all rows in the supplied System.Data.DataTable with the specified row state to a destination table specified by the DestinationTableName property of the SABulkCopy object.
public void	WriteToServer(IDataReader) [page 24]	Copies all rows in the supplied System.Data.IDataReader to a destination table specified by the DestinationTableName property of the SABulkCopy object.

In this section:

WriteToServer(DataRow[]) Method [page 21]
Copies all rows in the supplied array of System.Data.DataRow objects to a destination table specified by the DestinationTableName property of the SABulkCopy object.

WriteToServer(DataTable) Method [page 22]
Copies all rows in the supplied System.Data.DataTable to a destination table specified by the DestinationTableName property of the SABulkCopy object.

WriteToServer(DataTable, DataRowState) Method [page 23]
Copies all rows in the supplied System.Data.DataTable with the specified row state to a destination table specified by the DestinationTableName property of the SABulkCopy object.

WriteToServer(IDataReader) Method [page 24]
Copies all rows in the supplied System.Data.IDataReader to a destination table specified by the DestinationTableName property of the SABulkCopy object.

1.1.4.1 WriteToServer(DataRow[]) Method

Copies all rows in the supplied array of System.Data.DataRow objects to a destination table specified by the DestinationTableName property of the SABulkCopy object.

Syntax

Visual Basic

```
Public Sub WriteToServer (ByVal rows As DataRow())
```

C#

```
public void WriteToServer (DataRow[] rows)
```

Parameters

rows An array of System.Data.DataRow objects that are copied to the destination table.

Related Information

DestinationTableName Property [page 34]

1.1.4.2 WriteToServer(DataTable) Method

Copies all rows in the supplied System.Data.DataTable to a destination table specified by the DestinationTableName property of the SABulkCopy object.

Syntax

Visual Basic

```
Public Sub WriteToServer (ByVal table As DataTable)
```

C#

```
public void WriteToServer (DataTable table)
```

Parameters

table A System.Data.DataTable whose rows are copied to the destination table.

Related Information

DestinationTableName Property [page 34]

1.1.4.3 WriteToServer(DataTable, DataRowState) Method

Copies all rows in the supplied System.Data.DataTable with the specified row state to a destination table specified by the DestinationTableName property of the SABulkCopy object.

Syntax

Visual Basic

```
Public Sub WriteToServer (
    ByVal table As DataTable,
    ByVal rowState As DataRowState
)
```

C#

```
public void WriteToServer (
    DataTable table,
    DataRowState rowState
)
```

Parameters

table A System.Data.DataTable whose rows are copied to the destination table.
rowState A value from the System.Data.DataRowState enumeration. Only rows matching the row state are copied to the destination.

Remarks

Only those rows matching the row state are copied.

Related Information

DestinationTableName Property [page 34]

1.1.4.4 WriteToServer(IDataReader) Method

Copies all rows in the supplied System.Data.IDataReader to a destination table specified by the DestinationTableName property of the SABulkCopy object.

Syntax

Visual Basic

```
Public Sub WriteToServer (ByVal reader As IDataReader)
```

C#

```
public void WriteToServer (IDataReader reader)
```

Parameters

reader A System.Data.IDataReader whose rows are copied to the destination table.

Related Information

DestinationTableName Property [page 34]

1.1.5 WriteToServerAsync Method

Copies all rows from the supplied System.Data.DataRow array to a destination table specified by the SABulkCopy.DestinationTableName property of the SABulkCopy object.

Overload list

Modifier and Type	Overload name	Description
public Task	WriteToServerAsync(DataRow[]) [page 26]	Copies all rows from the supplied System.Data.DataRow array to a destination table specified by the SABulkCopy.DestinationTableName property of the SABulkCopy object.

Modifier and Type	Overload name	Description
public Task	WriteToServerAsync(DataRow[], CancellationToken) [page 27]	Copies all rows from the supplied System.Data.DataRow array to a destination table specified by the SABulkCopy.DestinationTableName property of the SABulkCopy object.
public Task	WriteToServerAsync(DataTable) [page 28]	Copies all rows in the supplied System.Data.DataTable to a destination table specified by the SABulkCopy.DestinationTableName property of the SABulkCopy object.
public Task	WriteToServerAsync(DataTable, CancellationToken) [page 28]	Copies all rows in the supplied System.Data.DataTable to a destination table specified by the SABulkCopy.DestinationTableName property of the SABulkCopy object. The cancellation token can be used to request that the operation be abandoned before the command timeout elapses.
public Task	WriteToServerAsync(DataTable, DataRowState) [page 29]	Copies only rows that match the supplied row state in the supplied System.Data.DataTable to a destination table specified by the SABulkCopy.DestinationTableName property of the SABulkCopy object.
public Task	WriteToServerAsync(DataTable, DataRowState, CancellationToken) [page 30]	Copies only rows that match the supplied row state in the supplied System.Data.DataTable to a destination table specified by the SABulkCopy.DestinationTableName property of the SABulkCopy object. The cancellation token can be used to request that the operation be abandoned before the command timeout elapses.
public Task	WriteToServerAsync(IDataReader) [page 31]	Copies all rows in the supplied System.Data.IDataReader to a destination table specified by the SABulkCopy.DestinationTableName property of the SABulkCopy object.
public Task	WriteToServerAsync(IDataReader, CancellationToken) [page 32]	Copies all rows in the supplied System.Data.IDataReader to a destination table specified by the SAClient.SqlBulkCopy.DestinationTableName property of the SAClient.SqlBulkCopy object. The cancellation token can be used to request that the operation be abandoned before the command timeout elapses.

In this section:

WriteToServerAsync(DataRow[]) Method [page 26]
Copies all rows from the supplied System.Data.DataRow array to a destination table specified by the SABulkCopy.DestinationTableName property of the SABulkCopy object.

WriteToServerAsync(DataRow[], CancellationToken) Method [page 27]
Copies all rows from the supplied System.Data.DataRow array to a destination table specified by the SABulkCopy.DestinationTableName property of the SABulkCopy object.

WriteToServerAsync(DataTable) Method [page 28]
Copies all rows in the supplied System.Data.DataTable to a destination table specified by the SABulkCopy.DestinationTableName property of the SABulkCopy object.

WriteToServerAsync(DataTable, CancellationToken) Method [page 28]
Copies only rows in the supplied System.Data.DataTable to a destination table specified by the SABulkCopy.DestinationTableName property of the SABulkCopy object.The cancellation token can be used to request that the operation be abandoned before the command timeout elapses.

WriteToServerAsync(DataTable, DataRowState) Method [page 29]
Copies only rows that match the supplied row state in the supplied System.Data.DataTable to a destination table specified by the SABulkCopy.DestinationTableName property of the SABulkCopy object.

WriteToServerAsync(DataTable, DataRowState, CancellationToken) Method [page 30]
Copies only rows that match the supplied row state in the supplied System.Data.DataTable to a destination table specified by the SABulkCopy.DestinationTableName property of the SABulkCopy object.The cancellation token can be used to request that the operation be abandoned before the command timeout elapses.

WriteToServerAsync(IDataReader) Method [page 31]
Copies all rows in the supplied System.Data.IDataReader to a destination table specified by the SABulkCopy.DestinationTableName property of the SABulkCopy object.

WriteToServerAsync(IDataReader, CancellationToken) Method [page 32]
Copies all rows in the supplied System.Data.IDataReader to a destination table specified by the SAClient.SqlBulkCopy.DestinationTableName property of the SAClient.SqlBulkCopy object.The cancellation token can be used to request that the operation be abandoned before the command timeout elapses.

1.1.5.1 WriteToServerAsync(DataRow[]) Method

Copies all rows from the supplied System.Data.DataRow array to a destination table specified by the SABulkCopy.DestinationTableName property of the SABulkCopy object.

Syntax

Visual Basic

```
Public Function WriteToServerAsync (ByVal rows As DataRow()) As Task
```

C#

```
public Task WriteToServerAsync (DataRow[] rows)
```

Parameters

rows An array of System.Data.DataRow objects that are copied to the destination table.

1.1.5.2 WriteToServerAsync(DataRow[], CancellationToken) Method

Copies all rows from the supplied System.Data.DataRow array to a destination table specified by the SABulkCopy.DestinationTableName property of the SABulkCopy object.

Syntax

Visual Basic

```
Public Function WriteToServerAsync (
    ByVal rows As DataRow(),
    ByVal cancellationToken As CancellationToken
) As Task
```

C#

```
public Task WriteToServerAsync (
    DataRow[] rows,
    CancellationToken cancellationToken
)
```

Parameters

rows An array of System.Data.DataRow objects that are copied to the destination table.

cancellationToken The cancellation instruction. A P:System.Threading.CancellationToken.None value in this parameter makes this method equivalent to SABulkCopy.WriteToServerAsync(System.Data.DataTable).

Returns

A task representing the asynchronous operation.

Exceptions

SAException class Returned in the task object, any error returned by the database server that occurred while copying data.

Remarks

The cancellation token can be used to request that the operation be abandoned before the command timeout elapses. Exceptions are reported via the returned Task object.

1.1.5.3 WriteToServerAsync(DataTable) Method

Copies all rows in the supplied System.Data.DataTable to a destination table specified by the SABulkCopy.DestinationTableName property of the SABulkCopy object.

Syntax

Visual Basic

```
Public Function WriteToServerAsync (ByVal table As DataTable) As Task
```

C#

```
public Task WriteToServerAsync (DataTable table)
```

Parameters

table A System.Data.DataTable whose rows are copied to the destination table.

1.1.5.4 WriteToServerAsync(DataTable, CancellationToken) Method

Copies only rows in the supplied System.Data.DataTable to a destination table specified by the SABulkCopy.DestinationTableName property of the SABulkCopy object.The cancellation token can be used to request that the operation be abandoned before the command timeout elapses.

Syntax

Visual Basic

```
Public Function WriteToServerAsync (
    ByVal table As DataTable,
    ByVal cancellationToken As CancellationToken
) As Task
```

C#

```
public Task WriteToServerAsync (
    DataTable table,
    CancellationToken cancellationToken
```

```
)
```

Parameters

table A System.Data.DataTable whose rows are copied to the destination table.

cancellationToken The cancellation instruction. A P:System.Threading.CancellationToken.None value in this parameter makes this method equivalent to SABulkCopy.WriteToServerAsync(System.Data.DataTable).

Returns

A task representing the asynchronous operation.

Exceptions

SAException class Returned in the task object, any error returned by SQL Server that occurred while copying data.

Remarks

Exceptions are reported via the returned Task object.

1.1.5.5 WriteToServerAsync(DataTable, DataRowState) Method

Copies only rows that match the supplied row state in the supplied System.Data.DataTable to a destination table specified by the SABulkCopy.DestinationTableName property of the SABulkCopy object.

Syntax

Visual Basic

```
Public Function WriteToServerAsync (
    ByVal table As DataTable,
    ByVal rowState As DataRowState
) As Task
```

C#

```
public Task WriteToServerAsync (
    DataTable table,
    DataRowState rowState
)
```

Parameters

table A System.Data.DataTable whose rows are copied to the destination table.

rowState A value from the System.Data.DataRowState enumeration. Only rows matching the row state are copied to the destination.

1.1.5.6 WriteToServerAsync(DataTable, DataRowState, CancellationToken) Method

Copies only rows that match the supplied row state in the supplied System.Data.DataTable to a destination table specified by the SABulkCopy.DestinationTableName property of the SABulkCopy object.The cancellation token can be used to request that the operation be abandoned before the command timeout elapses.

Syntax

Visual Basic

```
Public Function WriteToServerAsync (
    ByVal table As DataTable,
    ByVal rowState As DataRowState,
    ByVal cancellationToken As CancellationToken
) As Task
```

C#

```
public Task WriteToServerAsync (
    DataTable table,
    DataRowState rowState,
    CancellationToken cancellationToken
)
```

Parameters

table A System.Data.DataTable whose rows are copied to the destination table.

rowState A value from the System.Data.DataRowState enumeration. Only rows matching the row state are copied to the destination.

cancellationToken The cancellation instruction. A P:System.Threading.CancellationToken.None value in this parameter makes this method equivalent to SABulkCopy.WriteToServerAsync(System.Data.DataTable, System.Data.DataRowState).

Returns

A task representing the asynchronous operation.

Exceptions

SAException class Returned in the task object, any error returned by SQL Server that occurred while copying data.

Remarks

Exceptions are reported via the returned Task object.

1.1.5.7 WriteToServerAsync(IDataReader) Method

Copies all rows in the supplied System.Data.IDataReader to a destination table specified by the SABulkCopy.DestinationTableName property of the SABulkCopy object.

Syntax

Visual Basic

```
Public Function WriteToServerAsync (ByVal reader As IDataReader) As Task
```

C#

```
public Task WriteToServerAsync (IDataReader reader)
```

Parameters

reader A System.Data.IDataReader whose rows are copied to the destination table.

1.1.5.8 WriteToServerAsync(IDataReader, CancellationToken) Method

Copies all rows in the supplied System.Data.IDataReader to a destination table specified by the SAClient.SqlBulkCopy.DestinationTableName property of the SAClient.SqlBulkCopy object.The cancellation token can be used to request that the operation be abandoned before the command timeout elapses.

Syntax

Visual Basic

```
Public Function WriteToServerAsync (
    ByVal reader As IDataReader,
    ByVal cancellationToken As CancellationToken
) As Task
```

C#

```
public Task WriteToServerAsync (
    IDataReader reader,
    CancellationToken cancellationToken
)
```

Parameters

reader A System.Data.IDataReader whose rows are copied to the destination table.

cancellationToken The cancellation instruction. A P:System.Threading.CancellationToken.None value in this parameter makes this method equivalent to SABulkCopy.WriteToServerAsync(System.Data.IDataReader).

Returns

A task representing the asynchronous operation.

Exceptions

SAException class Returned in the task object, any error returned by SQL Server that occurred while copying data.

Remarks

Exceptions are reported via the returned Task object.

1.1.6 BatchSize Property

Gets or sets the number of rows in each batch.

Syntax

Visual Basic

```
Public Property BatchSize As Integer
```

C#

```
public int BatchSize {get;set;}
```

Remarks

At the end of each batch, the rows in the batch are sent to the database server.

The number of rows in each batch. The default is 0.

Setting this property to zero causes all the rows to be sent in one batch.

Setting this property to a value less than zero is an error.

If this value is changed while a batch is in progress, then the current batch completes and any further batches use the new value.

1.1.7 BulkCopyTimeout Property

Gets or sets the number of seconds for the operation to complete before it times out.

Syntax

Visual Basic

```
Public Property BulkCopyTimeout As Integer
```

C#

```
public int BulkCopyTimeout {get;set;}
```

Remarks

The default value is 30 seconds.

A value of zero indicates no limit. This value should be avoided because it may cause an indefinite wait.

If the operation times out, then all rows in the current transaction are rolled back and an SAException is raised.

Setting this property to a value less than zero is an error.

1.1.8 ColumnMappings Property

Returns a collection of SABulkCopyColumnMapping items.

Syntax

Visual Basic

```
Public ReadOnly Property ColumnMappings As
SABulkCopyColumnMappingCollection
```

C#

```
public SABulkCopyColumnMappingCollection ColumnMappings {get;}
```

Remarks

Column mappings define the relationships between columns in the data source and columns in the destination.

By default, it is an empty collection.

The property cannot be modified while WriteToServer is executing.

If ColumnMappings is empty when WriteToServer is executed, then the first column in the source is mapped to the first column in the destination, the second to the second, and so on. This behavior takes place as long as the column types are convertible, there are at least as many destination columns as source columns, and any extra destination columns are nullable.

1.1.9 DestinationTableName Property

Gets or sets the name of the destination table on the database server.

Syntax

Visual Basic

```
Public Property DestinationTableName As String
```

C#

```
public string DestinationTableName {get;set;}
```

Remarks

The default value is a null reference. In Visual Basic it is Nothing.

If the value is changed while WriteToServer is executing, then the change has no effect.

If the value has not been set before a call to WriteToServer, then an InvalidOperationException is raised.

It is an error to set the value to NULL or the empty string.

1.1.10 NotifyAfter Property

Gets or sets the number of rows to be processed before generating a notification event.

Syntax

Visual Basic

```
Public Property NotifyAfter As Integer
```

C#

```
public int NotifyAfter {get;set;}
```

Remarks

Zero is returned if the property has not been set.

Changes made to NotifyAfter, while executing WriteToServer, do not take effect until after the next notification.

Setting this property to a value less than zero is an error.

The values of NotifyAfter and BulkCopyTimeout are mutually exclusive, so the event can fire even if no rows have been sent to the database or committed.

Related Information

BulkCopyTimeout Property [page 33]

1.1.11 SARowsCopied Event

This event occurs every time the number of rows specified by the NotifyAfter property have been processed.

Syntax

Visual Basic

```
Public Event SARowsCopied  As SARowsCopiedEventHandler
```

C#

```
public SARowsCopiedEventHandler SARowsCopied;
```

Remarks

The receipt of an SARowsCopied event does not imply that any rows have been sent to the database server or committed. You cannot call the Close method from this event.

Related Information

NotifyAfter Property [page 35]

1.2 SABulkCopyColumnMapping Class

Defines the mapping between a column in an SABulkCopy instance's data source and a column in the instance's destination table.

Syntax

Visual Basic

```
Public NotInheritable Class SABulkCopyColumnMapping
```

C#

```
public sealed class SABulkCopyColumnMapping
```

Members

All members of SABulkCopyColumnMapping, including inherited members.

Constructors

Modifier and Type	Constructor	Description
public	SABulkCopyColumnMapping [page 38]	Creates a new column mapping, using column ordinals or names to refer to source and destination columns.

Properties

Modifier and Type	Property	Description
public string	DestinationColumn [page 41]	Gets or sets the name of the column in the destination database table being mapped to.
public int	DestinationOrdinal [page 42]	Gets or sets the ordinal value of the column in the destination table being mapped to.
public string	SourceColumn [page 43]	Gets or sets the name of the column being mapped in the data source.
public int	SourceOrdinal [page 44]	Gets or sets ordinal position of the source column within the data source.

In this section:

SABulkCopyColumnMapping Constructor [page 38]
Creates a new column mapping, using column ordinals or names to refer to source and destination columns.

DestinationColumn Property [page 41]
Gets or sets the name of the column in the destination database table being mapped to.

DestinationOrdinal Property [page 42]
Gets or sets the ordinal value of the column in the destination table being mapped to.

SourceColumn Property [page 43]
Gets or sets the name of the column being mapped in the data source.

SourceOrdinal Property [page 44]
Gets or sets ordinal position of the source column within the data source.

1.2.1 SABulkCopyColumnMapping Constructor

Creates a new column mapping, using column ordinals or names to refer to source and destination columns.

Overload list

Modifier and Type	Overload name	Description
public	SABulkCopyColumnMapping() [page 39]	Creates a new column mapping, using column ordinals or names to refer to source and destination columns.
public	SABulkCopyColumnMapping(int, int) [page 39]	Creates a new column mapping, using column ordinals to refer to source and destination columns.
public	SABulkCopyColumnMapping(int, string) [page 40]	Creates a new column mapping, using a column ordinal to refer to the source column and a column name to refer to the destination column.
public	SABulkCopyColumnMapping(string, int) [page 40]	Creates a new column mapping, using a column name to refer to the source column and a column ordinal to refer to the destination column.
public	SABulkCopyColumnMapping(string, string) [page 41]	Creates a new column mapping, using column names to refer to source and destination columns.

In this section:

SABulkCopyColumnMapping() Constructor [page 39]
Creates a new column mapping, using column ordinals or names to refer to source and destination columns.

SABulkCopyColumnMapping(int, int) Constructor [page 39]
Creates a new column mapping, using column ordinals to refer to source and destination columns.

SABulkCopyColumnMapping(int, string) Constructor [page 40]
Creates a new column mapping, using a column ordinal to refer to the source column and a column name to refer to the destination column.

SABulkCopyColumnMapping(string, int) Constructor [page 40]
Creates a new column mapping, using a column name to refer to the source column and a column ordinal to refer to the destination column.

SABulkCopyColumnMapping(string, string) Constructor [page 41]
Creates a new column mapping, using column names to refer to source and destination columns.

1.2.1.1 SABulkCopyColumnMapping() Constructor

Creates a new column mapping, using column ordinals or names to refer to source and destination columns.

Syntax

Visual Basic

```
Public Sub SABulkCopyColumnMapping ()
```

C#

```
public SABulkCopyColumnMapping ()
```

1.2.1.2 SABulkCopyColumnMapping(int, int) Constructor

Creates a new column mapping, using column ordinals to refer to source and destination columns.

Syntax

Visual Basic

```
Public Sub SABulkCopyColumnMapping (
    ByVal sourceColumnOrdinal As Integer,
    ByVal destinationColumnOrdinal As Integer
)
```

C#

```
public SABulkCopyColumnMapping (
    int sourceColumnOrdinal,
    int destinationColumnOrdinal
)
```

Parameters

sourceColumnOrdinal The ordinal position of the source column within the data source. The first column in a data source has ordinal position zero.

destinationColumnOrdinal The ordinal position of the destination column within the destination table. The first column in a table has ordinal position zero.

1.2.1.3 SABulkCopyColumnMapping(int, string) Constructor

Creates a new column mapping, using a column ordinal to refer to the source column and a column name to refer to the destination column.

Syntax

Visual Basic

```
Public Sub SABulkCopyColumnMapping (
    ByVal sourceColumnOrdinal As Integer,
    ByVal destinationColumn As String
)
```

C#

```
public SABulkCopyColumnMapping (
    int sourceColumnOrdinal,
    string destinationColumn
)
```

Parameters

sourceColumnOrdinal The ordinal position of the source column within the data source. The first column in a data source has ordinal position zero.

destinationColumn The name of the destination column within the destination table.

1.2.1.4 SABulkCopyColumnMapping(string, int) Constructor

Creates a new column mapping, using a column name to refer to the source column and a column ordinal to refer to the destination column.

Syntax

Visual Basic

```
Public Sub SABulkCopyColumnMapping (
    ByVal sourceColumn As String,
    ByVal destinationColumnOrdinal As Integer
)
```

C#

```
public SABulkCopyColumnMapping (
    string sourceColumn,
    int destinationColumnOrdinal
)
```

Parameters

sourceColumn The name of the source column within the data source.
destinationColumnOrdinal The ordinal position of the destination column within the destination table. The first column in a table has ordinal position zero.

1.2.1.5 SABulkCopyColumnMapping(string, string) Constructor

Creates a new column mapping, using column names to refer to source and destination columns.

Syntax

Visual Basic

```
Public Sub SABulkCopyColumnMapping (
    ByVal sourceColumn As String,
    ByVal destinationColumn As String
)
```

C#

```
public SABulkCopyColumnMapping (
    string sourceColumn,
    string destinationColumn
)
```

Parameters

sourceColumn The name of the source column within the data source.
destinationColumn The name of the destination column within the destination table.

1.2.2 DestinationColumn Property

Gets or sets the name of the column in the destination database table being mapped to.

Syntax

Visual Basic

```
Public Property DestinationColumn As String
```

C#

```
public string DestinationColumn {get;set;}
```

Remarks

A string specifying the name of the column in the destination table or a null reference (Nothing in Visual Basic) if the DestinationOrdinal property has priority.

The DestinationColumn property and DestinationOrdinal property are mutually exclusive. The most recently set value takes priority.

Setting the DestinationColumn property causes the DestinationOrdinal property to be set to -1. Setting the DestinationOrdinal property causes the DestinationColumn property to be set to a null reference (Nothing in Visual Basic).

It is an error to set DestinationColumn to null or the empty string.

Related Information

DestinationOrdinal Property [page 42]

1.2.3 DestinationOrdinal Property

Gets or sets the ordinal value of the column in the destination table being mapped to.

Syntax

Visual Basic

```
Public Property DestinationOrdinal As Integer
```

C#

```
public int DestinationOrdinal {get;set;}
```

Remarks

An integer specifying the ordinal of the column being mapped to in the destination table or -1 if the property is not set.

The DestinationColumn property and DestinationOrdinal property are mutually exclusive. The most recently set value takes priority.

Setting the DestinationColumn property causes the DestinationOrdinal property to be set to -1. Setting the DestinationOrdinal property causes the DestinationColumn property to be set to a null reference (Nothing in Visual Basic).

Related Information

DestinationColumn Property [page 41]

1.2.4 SourceColumn Property

Gets or sets the name of the column being mapped in the data source.

Syntax

Visual Basic

```
Public Property SourceColumn As String
```

C#

```
public string SourceColumn {get;set;}
```

Remarks

A string specifying the name of the column in the data source or a null reference (Nothing in Visual Basic) if the SourceOrdinal property has priority.

The SourceColumn property and SourceOrdinal property are mutually exclusive. The most recently set value takes priority.

Setting the SourceColumn property causes the SourceOrdinal property to be set to -1. Setting the SourceOrdinal property causes the SourceColumn property to be set to a null reference (Nothing in Visual Basic).

It is an error to set SourceColumn to null or the empty string.

Related Information

SourceOrdinal Property [page 44]

1.2.5 SourceOrdinal Property

Gets or sets ordinal position of the source column within the data source.

Syntax

Visual Basic

```
Public Property SourceOrdinal As Integer
```

C#

```
public int SourceOrdinal {get;set;}
```

Remarks

An integer specifying the ordinal of the column in the data source or -1 if the property is not set.

The SourceColumn property and SourceOrdinal property are mutually exclusive. The most recently set value takes priority.

Setting the SourceColumn property causes the SourceOrdinal property to be set to -1. Setting the SourceOrdinal property causes the SourceColumn property to be set to a null reference (Nothing in Visual Basic).

Related Information

SourceColumn Property [page 43]

1.3 SABulkCopyColumnMappingCollection Class

A collection of SABulkCopyColumnMapping objects that inherits from System.Collections.CollectionBase.

Syntax

Visual Basic

```
Public NotInheritable Class SABulkCopyColumnMappingCollection Inherits
System.Collections.CollectionBase
```

C#

```
public sealed class SABulkCopyColumnMappingCollection :
System.Collections.CollectionBase
```

Members

All members of SABulkCopyColumnMappingCollection, including inherited members.

Methods

Modifier and Type	Method	Description
public SABulkCopyColumnMapping	Add [page 46]	Adds the specified SABulkCopyColumnMapping object to the collection.
public bool	Contains(SABulkCopyColumnMapping) [page 50]	Gets a value indicating whether a specified SABulkCopyColumnMapping object exists in the collection.
public void	CopyTo(SABulkCopyColumnMapping[], int) [page 51]	Copies the elements of the SABulkCopyColumnMappingCollection to an array of SABulkCopyColumnMapping items, starting at a particular index.
public int	IndexOf(SABulkCopyColumnMapping) [page 51]	Gets or sets the index of the specified SABulkCopyColumnMapping object within the collection.
public void	Remove(SABulkCopyColumnMapping) [page 52]	Removes the specified SABulkCopyColumnMapping element from the SABulkCopyColumnMappingCollection.
public new void	RemoveAt(int) [page 52]	Removes the mapping at the specified index from the collection.

Properties

Modifier and Type	Property	Description
public SABulkCopyColumnMapping	this[int index] [page 53]	Gets the SABulkCopyColumnMapping object at the specified index.

Remarks

Implements: ICollection, IEnumerable, IList

In this section:

Add Method [page 46]
Adds the specified SABulkCopyColumnMapping object to the collection.

Contains(SABulkCopyColumnMapping) Method [page 50]
Gets a value indicating whether a specified SABulkCopyColumnMapping object exists in the collection.

CopyTo(SABulkCopyColumnMapping[], int) Method [page 51]
Copies the elements of the SABulkCopyColumnMappingCollection to an array of SABulkCopyColumnMapping items, starting at a particular index.

IndexOf(SABulkCopyColumnMapping) Method [page 51]
Gets or sets the index of the specified SABulkCopyColumnMapping object within the collection.

Remove(SABulkCopyColumnMapping) Method [page 52]
Removes the specified SABulkCopyColumnMapping element from the SABulkCopyColumnMappingCollection.

RemoveAt(int) Method [page 52]
Removes the mapping at the specified index from the collection.

this[int index] Property [page 53]
Gets the SABulkCopyColumnMapping object at the specified index.

1.3.1 Add Method

Adds the specified SABulkCopyColumnMapping object to the collection.

Overload list

Modifier and Type	Overload name	Description
public SABulkCopyColumnMapping	Add(SABulkCopyColumnMapping) [page 47]	Adds the specified SABulkCopyColumnMapping object to the collection.
public SABulkCopyColumnMapping	Add(int, int) [page 48]	Creates a new SABulkCopyColumnMapping object using ordinals to specify both source and destination columns, and adds the mapping to the collection.
public SABulkCopyColumnMapping	Add(int, string) [page 48]	Creates a new SABulkCopyColumnMapping object using a column ordinal to refer to the source column and a column name to refer to the destination column, and adds mapping to the collection.
public SABulkCopyColumnMapping	Add(string, int) [page 49]	Creates a new SABulkCopyColumnMapping object using a column name to refer to the source column and a column ordinal to refer to the destination the column, and adds the mapping to the collection.
public SABulkCopyColumnMapping	Add(string, string) [page 50]	Creates a new SABulkCopyColumnMapping object using column names to specify both source and destination columns, and adds the mapping to the collection.

In this section:

Add(SABulkCopyColumnMapping) Method [page 47]
Adds the specified SABulkCopyColumnMapping object to the collection.

Add(int, int) Method [page 48]
Creates a new SABulkCopyColumnMapping object using ordinals to specify both source and destination columns, and adds the mapping to the collection.

Add(int, string) Method [page 48]
Creates a new SABulkCopyColumnMapping object using a column ordinal to refer to the source column and a column name to refer to the destination column, and adds mapping to the collection.

Add(string, int) Method [page 49]
Creates a new SABulkCopyColumnMapping object using a column name to refer to the source column and a column ordinal to refer to the destination the column, and adds the mapping to the collection.

Add(string, string) Method [page 50]
Creates a new SABulkCopyColumnMapping object using column names to specify both source and destination columns, and adds the mapping to the collection.

1.3.1.1 Add(SABulkCopyColumnMapping) Method

Adds the specified SABulkCopyColumnMapping object to the collection.

Syntax

Visual Basic

```
Public Function Add (ByVal bulkCopyColumnMapping As
SABulkCopyColumnMapping) As SABulkCopyColumnMapping
```

C#

```
public SABulkCopyColumnMapping Add (SABulkCopyColumnMapping
bulkCopyColumnMapping)
```

Parameters

bulkCopyColumnMapping The SABulkCopyColumnMapping object that describes the mapping to be added to the collection.

Related Information

SABulkCopyColumnMapping Class [page 36]

1.3.1.2 Add(int, int) Method

Creates a new SABulkCopyColumnMapping object using ordinals to specify both source and destination columns, and adds the mapping to the collection.

Syntax

Visual Basic

```
Public Function Add (
    ByVal sourceColumnOrdinal As Integer,
    ByVal destinationColumnOrdinal As Integer
) As SABulkCopyColumnMapping
```

C#

```
public SABulkCopyColumnMapping Add (
    int sourceColumnOrdinal,
    int destinationColumnOrdinal
)
```

Parameters

sourceColumnOrdinal The ordinal position of the source column within the data source.

destinationColumnOrdinal The ordinal position of the destination column within the destination table.

1.3.1.3 Add(int, string) Method

Creates a new SABulkCopyColumnMapping object using a column ordinal to refer to the source column and a column name to refer to the destination column, and adds mapping to the collection.

Syntax

Visual Basic

```
Public Function Add (
    ByVal sourceColumnOrdinal As Integer,
    ByVal destinationColumn As String
) As SABulkCopyColumnMapping
```

C#

```
public SABulkCopyColumnMapping Add (
    int sourceColumnOrdinal,
    string destinationColumn
)
```

Parameters

sourceColumnOrdinal The ordinal position of the source column within the data source.
destinationColumn The name of the destination column within the destination table.

1.3.1.4 Add(string, int) Method

Creates a new SABulkCopyColumnMapping object using a column name to refer to the source column and a column ordinal to refer to the destination the column, and adds the mapping to the collection.

Syntax

Visual Basic

```
Public Function Add (
    ByVal sourceColumn As String,
    ByVal destinationColumnOrdinal As Integer
) As SABulkCopyColumnMapping
```

C#

```
public SABulkCopyColumnMapping Add (
    string sourceColumn,
    int destinationColumnOrdinal
)
```

Parameters

sourceColumn The name of the source column within the data source.
destinationColumnOrdinal The ordinal position of the destination column within the destination table.

Remarks

Creates a new column mapping, using column ordinals or names to refer to source and destination columns.

1.3.1.5 Add(string, string) Method

Creates a new SABulkCopyColumnMapping object using column names to specify both source and destination columns, and adds the mapping to the collection.

Syntax

Visual Basic

```
Public Function Add (
    ByVal sourceColumn As String,
    ByVal destinationColumn As String
) As SABulkCopyColumnMapping
```

C#

```
public SABulkCopyColumnMapping Add (
    string sourceColumn,
    string destinationColumn
)
```

Parameters

sourceColumn The name of the source column within the data source.
destinationColumn The name of the destination column within the destination table.

1.3.2 Contains(SABulkCopyColumnMapping) Method

Gets a value indicating whether a specified SABulkCopyColumnMapping object exists in the collection.

Syntax

Visual Basic

```
Public Function Contains (ByVal value As SABulkCopyColumnMapping) As
Boolean
```

C#

```
public bool Contains (SABulkCopyColumnMapping value)
```

Parameters

value A valid SABulkCopyColumnMapping object.

Returns

True if the specified mapping exists in the collection; false otherwise.

1.3.3 CopyTo(SABulkCopyColumnMapping[], int) Method

Copies the elements of the SABulkCopyColumnMappingCollection to an array of SABulkCopyColumnMapping items, starting at a particular index.

Syntax

Visual Basic

```
Public Sub CopyTo (
    ByVal array As SABulkCopyColumnMapping(),
    ByVal index As Integer
)
```

C#

```
public void CopyTo (
    SABulkCopyColumnMapping[] array,
    int index
)
```

Parameters

array The one-dimensional SABulkCopyColumnMapping array that is the destination of the elements copied from SABulkCopyColumnMappingCollection. The array must have zero-based indexing.
index The zero-based index in the array at which copying begins.

1.3.4 IndexOf(SABulkCopyColumnMapping) Method

Gets or sets the index of the specified SABulkCopyColumnMapping object within the collection.

Syntax

Visual Basic

```
Public Function IndexOf (ByVal value As SABulkCopyColumnMapping) As Integer
```

C#

```
public int IndexOf (SABulkCopyColumnMapping value)
```

Parameters

value The SABulkCopyColumnMapping object to search for.

Returns

The zero-based index of the column mapping is returned, or -1 is returned if the column mapping is not found in the collection.

1.3.5 Remove(SABulkCopyColumnMapping) Method

Removes the specified SABulkCopyColumnMapping element from the SABulkCopyColumnMappingCollection.

Syntax

Visual Basic

```
Public Sub Remove (ByVal value As SABulkCopyColumnMapping)
```

C#

```
public void Remove (SABulkCopyColumnMapping value)
```

Parameters

value The SABulkCopyColumnMapping object to be removed from the collection.

1.3.6 RemoveAt(int) Method

Removes the mapping at the specified index from the collection.

Syntax

Visual Basic

```
Public Shadows Sub RemoveAt (ByVal index As Integer)
```

C#

```
public new void RemoveAt (int index)
```

Parameters

index The zero-based index of the SABulkCopyColumnMapping object to be removed from the collection.

1.3.7 this[int index] Property

Gets the SABulkCopyColumnMapping object at the specified index.

Syntax

Visual Basic

```
Public ReadOnly Property Item (ByVal index As Integer) As
SABulkCopyColumnMapping
```

C#

```
public SABulkCopyColumnMapping this[int index] {get;}
```

Returns

An SABulkCopyColumnMapping object is returned.

1.4 SACommand Class

A SQL statement or stored procedure that is executed against a database.

Syntax

Visual Basic

```
Public NotInheritable Class SACommand Inherits
System.Data.Common.DbCommand Implements System.ICloneable
```

C#

```
public sealed class SACommand : System.Data.Common.DbCommand,
System.ICloneable
```

Members

All members of SACommand, including inherited members.

Constructors

Modifier and Type	Constructor	Description
public	SACommand [page 57]	Initializes an SACommand object.

Methods

Modifier and Type	Method	Description
public IAsyncResult	BeginExecuteNonQuery [page 60]	Initiates the asynchronous execution of a SQL statement or stored procedure that is described by this SACommand.
public IAsyncResult	BeginExecuteReader [page 63]	Initiates the asynchronous execution of a SQL statement or stored procedure that is described by this SACommand, and retrieves one or more result sets from the database server.
public override void	Cancel() [page 68]	Cancels the execution of an SACommand object.
protected override DbParameter	CreateDbParameter() [page 69]	Creates a new instance of a System.Data.Common.DbParameter object.
public new SAParameter	CreateParameter() [page 69]	Provides an SAParameter object for supplying parameters to SACommand objects.
protected override void	Dispose(bool) [page 70]	Frees the resources associated with the object.
public unsafe int	EndExecuteNonQuery(IAsyncResult) [page 70]	Finishes asynchronous execution of a SQL statement or stored procedure.
public unsafe SADataReader	EndExecuteReader(IAsyncResult) [page 73]	Finishes asynchronous execution of a SQL statement or stored procedure, returning the requested SADataReader.
protected override DbDataReader	ExecuteDbDataReader(CommandBehavior) [page 75]	Executes the command text against the connection.
public override unsafe int	ExecuteNonQuery() [page 75]	Executes a statement that does not return a result set, such as an INSERT, UPDATE, DELETE, or data definition statement.
public new SADataReader	ExecuteReader [page 76]	Executes a SQL statement that returns a result set.
public new Task< SADataReader >	ExecuteReaderAsync [page 79]	An asynchronous version of SACommand.ExecuteReader, which Executes a SQL statement that returns a result set.
public override object	ExecuteScalar() [page 83]	Executes a statement that returns a single value.

Modifier and Type	Method	Description
public override void	Prepare() [page 84]	Prepares or compiles the SACommand on the data source.
public void	ResetCommandTimeout() [page 85]	Resets the CommandTimeout property to its default value of 30 seconds.

Properties

Modifier and Type	Property	Description
public override string	CommandText [page 85]	Gets or sets the text of a SQL statement or stored procedure.
public override int	CommandTimeout [page 86]	This feature is not supported by the .NET Data Provider.
public override CommandType	CommandType [page 86]	Gets or sets the type of command represented by an SACommand.
public new SAConnection	Connection [page 87]	Gets or sets the connection object to which the SACommand object applies.
protected override DbConnection	DbConnection [page 87]	Gets or sets the System.Data.Common.DbConnection used by this SACommand object.
protected override DbParameterCollection	DbParameterCollection [page 88]	Gets the collection of System.Data.Common.DbParameter objects.
protected override DbTransaction	DbTransaction [page 88]	Gets or sets the System.Data.Common.DbTransaction within which this SACommand object executes.
public override bool	DesignTimeVisible [page 89]	Gets or sets a value that indicates if the SACommand should be visible in a Windows Form Designer control.
public new SAParameterCollection	Parameters [page 89]	Specifies a collection of parameters for the current statement.
public new SATransaction	Transaction [page 90]	Specifies the SATransaction object in which the SACommand executes.
public override UpdateRowSource	UpdatedRowSource [page 91]	Gets or sets how command results are applied to the DataRow when used by the Update method of the SADataAdapter.

Remarks

Implements: IDbCommand, ICloneable

For more information, see "Data access and manipulation".

In this section:

Gets or sets the type of command represented by an SACommand.

Connection Property [page 87]
Gets or sets the connection object to which the SACommand object applies.

DbConnection Property [page 87]
Gets or sets the System.Data.Common.DbConnection used by this SACommand object.

DbParameterCollection Property [page 88]
Gets the collection of System.Data.Common.DbParameter objects.

DbTransaction Property [page 88]
Gets or sets the System.Data.Common.DbTransaction within which this SACommand object executes.

DesignTimeVisible Property [page 89]
Gets or sets a value that indicates if the SACommand should be visible in a Windows Form Designer control.

Parameters Property [page 89]
Specifies a collection of parameters for the current statement.

Transaction Property [page 90]
Specifies the SATransaction object in which the SACommand executes.

UpdatedRowSource Property [page 91]
Gets or sets how command results are applied to the DataRow when used by the Update method of the SADataAdapter.

1.4.1 SACommand Constructor

Initializes an SACommand object.

Overload list

Modifier and Type	Overload name	Description
public	SACommand() [page 58]	Initializes an SACommand object.
public	SACommand(string) [page 58]	Initializes an SACommand object.
public	SACommand(string, SAConnection) [page 59]	A SQL statement or stored procedure that is executed against a database.
public	SACommand(string, SAConnection, SATransaction) [page 59]	A SQL statement or stored procedure that is executed against a database.

In this section:

SACommand() Constructor [page 58]
Initializes an SACommand object.

SACommand(string) Constructor [page 58]
Initializes an SACommand object.

SACommand(string, SAConnection) Constructor [page 59]
A SQL statement or stored procedure that is executed against a database.

SACommand(string, SAConnection, SATransaction) Constructor [page 59]
A SQL statement or stored procedure that is executed against a database.

1.4.1.1 SACommand() Constructor

Initializes an SACommand object.

Syntax

Visual Basic

```
Public Sub SACommand ()
```

C#

```
public SACommand ()
```

1.4.1.2 SACommand(string) Constructor

Initializes an SACommand object.

Syntax

Visual Basic

```
Public Sub SACommand (ByVal cmdText As String)
```

C#

```
public SACommand (string cmdText)
```

Parameters

cmdText The text of the SQL statement or stored procedure. For parameterized statements, use a question mark (?) placeholder to pass parameters.

1.4.1.3 SACommand(string, SAConnection) Constructor

A SQL statement or stored procedure that is executed against a database.

Syntax

Visual Basic

```
Public Sub SACommand (
    ByVal cmdText As String,
    ByVal connection As SAConnection
)
```

C#

```
public SACommand (
    string cmdText,
    SAConnection connection
)
```

Parameters

cmdText The text of the SQL statement or stored procedure. For parameterized statements, use a question mark (?) placeholder to pass parameters.

connection The current connection.

1.4.1.4 SACommand(string, SAConnection, SATransaction) Constructor

A SQL statement or stored procedure that is executed against a database.

Syntax

Visual Basic

```
Public Sub SACommand (
    ByVal cmdText As String,
    ByVal connection As SAConnection,
    ByVal transaction As SATransaction
)
```

C#

```
public SACommand (
    string cmdText,
    SAConnection connection,
    SATransaction transaction
)
```

Parameters

cmdText The text of the SQL statement or stored procedure. For parameterized statements, use a question mark (?) placeholder to pass parameters.
connection The current connection.
transaction The SATransaction object in which the SAConnection executes.

Related Information

SATransaction Class [page 335]

1.4.2 BeginExecuteNonQuery Method

Initiates the asynchronous execution of a SQL statement or stored procedure that is described by this SACommand.

Overload list

Modifier and Type	Overload name	Description
public IAsyncResult	BeginExecuteNonQuery() [page 61]	Initiates the asynchronous execution of a SQL statement or stored procedure that is described by this SACommand.
public IAsyncResult	BeginExecuteNonQuery(AsyncCallback, object) [page 62]	Initiates the asynchronous execution of a SQL statement or stored procedure that is described by this SACommand, given a callback procedure and state information.

In this section:

BeginExecuteNonQuery() Method [page 61]
Initiates the asynchronous execution of a SQL statement or stored procedure that is described by this SACommand.

BeginExecuteNonQuery(AsyncCallback, object) Method [page 62]
Initiates the asynchronous execution of a SQL statement or stored procedure that is described by this SACommand, given a callback procedure and state information.

1.4.2.1 BeginExecuteNonQuery() Method

Initiates the asynchronous execution of a SQL statement or stored procedure that is described by this SACommand.

Syntax

Visual Basic

```
Public Function BeginExecuteNonQuery () As IAsyncResult
```

C#

```
public IAsyncResult BeginExecuteNonQuery ()
```

Returns

A System.IAsyncResult that can be used to poll, wait for results, or both; this value is also needed when invoking EndExecuteNonQuery(IAsyncResult), which returns the number of affected rows.

Exceptions

SAException class Any error that occurred while executing the command text.

Remarks

For asynchronous command, the order of parameters must be consistent with CommandText.

Related Information

EndExecuteNonQuery(IAsyncResult) Method [page 70]

1.4.2.2 BeginExecuteNonQuery(AsyncCallback, object) Method

Initiates the asynchronous execution of a SQL statement or stored procedure that is described by this SACommand, given a callback procedure and state information.

Syntax

Visual Basic

```
Public Function BeginExecuteNonQuery (
    ByVal callback As AsyncCallback,
    ByVal stateObject As Object
) As IAsyncResult
```

C#

```
public IAsyncResult BeginExecuteNonQuery (
    AsyncCallback callback,
    object stateObject
)
```

Parameters

callback A System.AsyncCallback delegate that is invoked when the command's execution has completed. Pass null (Nothing in Microsoft Visual Basic) to indicate that no callback is required.

stateObject A user-defined state object that is passed to the callback procedure. Retrieve this object from within the callback procedure using the System.IAsyncResult.AsyncState property.

Returns

A System.IAsyncResult that can be used to poll, wait for results, or both; this value is also needed when invoking EndExecuteNonQuery(IAsyncResult), which returns the number of affected rows.

Exceptions

SAException class Any error that occurred while executing the command text.

Remarks

For asynchronous command, the order of parameters must be consistent with CommandText.

Related Information

EndExecuteNonQuery(IAsyncResult) Method [page 70]

1.4.3 BeginExecuteReader Method

Initiates the asynchronous execution of a SQL statement or stored procedure that is described by this SACommand, and retrieves one or more result sets from the database server.

Overload list

Modifier and Type	Overload name	Description
public IAsyncResult	BeginExecuteReader() [page 64]	Initiates the asynchronous execution of a SQL statement or stored procedure that is described by this SACommand, and retrieves one or more result sets from the database server.
public IAsyncResult	BeginExecuteReader(AsyncCallback, object) [page 65]	Initiates the asynchronous execution of a SQL statement that is described by the SACommand object, and retrieves the result set, given a callback procedure and state information.
public IAsyncResult	BeginExecuteReader(AsyncCallback, object, CommandBehavior) [page 66]	Initiates the asynchronous execution of a SQL statement or stored procedure that is described by this SACommand, and retrieves one or more result sets from the database server.
public IAsyncResult	BeginExecuteReader(CommandBehavior) [page 67]	Initiates the asynchronous execution of a SQL statement or stored procedure that is described by this SACommand, and retrieves one or more result sets from the database server.

In this section:

BeginExecuteReader() Method [page 64]
Initiates the asynchronous execution of a SQL statement or stored procedure that is described by this SACommand, and retrieves one or more result sets from the database server.

BeginExecuteReader(AsyncCallback, object) Method [page 65]
Initiates the asynchronous execution of a SQL statement that is described by the SACommand object, and retrieves the result set, given a callback procedure and state information.

BeginExecuteReader(AsyncCallback, object, CommandBehavior) Method [page 66]
Initiates the asynchronous execution of a SQL statement or stored procedure that is described by this SACommand, and retrieves one or more result sets from the database server.

BeginExecuteReader(CommandBehavior) Method [page 67]
Initiates the asynchronous execution of a SQL statement or stored procedure that is described by this SACommand, and retrieves one or more result sets from the database server.

1.4.3.1 BeginExecuteReader() Method

Initiates the asynchronous execution of a SQL statement or stored procedure that is described by this SACommand, and retrieves one or more result sets from the database server.

Syntax

Visual Basic

```
Public Function BeginExecuteReader () As IAsyncResult
```

C#

```
public IAsyncResult BeginExecuteReader ()
```

Returns

A System.IAsyncResult that can be used to poll, wait for results, or both; this value is also needed when invoking EndExecuteReader(IAsyncResult), which returns an SADataReader object that can be used to retrieve the returned rows.

Exceptions

SAException class Any error that occurred while executing the command text.

Remarks

For asynchronous command, the order of parameters must be consistent with CommandText.

Related Information

EndExecuteReader(IAsyncResult) Method [page 73]
SADataReader Class [page 205]

1.4.3.2 BeginExecuteReader(AsyncCallback, object) Method

Initiates the asynchronous execution of a SQL statement that is described by the SACommand object, and retrieves the result set, given a callback procedure and state information.

Syntax

Visual Basic

```
Public Function BeginExecuteReader (
    ByVal callback As AsyncCallback,
    ByVal stateObject As Object
) As IAsyncResult
```

C#

```
public IAsyncResult BeginExecuteReader (
    AsyncCallback callback,
    object stateObject
)
```

Parameters

callback A System.AsyncCallback delegate that is invoked when the command's execution has completed. Pass null (Nothing in Microsoft Visual Basic) to indicate that no callback is required.

stateObject A user-defined state object that is passed to the callback procedure. Retrieve this object from within the callback procedure using the System.IAsyncResult.AsyncState property.

Returns

A System.IAsyncResult that can be used to poll, wait for results, or both; this value is also needed when invoking EndExecuteReader(IAsyncResult), which returns an SADataReader object that can be used to retrieve the returned rows.

Exceptions

SAException class Any error that occurred while executing the command text.

Remarks

For asynchronous command, the order of parameters must be consistent with CommandText.

Related Information

EndExecuteReader(IAsyncResult) Method [page 73]
SADataReader Class [page 205]

1.4.3.3 BeginExecuteReader(AsyncCallback, object, CommandBehavior) Method

Initiates the asynchronous execution of a SQL statement or stored procedure that is described by this SACommand, and retrieves one or more result sets from the database server.

Syntax

Visual Basic

```
Public Function BeginExecuteReader (
    ByVal callback As AsyncCallback,
    ByVal stateObject As Object,
    ByVal behavior As CommandBehavior
) As IAsyncResult
```

C#

```
public IAsyncResult BeginExecuteReader (
    AsyncCallback callback,
    object stateObject,
    CommandBehavior behavior
)
```

Parameters

callback A System.AsyncCallback delegate that is invoked when the command's execution has completed. Pass null (Nothing in Microsoft Visual Basic) to indicate that no callback is required.

stateObject A user-defined state object that is passed to the callback procedure. Retrieve this object from within the callback procedure using the System.IAsyncResult.AsyncState property.

behavior A bitwise combination of System.Data.CommandBehavior flags describing the results of the query and its effect on the connection.

Returns

A System.IAsyncResult that can be used to poll, wait for results, or both; this value is also needed when invoking EndExecuteReader(IAsyncResult), which returns an SADataReader object that can be used to retrieve the returned rows.

Exceptions

SAException class Any error that occurred while executing the command text.

Remarks

For asynchronous command, the order of parameters must be consistent with CommandText.

Related Information

EndExecuteReader(IAsyncResult) Method [page 73]
SADataReader Class [page 205]

1.4.3.4 BeginExecuteReader(CommandBehavior) Method

Initiates the asynchronous execution of a SQL statement or stored procedure that is described by this SACommand, and retrieves one or more result sets from the database server.

Syntax

Visual Basic

```
Public Function BeginExecuteReader (ByVal behavior As CommandBehavior) As
IAsyncResult
```

C#

```
public IAsyncResult BeginExecuteReader (CommandBehavior behavior)
```

Parameters

behavior A bitwise combination of System.Data.CommandBehavior flags describing the results of the query and its effect on the connection.

Returns

A System.IAsyncResult that can be used to poll, wait for results, or both; this value is also needed when invoking EndExecuteReader(IAsyncResult), which returns an SADataReader object that can be used to retrieve the returned rows.

Exceptions

SAException class Any error that occurred while executing the command text.

Remarks

For asynchronous command, the order of parameters must be consistent with CommandText.

Related Information

EndExecuteReader(IAsyncResult) Method [page 73]
SADataReader Class [page 205]

1.4.4 Cancel() Method

Cancels the execution of an SACommand object.

Syntax

Visual Basic

```
Public Overrides Sub Cancel ()
```

C#

```
public override void Cancel ()
```

Remarks

If there is nothing to cancel, then nothing happens. If there is a command in process, then a "Statement interrupted by user" exception is thrown.

1.4.5 CreateDbParameter() Method

Creates a new instance of a System.Data.Common.DbParameter object.

Syntax

Visual Basic

```
Protected Overrides Function CreateDbParameter () As DbParameter
```

C#

```
protected override DbParameter CreateDbParameter ()
```

Returns

A System.Data.Common.DbParameter object.

1.4.6 CreateParameter() Method

Provides an SAParameter object for supplying parameters to SACommand objects.

Syntax

Visual Basic

```
Public Shadows Function CreateParameter () As SAParameter
```

C#

```
public new SAParameter CreateParameter ()
```

Returns

A new parameter, as an SAParameter object.

Remarks

Stored procedures and some other SQL statements can take parameters, indicated in the text of a statement by a question mark (?).

The CreateParameter method provides an SAParameter object. Set properties on the SAParameter object to specify the value, data type, and so on for the parameter.

Related Information

SAParameter Class [page 268]

1.4.7 Dispose(bool) Method

Frees the resources associated with the object.

Syntax

Visual Basic

```
Protected Overrides Sub Dispose (ByVal disposing As Boolean)
```

C#

```
protected override void Dispose (bool disposing)
```

1.4.8 EndExecuteNonQuery(IAsyncResult) Method

Finishes asynchronous execution of a SQL statement or stored procedure.

Syntax

Visual Basic

```
Public Function EndExecuteNonQuery (ByVal asyncResult As IAsyncResult) As
Integer
```

C#

```
public unsafe int EndExecuteNonQuery (IAsyncResult asyncResult)
```

Parameters

asyncResult The IAsyncResult returned by the call to SACommand.BeginExecuteNonQuery.

Returns

The number of rows affected (the same behavior as SACommand.ExecuteNonQuery).

Exceptions

ArgumentException The asyncResult parameter is null (Nothing in Microsoft Visual Basic).
InvalidOperationException The SACommand.EndExecuteNonQuery(IAsyncResult) was called more than once for a single command execution, or the method was mismatched against its execution method.

Remarks

Call EndExecuteNonQuery once for every call to BeginExecuteNonQuery. The call must be after BeginExecuteNonQuery has returned. ADO.NET is not thread safe; it is your responsibility to ensure that BeginExecuteNonQuery has returned. The IAsyncResult passed to EndExecuteNonQuery must be the same as the one returned from the BeginExecuteNonQuery call that is being completed. It is an error to call EndExecuteNonQuery to end a call to BeginExecuteReader, and vice versa.

If an error occurs while executing the command, then the exception is thrown when EndExecuteNonQuery is called.

There are four ways to wait for execution to complete:

(1) Call EndExecuteNonQuery.

Calling EndExecuteNonQuery blocks until the command completes. For example:

```
SAConnection conn = new SAConnection("DSN=SQL Anywhere 17 Demo");
conn.Open();
SACommand cmd = new SACommand(
   "UPDATE Departments"
    + " SET DepartmentName = 'Engineering'"
    + " WHERE DepartmentID=100",
   conn );
IAsyncResult res = cmd.BeginExecuteNonQuery();
// perform other work
// this will block until the command completes
int rowCount = cmd.EndExecuteNonQuery( res );
```

(2) Poll the IsCompleted property of the IAsyncResult.

For example:

```
SAConnection conn = new SAConnection("DSN=SQL Anywhere 17 Demo");
conn.Open();
SACommand cmd = new SACommand(
   "UPDATE Departments"
    + " SET DepartmentName = 'Engineering'"
    + " WHERE DepartmentID=100",
   conn );
IAsyncResult res = cmd.BeginExecuteNonQuery();
while( !res.IsCompleted ) {
   // do other work
```

```
}
// this will not block because the command is finished
int rowCount = cmd.EndExecuteNonQuery( res );
```

(3) Use the IAsyncResult.AsyncWaitHandle property to get a synchronization object, and wait on that.

For example:

```
SAConnection conn = new SAConnection("DSN=SQL Anywhere 17 Demo");
conn.Open();
SACommand cmd = new SACommand(
   "UPDATE Departments"
    + " SET DepartmentName = 'Engineering'"
    + " WHERE DepartmentID=100",
   conn );
IAsyncResult res = cmd.BeginExecuteNonQuery();
// perform other work
WaitHandle wh = res.AsyncWaitHandle;
wh.WaitOne();
// this will not block because the command is finished
int rowCount = cmd.EndExecuteNonQuery( res );
```

(4) Specify a callback function when calling BeginExecuteNonQuery.

For example:

```
private void callbackFunction( IAsyncResult ar ) {
   SACommand cmd = (SACommand) ar.AsyncState;
   // this won't block since the command has completed
   int rowCount = cmd.EndExecuteNonQuery( ar );
}
// elsewhere in the code
private void DoStuff() {
   SAConnection conn = new SAConnection("DSN=SQL Anywhere 17 Demo");
   conn.Open();
   SACommand cmd = new SACommand(
    "UPDATE Departments"
        + " SET DepartmentName = 'Engineering'"
        + " WHERE DepartmentID=100",
    conn );
   IAsyncResult res = cmd.BeginExecuteNonQuery( callbackFunction, cmd );
   // perform other work.  The callback function is
   // called when the command completes
}
```

The callback function executes in a separate thread, so the usual caveats related to updating the user interface in a threaded program apply.

Related Information

BeginExecuteNonQuery() Method [page 61]

1.4.9 EndExecuteReader(IAsyncResult) Method

Finishes asynchronous execution of a SQL statement or stored procedure, returning the requested SADataReader.

Syntax

Visual Basic

```
Public Function EndExecuteReader (ByVal asyncResult As IAsyncResult) As
SADataReader
```

C#

```
public unsafe SADataReader EndExecuteReader (IAsyncResult asyncResult)
```

Parameters

asyncResult The IAsyncResult returned by the call to SACommand.BeginExecuteReader.

Returns

An SADataReader object that can be used to retrieve the requested rows (the same behavior as SACommand.ExecuteReader).

Exceptions

ArgumentException The asyncResult parameter is null (Nothing in Microsoft Visual Basic)
InvalidOperationException The SACommand.EndExecuteReader(IAsyncResult) was called more than once for a single command execution, or the method was mismatched against its execution method.

Remarks

Call EndExecuteReader once for every call to BeginExecuteReader. The call must be after BeginExecuteReader has returned. ADO.NET is not thread safe; it is your responsibility to ensure that BeginExecuteReader has returned. The IAsyncResult passed to EndExecuteReader must be the same as the one returned from the BeginExecuteReader call that is being completed. It is an error to call EndExecuteReader to end a call to BeginExecuteNonQuery, and vice versa.

If an error occurs while executing the command, then the exception is thrown when EndExecuteReader is called.

There are four ways to wait for execution to complete:

(1) Call EndExecuteReader.

Calling EndExecuteReader blocks until the command completes. For example:

```
SAConnection conn = new SAConnection("DSN=SQL Anywhere 17 Demo");
conn.Open();
SACommand cmd = new SACommand( "SELECT * FROM Departments", conn );
IAsyncResult res = cmd.BeginExecuteReader();
// perform other work
// this blocks until the command completes
SADataReader reader = cmd.EndExecuteReader( res );
```

(2) Poll the IsCompleted property of the IAsyncResult.

For example:

```
SAConnection conn = new SAConnection("DSN=SQL Anywhere 17 Demo");
conn.Open();
SACommand cmd = new SACommand( "SELECT * FROM Departments", conn );
IAsyncResult res = cmd.BeginExecuteReader();
while( !res.IsCompleted ) {
   // do other work
}
// this does not block because the command is finished
SADataReader reader = cmd.EndExecuteReader( res );
```

(3) Use the IAsyncResult.AsyncWaitHandle property to get a synchronization object, and wait on that.

For example:

```
SAConnection conn = new SAConnection("DSN=SQL Anywhere 17 Demo");
conn.Open();
SACommand cmd = new SACommand( "SELECT * FROM Departments", conn );
IAsyncResult res = cmd.BeginExecuteReader();
// perform other work
WaitHandle wh = res.AsyncWaitHandle;
wh.WaitOne();
// this does not block because the command is finished
SADataReader reader = cmd.EndExecuteReader( res );
```

(4) Specify a callback function when calling BeginExecuteReader

For example:

```
private void callbackFunction( IAsyncResult ar ) {
   SACommand cmd = (SACommand) ar.AsyncState;
   // this does not block since the command has completed
   SADataReader reader = cmd.EndExecuteReader();
}
// elsewhere in the code
private void DoStuff() {
   SAConnection conn = new SAConnection("DSN=SQL Anywhere 17 Demo");
   conn.Open();
   SACommand cmd = new SACommand( "SELECT * FROM Departments", conn );
   IAsyncResult res = cmd.BeginExecuteReader( callbackFunction, cmd );
   // perform other work.  The callback function is
   // called when the command completes
}
```

The callback function executes in a separate thread, so the usual caveats related to updating the user interface in a threaded program apply.

Related Information

BeginExecuteReader() Method [page 64]
SADataReader Class [page 205]

1.4.10 ExecuteDbDataReader(CommandBehavior) Method

Executes the command text against the connection.

Syntax

Visual Basic

```
Protected Overrides Function ExecuteDbDataReader (ByVal behavior As
CommandBehavior) As DbDataReader
```

C#

```
protected override DbDataReader ExecuteDbDataReader (CommandBehavior
behavior)
```

Parameters

behavior An instance of System.Data.CommandBehavior.

Returns

A System.Data.Common.DbDataReader.

1.4.11 ExecuteNonQuery() Method

Executes a statement that does not return a result set, such as an INSERT, UPDATE, DELETE, or data definition statement.

Syntax

Visual Basic

```
Public Overrides Function ExecuteNonQuery () As Integer
```

C#

```
public override unsafe int ExecuteNonQuery ()
```

Returns

The number of rows affected.

Remarks

Use ExecuteNonQuery to change the data in a database without using a DataSet. Do this by executing UPDATE, INSERT, or DELETE statements.

Although ExecuteNonQuery does not return any rows, output parameters or return values that are mapped to parameters are populated with data.

For UPDATE, INSERT, and DELETE statements, the return value is the number of rows affected by the command. For all other types of statements, and for rollbacks, the return value is -1.

Related Information

ExecuteReader() Method [page 77]

1.4.12 ExecuteReader Method

Executes a SQL statement that returns a result set.

Overload list

Modifier and Type	Overload name	Description
public new SADataReader	ExecuteReader() [page 77]	Executes a SQL statement that returns a result set.
public new SADataReader	ExecuteReader(CommandBehavior) [page 77]	Executes a SQL statement that returns a result set.

In this section:

ExecuteReader() Method [page 77]
Executes a SQL statement that returns a result set.

ExecuteReader(CommandBehavior) Method [page 77]
Executes a SQL statement that returns a result set.

1.4.12.1 ExecuteReader() Method

Executes a SQL statement that returns a result set.

Syntax

Visual Basic

```
Public Shadows Function ExecuteReader () As SADataReader
```

C#

```
public new SADataReader ExecuteReader ()
```

Returns

The result set as an SADataReader object.

Remarks

The statement is the current SACommand object, with CommandText and Parameters as needed. The SADataReader object is a read-only, forward-only result set. For modifiable result sets, use an SADataAdapter.

Related Information

ExecuteNonQuery() Method [page 75]
SADataReader Class [page 205]
SADataAdapter Class [page 181]
CommandText Property [page 85]
Parameters Property [page 89]

1.4.12.2 ExecuteReader(CommandBehavior) Method

Executes a SQL statement that returns a result set.

Syntax

Visual Basic

```
Public Shadows Function ExecuteReader (ByVal behavior As CommandBehavior)
As SADataReader
```

C#

```
public new SADataReader ExecuteReader (CommandBehavior behavior)
```

Parameters

behavior One of CloseConnection, Default, KeyInfo, SchemaOnly, SequentialAccess, SingleResult, or SingleRow. For more information about this parameter, see the .NET Framework documentation for CommandBehavior Enumeration.

Returns

The result set as an SADataReader object.

Remarks

The statement is the current SACommand object, with CommandText and Parameters as needed. The SADataReader object is a read-only, forward-only result set. For modifiable result sets, use an SADataAdapter.

Related Information

ExecuteNonQuery() Method [page 75]
SADataReader Class [page 205]
SADataAdapter Class [page 181]
CommandText Property [page 85]
Parameters Property [page 89]

1.4.13 ExecuteReaderAsync Method

An asynchronous version of SACommand.ExecuteReader, which Executes a SQL statement that returns a result set.

Overload list

Modifier and Type	Overload name	Description
public new Task< SADataReader >	ExecuteReaderAsync() [page 80]	An asynchronous version of SACommand.ExecuteReader, which Executes a SQL statement that returns a result set.
public new Task< SADataReader >	ExecuteReaderAsync(CancellationToken) [page 80]	An asynchronous version of SACommand.ExecuteReader, which Executes a SQL statement that returns a result set.
public new Task< SADataReader >	ExecuteReaderAsync(CommandBehavior) [page 81]	An asynchronous version of SACommand.ExecuteReader, which Executes a SQL statement that returns a result set.
public new Task< SADataReader >	ExecuteReaderAsync(CommandBehavior, CancellationToken) [page 82]	An asynchronous version of SACommand.ExecuteReader, which Executes a SQL statement that returns a result set.

In this section:

ExecuteReaderAsync() Method [page 80]
An asynchronous version of SACommand.ExecuteReader, which Executes a SQL statement that returns a result set.

ExecuteReaderAsync(CancellationToken) Method [page 80]
An asynchronous version of SACommand.ExecuteReader, which Executes a SQL statement that returns a result set.

ExecuteReaderAsync(CommandBehavior) Method [page 81]
An asynchronous version of SACommand.ExecuteReader, which Executes a SQL statement that returns a result set.

ExecuteReaderAsync(CommandBehavior, CancellationToken) Method [page 82]
An asynchronous version of SACommand.ExecuteReader, which Executes a SQL statement that returns a result set.

1.4.13.1 ExecuteReaderAsync() Method

An asynchronous version of SACommand.ExecuteReader, which Executes a SQL statement that returns a result set.

Syntax

Visual Basic

```
Public Shadows Function ExecuteReaderAsync () As Task< SADataReader >
```

C#

```
public new Task< SADataReader > ExecuteReaderAsync ()
```

Returns

A task representing the asynchronous operation.

Exceptions

SAException class The database server returned an error while executing the command text.

Remarks

Exceptions are reported via the returned Task object.

1.4.13.2 ExecuteReaderAsync(CancellationToken) Method

An asynchronous version of SACommand.ExecuteReader, which Executes a SQL statement that returns a result set.

Syntax

Visual Basic

```
Public Shadows Function ExecuteReaderAsync (ByVal cancellationToken As
CancellationToken) As Task< SADataReader >
```

C#

```
public new Task< SADataReader > ExecuteReaderAsync (CancellationToken
cancellationToken)
```

Parameters

cancellationToken The cancellation instruction.

Returns

A task representing the asynchronous operation.

Exceptions

SAException class The database server returned an error while executing the command text.

Remarks

The cancellation token can be used to request that the operation be abandoned before the command timeout elapses. Exceptions are reported via the returned Task object.

1.4.13.3 ExecuteReaderAsync(CommandBehavior) Method

An asynchronous version of SACommand.ExecuteReader, which Executes a SQL statement that returns a result set.

Syntax

Visual Basic

```
Public Shadows Function ExecuteReaderAsync (ByVal behavior As
CommandBehavior) As Task< SADataReader >
```

C#

```
public new Task< SADataReader > ExecuteReaderAsync (CommandBehavior
behavior)
```

Parameters

behavior Options for statement execution and data retrieval.

Returns

A task representing the asynchronous operation.

Exceptions

SAException class The database server returned an error while executing the command text.

Remarks

Exceptions are reported via the returned Task object.

1.4.13.4 ExecuteReaderAsync(CommandBehavior, CancellationToken) Method

An asynchronous version of SACommand.ExecuteReader, which Executes a SQL statement that returns a result set.

Syntax

Visual Basic

```
Public Shadows Function ExecuteReaderAsync (
    ByVal behavior As CommandBehavior,
    ByVal cancellationToken As CancellationToken
) As Task< SADataReader >
```

C#

```
public new Task< SADataReader > ExecuteReaderAsync (
    CommandBehavior behavior,
    CancellationToken cancellationToken
)
```

Parameters

behavior Options for statement execution and data retrieval.

cancellationToken The cancellation instruction.

Returns

A task representing the asynchronous operation.

Exceptions

SAException class The database server returned an error while executing the command text.

Remarks

The cancellation token can be used to request that the operation be abandoned before the command timeout elapses. Exceptions are reported via the returned Task object.

1.4.14 ExecuteScalar() Method

Executes a statement that returns a single value.

Syntax

Visual Basic

```
Public Overrides Function ExecuteScalar () As Object
```

C#

```
public override object ExecuteScalar ()
```

Returns

The first column of the first row in the result set, or a null reference if the result set is empty.

Remarks

If this method is called on a query that returns multiple rows and columns, then only the first column of the first row is returned.

1.4.15 Prepare() Method

Prepares or compiles the SACommand on the data source.

Syntax

Visual Basic

```
Public Overrides Sub Prepare ()
```

C#

```
public override void Prepare ()
```

Remarks

If you call one of the ExecuteNonQuery, ExecuteReader, or ExecuteScalar methods after calling Prepare, then any parameter value that is larger than the value specified by the Size property is automatically truncated to the original specified size of the parameter, and no truncation errors are returned.

The truncation only happens for the following data types:

- CHAR
- VARCHAR
- LONG VARCHAR
- TEXT
- NCHAR
- NVARCHAR
- LONG NVARCHAR
- NTEXT
- BINARY
- LONG BINARY
- VARBINARY
- IMAGE

If the Size property is not specified, then the default value is used and the data is not truncated.

Related Information

ExecuteNonQuery() Method [page 75]
ExecuteReader() Method [page 77]
ExecuteScalar() Method [page 83]

1.4.16 ResetCommandTimeout() Method

Resets the CommandTimeout property to its default value of 30 seconds.

Syntax

Visual Basic

```
Public Sub ResetCommandTimeout ()
```

C#

```
public void ResetCommandTimeout ()
```

1.4.17 CommandText Property

Gets or sets the text of a SQL statement or stored procedure.

Syntax

Visual Basic

```
Public Overrides Property CommandText As String
```

C#

```
public override string CommandText {get;set;}
```

Remarks

The SQL statement or the name of the stored procedure to execute. The default is an empty string.

Related Information

SACommand() Constructor [page 58]

1.4.18 CommandTimeout Property

This feature is not supported by the .NET Data Provider.

Syntax

Visual Basic

```
Public Overrides Property CommandTimeout As Integer
```

C#

```
public override int CommandTimeout {get;set;}
```

Remarks

To set a request timeout, use the following example.

```
cmd.CommandText = "SET OPTION request_timeout = 30";
cmd.ExecuteNonQuery();
```

1.4.19 CommandType Property

Gets or sets the type of command represented by an SACommand.

Syntax

Visual Basic

```
Public Overrides Property CommandType As CommandType
```

C#

```
public override CommandType CommandType {get;set;}
```

Remarks

One of the System.Data.CommandType values. The default is System.Data.CommandType.Text.

Supported command types are as follows:

- System.Data.CommandType.StoredProcedure When you specify this CommandType, the command text must be the name of a stored procedure and you must supply any arguments as SAParameter objects.
- System.Data.CommandType.Text This is the default value.

When the CommandType property is set to StoredProcedure, the CommandText property should be set to the name of the stored procedure. The command executes this stored procedure when you call one of the Execute methods.

Use a question mark (?) placeholder to pass parameters. For example:

```
SELECT * FROM Customers WHERE ID = ?
```

The order in which SAParameter objects are added to the SAParameterCollection must directly correspond to the position of the question mark placeholder for the parameter.

1.4.20 Connection Property

Gets or sets the connection object to which the SACommand object applies.

Syntax

Visual Basic

```
Public Shadows Property Connection As SAConnection
```

C#

```
public new SAConnection Connection {get;set;}
```

Remarks

The default value is a null reference. In Visual Basic it is Nothing.

1.4.21 DbConnection Property

Gets or sets the System.Data.Common.DbConnection used by this SACommand object.

Syntax

Visual Basic

```
Protected Overrides Property DbConnection As DbConnection
```

C#

```
protected override DbConnection DbConnection {get;set;}
```

Returns

The connection to the data source.

Related Information

SACommand Class [page 53]

1.4.22 DbParameterCollection Property

Gets the collection of System.Data.Common.DbParameter objects.

Syntax

Visual Basic

```
Protected ReadOnly Overrides Property DbParameterCollection As
DbParameterCollection
```

C#

```
protected override DbParameterCollection DbParameterCollection {get;}
```

Returns

The parameters of the SQL statement or stored procedure.

1.4.23 DbTransaction Property

Gets or sets the System.Data.Common.DbTransaction within which this SACommand object executes.

Syntax

Visual Basic

```
Protected Overrides Property DbTransaction As DbTransaction
```

C#

```
protected override DbTransaction DbTransaction {get;set;}
```

Returns

The transaction within which a Command object of a .NET Framework data provider executes. The default value is a null reference (Nothing in Visual Basic).

1.4.24 DesignTimeVisible Property

Gets or sets a value that indicates if the SACommand should be visible in a Windows Form Designer control.

Syntax

Visual Basic

```
Public Overrides Property DesignTimeVisible As Boolean
```

C#

```
public override bool DesignTimeVisible {get;set;}
```

Remarks

The default is true.

True if this SACommand instance should be visible; false if this instance should not be visible.

1.4.25 Parameters Property

Specifies a collection of parameters for the current statement.

Syntax

Visual Basic

```
Public ReadOnly Shadows Property Parameters As SAParameterCollection
```

C#

```
public new SAParameterCollection Parameters {get;}
```

Remarks

Use question marks in the CommandText to indicate parameters.

The parameters of the SQL statement or stored procedure. The default value is an empty collection.

When CommandType is set to Text, pass parameters using the question mark placeholder. For example:

```
SELECT * FROM Customers WHERE ID = ?
```

The order in which SAParameter objects are added to the SAParameterCollection must directly correspond to the position of the question mark placeholder for the parameter in the command text.

When the parameters in the collection do not match the requirements of the query to be executed, an error may result or an exception may be thrown.

Related Information

SAParameterCollection Class [page 284]

1.4.26 Transaction Property

Specifies the SATransaction object in which the SACommand executes.

Syntax

Visual Basic

```
Public Shadows Property Transaction As SATransaction
```

C#

```
public new SATransaction Transaction {get;set;}
```

Remarks

The default value is a null reference. In Visual Basic, this is Nothing.

You cannot set the Transaction property if it is already set to a specific value and the command is executing. If you set the transaction property to an SATransaction object that is not connected to the same SAConnection object as the SACommand object, then an exception is thrown the next time you attempt to execute a statement.

For more information, see "Transaction processing".

Related Information

SATransaction Class [page 335]

1.4.27 UpdatedRowSource Property

Gets or sets how command results are applied to the DataRow when used by the Update method of the SADataAdapter.

Syntax

Visual Basic

```
Public Overrides Property UpdatedRowSource As UpdateRowSource
```

C#

```
public override UpdateRowSource UpdatedRowSource {get;set;}
```

Remarks

One of the UpdatedRowSource values. The default value is UpdateRowSource.OutputParameters. If the command is automatically generated, then this property is UpdateRowSource.None.

UpdatedRowSource.Both, which returns both resultset and output parameters, is not supported.

1.5 SACommandBuilder Class

A way to generate single-table SQL statements that reconcile changes made to a DataSet with the data in the associated database.

Syntax

Visual Basic

```
Public NotInheritable Class SACommandBuilder Inherits
System.Data.Common.DbCommandBuilder
```

C#

```
public sealed class SACommandBuilder : System.Data.Common.DbCommandBuilder
```

Members

All members of SACommandBuilder, including inherited members.

Constructors

Modifier and Type	Constructor	Description
public	SACommandBuilder [page 94]	Initializes an SACommandBuilder object.

Methods

Modifier and Type	Method	Description
protected override void	ApplyParameterInfo(DbParameter, DataRow, StatementType, bool) [page 95]	Allows the provider implementation of System.Data.Common.DbCommandBuilder to handle additional parameter properties.
public static void	DeriveParameters(SACommand) [page 96]	Populates the Parameters collection of the specified SACommand object.
public new SACommand	GetDeleteCommand [page 97]	Returns the generated SACommand object that performs DELETE operations on the database when SADataAdapter.Update is called.
public new SACommand	GetInsertCommand [page 99]	Returns the generated SACommand object that performs INSERT operations on the database when an Update is called.
protected override string	GetParameterName [page 102]	Returns the name of the specified parameter in the format of @p#.
protected override string	GetParameterPlaceholder(int) [page 104]	Returns the placeholder for the parameter in the associated SQL statement.
protected override DataTable	GetSchemaTable(DbCommand) [page 104]	Returns the schema table for the SACommandBuilder object.
public new SACommand	GetUpdateCommand [page 105]	Returns the generated SACommand object that performs UPDATE operations on the database when an Update is called.
protected override DbCommand	InitializeCommand(DbCommand) [page 107]	Resets the System.Data.Common.DbCommand.CommandTimeout, System.Data.Common.DbCommand.Transaction, System.Data.Common.DbCommand.CommandType, and System.Data.Common.DbCommand.UpdatedRowSource properties on the System.Data.Common.DbCommand.
public override string	QuoteIdentifier(string) [page 108]	Returns the correct quoted form of an unquoted identifier, including properly escaping any embedded quotes in the identifier.
protected override void	SetRowUpdatingHandler(DbDataAdapter) [page 109]	Registers the SACommandBuilder object to handle the SADataAdapter.RowUpdating event for an SADataAdapter object.

Modifier and Type	Method	Description
public override string	UnquoteIdentifier(string) [page 109]	Returns the correct unquoted form of a quoted identifier, including properly un-escaping any embedded quotes in the identifier.

Properties

Modifier and Type	Property	Description
public new SADataAdapter	DataAdapter [page 110]	Specifies the SADataAdapter for which to generate statements.

In this section:

SACommandBuilder Constructor [page 94]
Initializes an SACommandBuilder object.

ApplyParameterInfo(DbParameter, DataRow, StatementType, bool) Method [page 95]
Allows the provider implementation of System.Data.Common.DbCommandBuilder to handle additional parameter properties.

DeriveParameters(SACommand) Method [page 96]
Populates the Parameters collection of the specified SACommand object.

GetDeleteCommand Method [page 97]
Returns the generated SACommand object that performs DELETE operations on the database when SADataAdapter.Update is called.

GetInsertCommand Method [page 99]
Returns the generated SACommand object that performs INSERT operations on the database when an Update is called.

GetParameterName Method [page 102]
Returns the name of the specified parameter in the format of @p#.

GetParameterPlaceholder(int) Method [page 104]
Returns the placeholder for the parameter in the associated SQL statement.

GetSchemaTable(DbCommand) Method [page 104]
Returns the schema table for the SACommandBuilder object.

GetUpdateCommand Method [page 105]
Returns the generated SACommand object that performs UPDATE operations on the database when an Update is called.

InitializeCommand(DbCommand) Method [page 107]
Resets the System.Data.Common.DbCommand.CommandTimeout, System.Data.Common.DbCommand.Transaction, System.Data.Common.DbCommand.CommandType, and System.Data.Common.DbCommand.UpdatedRowSource properties on the System.Data.Common.DbCommand.

QuoteIdentifier(string) Method [page 108]
Returns the correct quoted form of an unquoted identifier, including properly escaping any embedded quotes in the identifier.

SetRowUpdatingHandler(DbDataAdapter) Method [page 109]
Registers the SACommandBuilder object to handle the SADataAdapter.RowUpdating event for an SADataAdapter object.

UnquoteIdentifier(string) Method [page 109]
Returns the correct unquoted form of a quoted identifier, including properly un-escaping any embedded quotes in the identifier.

DataAdapter Property [page 110]
Specifies the SADataAdapter for which to generate statements.

1.5.1 SACommandBuilder Constructor

Initializes an SACommandBuilder object.

Overload list

Modifier and Type	Overload name	Description
public	SACommandBuilder() [page 94]	Initializes an SACommandBuilder object.
public	SACommandBuilder(SADataAdapter) [page 95]	Initializes an SACommandBuilder object.

In this section:

SACommandBuilder() Constructor [page 94]
Initializes an SACommandBuilder object.

SACommandBuilder(SADataAdapter) Constructor [page 95]
Initializes an SACommandBuilder object.

1.5.1.1 SACommandBuilder() Constructor

Initializes an SACommandBuilder object.

Syntax

Visual Basic

```
Public Sub SACommandBuilder ()
```

C#

```
public SACommandBuilder ()
```

1.5.1.2 SACommandBuilder(SADataAdapter) Constructor

Initializes an SACommandBuilder object.

Syntax

Visual Basic

```
Public Sub SACommandBuilder (ByVal adapter As SADataAdapter)
```

C#

```
public SACommandBuilder (SADataAdapter adapter)
```

Parameters

adapter An SADataAdapter object for which to generate reconciliation statements.

1.5.2 ApplyParameterInfo(DbParameter, DataRow, StatementType, bool) Method

Allows the provider implementation of System.Data.Common.DbCommandBuilder to handle additional parameter properties.

Syntax

Visual Basic

```
Protected Overrides Sub ApplyParameterInfo (
    ByVal parameter As DbParameter,
    ByVal row As DataRow,
    ByVal statementType As StatementType,
    ByVal whereClause As Boolean
)
```

C#

```
protected override void ApplyParameterInfo (
    DbParameter parameter,
    DataRow row,
    StatementType statementType,
    bool whereClause
)
```

Parameters

parameter A System.Data.Common.DbParameter to which the additional modifications are applied.
row The System.Data.DataRow from the schema table provided by SADataReader.GetSchemaTable.
statementType The type of command being generated: INSERT, UPDATE or DELETE.
whereClause The value is true if the parameter is part of the UPDATE or DELETE WHERE clause, and false if it is part of the INSERT or UPDATE values.

Related Information

GetSchemaTable() Method [page 226]

1.5.3 DeriveParameters(SACommand) Method

Populates the Parameters collection of the specified SACommand object.

Syntax

Visual Basic

```
Public Shared Sub DeriveParameters (ByVal command As SACommand)
```

C#

```
public static void DeriveParameters (SACommand command)
```

Parameters

command An SACommand object for which to derive parameters.

Remarks

This method is used for the stored procedure specified in the SACommand.

DeriveParameters overwrites any existing parameter information for the SACommand.

DeriveParameters requires an extra call to the database server. If the parameter information is known in advance, then it is more efficient to populate the Parameters collection by setting the information explicitly.

1.5.4 GetDeleteCommand Method

Returns the generated SACommand object that performs DELETE operations on the database when SADataAdapter.Update is called.

Overload list

Modifier and Type	Overload name	Description
public new SACommand	GetDeleteCommand() [page 97]	Returns the generated SACommand object that performs DELETE operations on the database when SADataAdapter.Update is called.
public new SACommand	GetDeleteCommand(bool) [page 98]	Returns the generated SACommand object that performs DELETE operations on the database when SADataAdapter.Update is called.

In this section:

GetDeleteCommand() Method [page 97]
Returns the generated SACommand object that performs DELETE operations on the database when SADataAdapter.Update is called.

GetDeleteCommand(bool) Method [page 98]
Returns the generated SACommand object that performs DELETE operations on the database when SADataAdapter.Update is called.

1.5.4.1 GetDeleteCommand() Method

Returns the generated SACommand object that performs DELETE operations on the database when SADataAdapter.Update is called.

Syntax

Visual Basic

```
Public Shadows Function GetDeleteCommand () As SACommand
```

C#

```
public new SACommand GetDeleteCommand ()
```

Returns

The automatically generated SACommand object required to perform deletions.

Remarks

The GetDeleteCommand method returns the SACommand object to be executed, so it is useful for informational or troubleshooting purposes.

Alternatively, use GetDeleteCommand as the basis of a modified command. For example, you might call GetDeleteCommand and modify the CommandTimeout value, and then explicitly set that value on the SADataAdapter.

SQL statements are first generated when the application calls Update or GetDeleteCommand. After the SQL statement is first generated, the application must explicitly call RefreshSchema if it changes the statement in any way. Otherwise, the GetDeleteCommand continues to use information from the previous statement.

1.5.4.2 GetDeleteCommand(bool) Method

Returns the generated SACommand object that performs DELETE operations on the database when SADataAdapter.Update is called.

Syntax

Visual Basic

```
Public Shadows Function GetDeleteCommand (ByVal
useColumnsForParameterNames As Boolean) As SACommand
```

C#

```
public new SACommand GetDeleteCommand (bool useColumnsForParameterNames)
```

Parameters

useColumnsForParameterNames If true, then generate parameter names matching column names if possible. If false, then generate @p1, @p2, and so on.

Returns

The automatically generated SACommand object required to perform deletions.

Remarks

The GetDeleteCommand method returns the SACommand object to be executed, so it is useful for informational or troubleshooting purposes.

Alternatively, use GetDeleteCommand as the basis of a modified command. For example, you might call GetDeleteCommand and modify the CommandTimeout value, and then explicitly set that value on the SADataAdapter.

SQL statements are first generated when the application calls Update or GetDeleteCommand. After the SQL statement is first generated, the application must explicitly call RefreshSchema if it changes the statement in any way. Otherwise, the GetDeleteCommand continues to use information from the previous statement.

1.5.5 GetInsertCommand Method

Returns the generated SACommand object that performs INSERT operations on the database when an Update is called.

Overload list

Modifier and Type	Overload name	Description
public new SACommand	GetInsertCommand() [page 100]	Returns the generated SACommand object that performs INSERT operations on the database when an Update is called.
public new SACommand	GetInsertCommand(bool) [page 101]	Returns the generated SACommand object that performs INSERT operations on the database when an Update is called.

In this section:

GetInsertCommand() Method [page 100]
Returns the generated SACommand object that performs INSERT operations on the database when an Update is called.

GetInsertCommand(bool) Method [page 101]
Returns the generated SACommand object that performs INSERT operations on the database when an Update is called.

1.5.5.1 GetInsertCommand() Method

Returns the generated SACommand object that performs INSERT operations on the database when an Update is called.

Syntax

Visual Basic

```
Public Shadows Function GetInsertCommand () As SACommand
```

C#

```
public new SACommand GetInsertCommand ()
```

Returns

The automatically generated SACommand object required to perform insertions.

Remarks

The GetInsertCommand method returns the SACommand object to be executed, so it is useful for informational or troubleshooting purposes.

Alternatively, use GetInsertCommand as the basis of a modified command. For example, you might call GetInsertCommand and modify the CommandTimeout value, and then explicitly set that value on the SADataAdapter.

SQL statements are first generated either when the application calls Update or GetInsertCommand. After the SQL statement is first generated, the application must explicitly call RefreshSchema if it changes the statement in any way. Otherwise, the GetInsertCommand continues to use information from the previous statement, which might not be correct.

Related Information

GetDeleteCommand() Method [page 97]

1.5.5.2 GetInsertCommand(bool) Method

Returns the generated SACommand object that performs INSERT operations on the database when an Update is called.

Syntax

Visual Basic

```
Public Shadows Function GetInsertCommand (ByVal
useColumnsForParameterNames As Boolean) As SACommand
```

C#

```
public new SACommand GetInsertCommand (bool useColumnsForParameterNames)
```

Parameters

useColumnsForParameterNames If true, then generate parameter names matching column names if possible. If false, then generate @p1, @p2, and so on.

Returns

The automatically generated SACommand object required to perform insertions.

Remarks

The GetInsertCommand method returns the SACommand object to be executed, so it is useful for informational or troubleshooting purposes.

Alternatively, use GetInsertCommand as the basis of a modified command. For example, you might call GetInsertCommand and modify the CommandTimeout value, and then explicitly set that value on the SADataAdapter.

SQL statements are first generated either when the application calls Update or GetInsertCommand. After the SQL statement is first generated, the application must explicitly call RefreshSchema if it changes the statement in any way. Otherwise, the GetInsertCommand continues to use information from the previous statement, which might not be correct.

Related Information

GetDeleteCommand() Method [page 97]

1.5.6 GetParameterName Method

Returns the name of the specified parameter in the format of @p#.

Overload list

Modifier and Type	Overload name	Description
protected override string	GetParameterName(int) [page 102]	Returns the name of the specified parameter in the format of @p#.
protected override string	GetParameterName(string) [page 103]	Returns the full parameter name, given the partial parameter name.

In this section:

GetParameterName(int) Method [page 102]
Returns the name of the specified parameter in the format of @p#.

GetParameterName(string) Method [page 103]
Returns the full parameter name, given the partial parameter name.

1.5.6.1 GetParameterName(int) Method

Returns the name of the specified parameter in the format of @p#.

Syntax

Visual Basic

```
Protected Overrides Function GetParameterName (ByVal index As Integer) As
String
```

C#

```
protected override string GetParameterName (int index)
```

Parameters

index The number to be included as part of the parameter's name.

Returns

The name of the parameter with the specified number appended as part of the parameter name.

Remarks

Use when building a custom command builder.

1.5.6.2 GetParameterName(string) Method

Returns the full parameter name, given the partial parameter name.

Syntax

Visual Basic

```
Protected Overrides Function GetParameterName (ByVal parameterName As
String) As String
```

C#

```
protected override string GetParameterName (string parameterName)
```

Parameters

parameterName The partial name of the parameter.

Returns

The full parameter name corresponding to the partial parameter name requested.

1.5.7 GetParameterPlaceholder(int) Method

Returns the placeholder for the parameter in the associated SQL statement.

Syntax

Visual Basic

```
Protected Overrides Function GetParameterPlaceholder (ByVal index As
Integer) As String
```

C#

```
protected override string GetParameterPlaceholder (int index)
```

Parameters

index The number to be included as part of the parameter's name.

Returns

The name of the parameter with the specified number appended.

1.5.8 GetSchemaTable(DbCommand) Method

Returns the schema table for the SACommandBuilder object.

Syntax

Visual Basic

```
Protected Overrides Function GetSchemaTable (ByVal sourceCommand As
DbCommand) As DataTable
```

C#

```
protected override DataTable GetSchemaTable (DbCommand sourceCommand)
```

Parameters

sourceCommand The System.Data.Common.DbCommand for which to retrieve the corresponding schema table.

Returns

A System.Data.DataTable that represents the schema for the specific System.Data.Common.DbCommand.

Related Information

SACommandBuilder Class [page 91]

1.5.9 GetUpdateCommand Method

Returns the generated SACommand object that performs UPDATE operations on the database when an Update is called.

Overload list

Modifier and Type	Overload name	Description
public new SACommand	GetUpdateCommand() [page 106]	Returns the generated SACommand object that performs UPDATE operations on the database when an Update is called.
public new SACommand	GetUpdateCommand(bool) [page 106]	Returns the generated SACommand object that performs UPDATE operations on the database when an Update is called.

In this section:

GetUpdateCommand() Method [page 106]
Returns the generated SACommand object that performs UPDATE operations on the database when an Update is called.

GetUpdateCommand(bool) Method [page 106]
Returns the generated SACommand object that performs UPDATE operations on the database when an Update is called.

1.5.9.1 GetUpdateCommand() Method

Returns the generated SACommand object that performs UPDATE operations on the database when an Update is called.

Syntax

Visual Basic

```
Public Shadows Function GetUpdateCommand () As SACommand
```

C#

```
public new SACommand GetUpdateCommand ()
```

Returns

The automatically generated SACommand object required to perform updates.

Remarks

The GetUpdateCommand method returns the SACommand object to be executed, so it is useful for informational or troubleshooting purposes.

Alternatively, use GetUpdateCommand as the basis of a modified command. For example, you might call GetUpdateCommand and modify the CommandTimeout value, and then explicitly set that value on the SADataAdapter.

SQL statements are first generated when the application calls Update or GetUpdateCommand. After the SQL statement is first generated, the application must explicitly call RefreshSchema if it changes the statement in any way. Otherwise, the GetUpdateCommand continues to use information from the previous statement, which might not be correct.

1.5.9.2 GetUpdateCommand(bool) Method

Returns the generated SACommand object that performs UPDATE operations on the database when an Update is called.

Syntax

Visual Basic

```
Public Shadows Function GetUpdateCommand (ByVal
useColumnsForParameterNames As Boolean) As SACommand
```

C#

```
public new SACommand GetUpdateCommand (bool useColumnsForParameterNames)
```

Parameters

useColumnsForParameterNames If true, then generate parameter names matching column names if possible. If false, then generate @p1, @p2, and so on.

Returns

The automatically generated SACommand object required to perform updates.

Remarks

The GetUpdateCommand method returns the SACommand object to be executed, so it is useful for informational or troubleshooting purposes.

Alternatively, use GetUpdateCommand as the basis of a modified command. For example, you might call GetUpdateCommand and modify the CommandTimeout value, and then explicitly set that value on the SADataAdapter.

SQL statements are first generated when the application calls Update or GetUpdateCommand. After the SQL statement is first generated, the application must explicitly call RefreshSchema if it changes the statement in any way. Otherwise, the GetUpdateCommand continues to use information from the previous statement, which might not be correct.

1.5.10 InitializeCommand(DbCommand) Method

Resets the System.Data.Common.DbCommand.CommandTimeout, System.Data.Common.DbCommand.Transaction, System.Data.Common.DbCommand.CommandType, and System.Data.Common.DbCommand.UpdatedRowSource properties on the System.Data.Common.DbCommand.

Syntax

Visual Basic

```
Protected Overrides Function InitializeCommand (ByVal command As
DbCommand) As DbCommand
```

C#

```
protected override DbCommand InitializeCommand (DbCommand command)
```

Parameters

command The System.Data.Common.DbCommand to be used by the command builder for the corresponding insert, update, or delete command.

Returns

A System.Data.Common.DbCommand instance to use for each insert, update, or delete operation. Passing a null value allows the InitializeCommand method to create a System.Data.Common.DbCommand object based on the SELECT statement associated with the SACommandBuilder object.

Related Information

SACommandBuilder Class [page 91]

1.5.11 QuoteIdentifier(string) Method

Returns the correct quoted form of an unquoted identifier, including properly escaping any embedded quotes in the identifier.

Syntax

Visual Basic

```
Public Overrides Function QuoteIdentifier (ByVal unquotedIdentifier As
String) As String
```

C#

```
public override string QuoteIdentifier (string unquotedIdentifier)
```

Parameters

unquotedIdentifier The string representing the unquoted identifier that will have to be quoted.

Returns

Returns a string representing the quoted form of an unquoted identifier with embedded quotes properly escaped.

1.5.12 SetRowUpdatingHandler(DbDataAdapter) Method

Registers the SACommandBuilder object to handle the SADataAdapter.RowUpdating event for an SADataAdapter object.

Syntax

Visual Basic

```
Protected Overrides Sub SetRowUpdatingHandler (ByVal adapter As
DbDataAdapter)
```

C#

```
protected override void SetRowUpdatingHandler (DbDataAdapter adapter)
```

Parameters

adapter The SADataAdapter object to be used for the update.

Related Information

SACommandBuilder Class [page 91]
SADataAdapter Class [page 181]
RowUpdating Event [page 204]

1.5.13 UnquoteIdentifier(string) Method

Returns the correct unquoted form of a quoted identifier, including properly un-escaping any embedded quotes in the identifier.

Syntax

Visual Basic

```
Public Overrides Function UnquoteIdentifier (ByVal quotedIdentifier As
String) As String
```

C#

```
public override string UnquoteIdentifier (string quotedIdentifier)
```

Parameters

quotedIdentifier The string representing the quoted identifier that will have its embedded quotes removed.

Returns

Returns a string representing the unquoted form of a quoted identifier with embedded quotes properly un-escaped.

1.5.14 DataAdapter Property

Specifies the SADataAdapter for which to generate statements.

Syntax

Visual Basic

```
Public Shadows Property DataAdapter As SADataAdapter
```

C#

```
public new SADataAdapter DataAdapter {get;set;}
```

Remarks

An SADataAdapter object.

When you create a new instance of SACommandBuilder, any existing SACommandBuilder that is associated with this SADataAdapter is released.

1.6 SACommLinksOptionsBuilder Class

Provides a simple way to create and manage the CommLinks options portion of connection strings used by the SAConnection class.

Syntax

Visual Basic

```
Public NotInheritable Class SACommLinksOptionsBuilder
```

C#

```
public sealed class SACommLinksOptionsBuilder
```

Members

All members of SACommLinksOptionsBuilder, including inherited members.

Constructors

Modifier and Type	Constructor	Description
public	SACommLinksOptionsBuilder [page 113]	Initializes an SACommLinksOptions-Builder object.

Methods

Modifier and Type	Method	Description
public bool	GetUseLongNameAsKeyword() [page 114]	Gets a boolean value that indicates whether long connection parameter names are used in the connection string.
public void	SetUseLongNameAsKeyword(bool) [page 115]	Sets a boolean value that indicates whether long connection parameter names are used in the connection string.
public override string	ToString() [page 116]	Converts the SACommLinksOptions-Builder object to a string representation.

Properties

Modifier and Type	Property	Description
public bool	All [page 116]	Gets or sets the ALL CommLinks option.
public string	ConnectionString [page 117]	Gets or sets the connection string being built.

Modifier and Type	Property	Description
public bool	SharedMemory [page 117]	Gets or sets the SharedMemory protocol.
public SATcpOptionsBuilder	TcpOptionsBuilder [page 117]	Gets or sets an SATcpOptionsBuilder object used to create a TCP options string.
public string	TcpOptionsString [page 118]	Gets or sets a string of TCP options.

Remarks

For a list of connection parameters, see "Connection parameters".

In this section:

SACommLinksOptionsBuilder Constructor [page 113]
Initializes an SACommLinksOptionsBuilder object.

GetUseLongNameAsKeyword() Method [page 114]
Gets a boolean value that indicates whether long connection parameter names are used in the connection string.

SetUseLongNameAsKeyword(bool) Method [page 115]
Sets a boolean value that indicates whether long connection parameter names are used in the connection string.

ToString() Method [page 116]
Converts the SACommLinksOptionsBuilder object to a string representation.

All Property [page 116]
Gets or sets the ALL CommLinks option.

ConnectionString Property [page 117]
Gets or sets the connection string being built.

SharedMemory Property [page 117]
Gets or sets the SharedMemory protocol.

TcpOptionsBuilder Property [page 117]
Gets or sets an SATcpOptionsBuilder object used to create a TCP options string.

TcpOptionsString Property [page 118]
Gets or sets a string of TCP options.

1.6.1 SACommLinksOptionsBuilder Constructor

Initializes an SACommLinksOptionsBuilder object.

Overload list

Modifier and Type	Overload name	Description
public	SACommLinksOptionsBuilder() [page 113]	Initializes an SACommLinksOptions-Builder object.
public	SACommLinksOptionsBuilder(string) [page 114]	Initializes an SACommLinksOptions-Builder object.

In this section:

SACommLinksOptionsBuilder() Constructor [page 113]
Initializes an SACommLinksOptionsBuilder object.

SACommLinksOptionsBuilder(string) Constructor [page 114]
Initializes an SACommLinksOptionsBuilder object.

1.6.1.1 SACommLinksOptionsBuilder() Constructor

Initializes an SACommLinksOptionsBuilder object.

Syntax

Visual Basic

```
Public Sub SACommLinksOptionsBuilder ()
```

C#

```
public SACommLinksOptionsBuilder ()
```

Example

The following statement initializes an SACommLinksOptionsBuilder object.

```
SACommLinksOptionsBuilder commLinks = new SACommLinksOptionsBuilder( );
```

1.6.1.2 SACommLinksOptionsBuilder(string) Constructor

Initializes an SACommLinksOptionsBuilder object.

Syntax

Visual Basic

```
Public Sub SACommLinksOptionsBuilder (ByVal options As String)
```

C#

```
public SACommLinksOptionsBuilder (string options)
```

Parameters

options A CommLinks connection parameter string. For a list of connection parameters, see "Connection parameters".

Example

The following statement initializes an SACommLinksOptionsBuilder object.

```
SACommLinksOptionsBuilder commLinks =
 new SACommLinksOptionsBuilder("TCPIP(DoBroadcast=ALL;Timeout=20)");
```

1.6.2 GetUseLongNameAsKeyword() Method

Gets a boolean value that indicates whether long connection parameter names are used in the connection string.

Syntax

Visual Basic

```
Public Function GetUseLongNameAsKeyword () As Boolean
```

C#

```
public bool GetUseLongNameAsKeyword ()
```

Returns

True if long connection parameter names are used to build connection strings; false otherwise.

Remarks

Connection parameters have both long and short forms of their names. For example, to specify the name of an ODBC data source in your connection string, use either of the following values: DataSourceName or DSN. By default, long connection parameter names are used to build connection strings.

Related Information

SetUseLongNameAsKeyword(bool) Method [page 115]

1.6.3 SetUseLongNameAsKeyword(bool) Method

Sets a boolean value that indicates whether long connection parameter names are used in the connection string.

Syntax

Visual Basic

```
Public Sub SetUseLongNameAsKeyword (ByVal useLongNameAsKeyword As Boolean)
```

C#

```
public void SetUseLongNameAsKeyword (bool useLongNameAsKeyword)
```

Parameters

useLongNameAsKeyword A boolean value that indicates whether the long connection parameter name is used in the connection string.

Remarks

Long connection parameter names are used by default.

Related Information

GetUseLongNameAsKeyword() Method [page 114]

1.6.4 ToString() Method

Converts the SACommLinksOptionsBuilder object to a string representation.

Syntax

Visual Basic

```
Public Overrides Function ToString () As String
```

C#

```
public override string ToString ()
```

Returns

The options string being built.

1.6.5 All Property

Gets or sets the ALL CommLinks option.

Syntax

Visual Basic

```
Public Property All As Boolean
```

C#

```
public bool All {get;set;}
```

Remarks

Attempt to connect using the shared memory protocol first, followed by all remaining and available communication protocols. Use this setting if you are unsure of which communication protocol(s) to use.

1.6.6 ConnectionString Property

Gets or sets the connection string being built.

Syntax

Visual Basic

```
Public Property ConnectionString As String
```

C#

```
public string ConnectionString {get;set;}
```

1.6.7 SharedMemory Property

Gets or sets the SharedMemory protocol.

Syntax

Visual Basic

```
Public Property SharedMemory As Boolean
```

C#

```
public bool SharedMemory {get;set;}
```

1.6.8 TcpOptionsBuilder Property

Gets or sets an SATcpOptionsBuilder object used to create a TCP options string.

Syntax

Visual Basic

```
Public Property TcpOptionsBuilder As SATcpOptionsBuilder
```

C#

```
public SATcpOptionsBuilder TcpOptionsBuilder {get;set;}
```

1.6.9 TcpOptionsString Property

Gets or sets a string of TCP options.

Syntax

Visual Basic

```
Public Property TcpOptionsString As String
```

C#

```
public string TcpOptionsString {get;set;}
```

1.7 SAConnection Class

Represents a connection to a database.

Syntax

Visual Basic

```
Public NotInheritable Class SAConnection Inherits
System.Data.Common.DbConnection
```

C#

```
public sealed class SAConnection : System.Data.Common.DbConnection
```

Members

All members of SAConnection, including inherited members.

Constructors

Modifier and Type	Constructor	Description
public	SAConnection [page 122]	Initializes an SAConnection object.

Methods

Modifier and Type	Method	Description
protected override DbTransaction	BeginDbTransaction(IsolationLevel) [page 124]	Starts a database transaction.
public new SATransaction	BeginTransaction [page 125]	Returns a transaction object.

Modifier and Type	Method	Description
public override void	ChangeDatabase(string) [page 128]	Changes the current database for an open SAConnection.
public static void	ChangePassword(string, string) [page 129]	Changes the password to the supplied new password for the user indicated in the connection string.
public static void	ClearAllPools() [page 130]	Empties all connection pools.
public static void	ClearPool(SAConnection) [page 130]	Empties the connection pool associated with the specified connection.
public override void	Close() [page 131]	Closes a database connection.
public new SACommand	CreateCommand() [page 131]	Initializes an SACommand object.
protected override DbCommand	CreateDbCommand() [page 132]	Creates and returns a System.Data.Common.DbCommand object associated with the current connection.
protected override void	Dispose(bool) [page 132]	Frees the resources associated with the object.
public void	EnlistDistributedTransaction(System.EnterpriseServices.ITransaction) [page 133]	Enlists in the specified transaction as a distributed transaction.
public override void	EnlistTransaction(System.Transactions.Transaction) [page 133]	Enlists in the specified transaction as a distributed transaction.
public override DataTable	GetSchema [page 134]	Returns the list of supported schema collections.
public override unsafe void	Open() [page 141]	Opens a database connection with the property settings specified by the SAConnection.ConnectionString.

Properties

Modifier and Type	Property	Description
public override string	ConnectionString [page 141]	Provides the database connection string.
public override int	ConnectionTimeout [page 142]	Gets the number of seconds before a connection attempt times out with an error.
public SACredential	Credential [page 143]	Gets or sets the SACredential object for this connection.
public override string	Database [page 144]	Gets the name of the current database.
public override string	DataSource [page 144]	Gets the name of the database server.
public string	InitString [page 145]	A command that is executed immediately after the connection is established.

Modifier and Type	Property	Description
public override string	ServerVersion [page 145]	Gets a string that contains the version of the instance of the database server to which the client is connected.
public override ConnectionState	State [page 146]	Indicates the state of the SAConnection object.

Events

Modifier and Type	Event	Description
public SAInfoMessageEventHandler	InfoMessage [page 146]	Occurs when the database server returns a warning or informational message.
public override StateChangeEventHandler	StateChange [page 147]	Occurs when the state of the SAConnection object changes.

Remarks

For a list of connection parameters, see ".NET connection parameters" and "Connection parameters".

In this section:

SAConnection Constructor [page 122]
Initializes an SAConnection object.

BeginDbTransaction(IsolationLevel) Method [page 124]
Starts a database transaction.

BeginTransaction Method [page 125]
Returns a transaction object.

ChangeDatabase(string) Method [page 128]
Changes the current database for an open SAConnection.

ChangePassword(string, string) Method [page 129]
Changes the password to the supplied new password for the user indicated in the connection string.

ClearAllPools() Method [page 130]
Empties all connection pools.

ClearPool(SAConnection) Method [page 130]
Empties the connection pool associated with the specified connection.

Close() Method [page 131]
Closes a database connection.

CreateCommand() Method [page 131]
Initializes an SACommand object.

CreateDbCommand() Method [page 132]
Creates and returns a System.Data.Common.DbCommand object associated with the current connection.

1.7.1 SAConnection Constructor

Initializes an SAConnection object.

Overload list

Modifier and Type	Overload name	Description
public	SAConnection() [page 122]	Initializes an SAConnection object.
public	SAConnection(string) [page 123]	Initializes an SAConnection object.
public	SAConnection(string, SACredential) [page 124]	Initializes a new instance of the SAConnection class given a connection string, and a SACredential object that contains the user ID and password.

In this section:

SAConnection() Constructor [page 122]
Initializes an SAConnection object.

SAConnection(string) Constructor [page 123]
Initializes an SAConnection object.

SAConnection(string, SACredential) Constructor [page 124]
Initializes a new instance of the SAConnection class given a connection string, and a SACredential object that contains the user ID and password.

1.7.1.1 SAConnection() Constructor

Initializes an SAConnection object.

Syntax

Visual Basic

```
Public Sub SAConnection ()
```

C#

```
public SAConnection ()
```

Remarks

The connection must be opened before you perform any operations against the database.

1.7.1.2 SAConnection(string) Constructor

Initializes an SAConnection object.

Syntax

Visual Basic

```
Public Sub SAConnection (ByVal connectionString As String)
```

C#

```
public SAConnection (string connectionString)
```

Parameters

connectionString A connection string. A connection string is a semicolon-separated list of keyword=value pairs. For a list of connection parameters, see "Connection parameters".

Remarks

The connection must then be opened before you perform any operations against the database.

Example

The following statement initializes an SAConnection object for a connection to a database named policies running on a database server named hr. The connection uses the user ID admin and the password money.

```
SAConnection conn = new SAConnection(
   "UID=admin;PWD=money;SERVER=hr;DBN=policies" );
conn.Open();
```

Related Information

SAConnection Class [page 118]

1.7.1.3 SAConnection(string, SACredential) Constructor

Initializes a new instance of the SAConnection class given a connection string, and a SACredential object that contains the user ID and password.

Syntax

Visual Basic

```
Public Sub SAConnection (
    ByVal connectionString As String,
    ByVal credential As SACredential
)
```

C#

```
public SAConnection (
    string connectionString,
    SACredential credential
)
```

Parameters

connectionString A connection string. A connection string is a semicolon-separated list of keyword=value pairs. For a list of connection parameters, see "Connection parameters".

credential A SACredential object. If credential is null, then SAConnection(String, SACredential) is functionally equivalent to SAConnection(String).

Related Information

SAConnection Class [page 118]

1.7.2 BeginDbTransaction(IsolationLevel) Method

Starts a database transaction.

Syntax

Visual Basic

```
Protected Overrides Function BeginDbTransaction (ByVal isolationLevel As
IsolationLevel) As DbTransaction
```

C#

```
protected override DbTransaction BeginDbTransaction (IsolationLevel
isolationLevel)
```

Parameters

isolationLevel Specifies the isolation level for the transaction.

Returns

An object representing the new transaction.

1.7.3 BeginTransaction Method

Returns a transaction object.

Overload list

Modifier and Type	Overload name	Description
public new SATransaction	BeginTransaction() [page 125]	Returns a transaction object.
public new SATransaction	BeginTransaction(IsolationLevel) [page 126]	Returns a transaction object.
public SATransaction	BeginTransaction(SAIsolationLevel) [page 127]	Returns a transaction object.

In this section:

BeginTransaction() Method [page 125]
Returns a transaction object.

BeginTransaction(IsolationLevel) Method [page 126]
Returns a transaction object.

BeginTransaction(SAIsolationLevel) Method [page 127]
Returns a transaction object.

1.7.3.1 BeginTransaction() Method

Returns a transaction object.

Syntax

Visual Basic

```
Public Shadows Function BeginTransaction () As SATransaction
```

C#

```
public new SATransaction BeginTransaction ()
```

Returns

An SATransaction object representing the new transaction.

Remarks

Commands associated with a transaction object are executed as a single transaction. The transaction is terminated with a call to the Commit or Rollback methods.

To associate a command with a transaction object, use the SACommand.Transaction property.

Related Information

SATransaction Class [page 335]
Transaction Property [page 90]

1.7.3.2 BeginTransaction(IsolationLevel) Method

Returns a transaction object.

Syntax

Visual Basic

```
Public Shadows Function BeginTransaction (ByVal isolationLevel As
IsolationLevel) As SATransaction
```

C#

```
public new SATransaction BeginTransaction (IsolationLevel isolationLevel)
```

Parameters

isolationLevel A member of the SAIsolationLevel enumeration. The default value is ReadCommitted.

Returns

An SATransaction object representing the new transaction.

Remarks

Commands associated with a transaction object are executed as a single transaction. The transaction is terminated with a call to the Commit or Rollback methods.

To associate a command with a transaction object, use the SACommand.Transaction property.

Example

```
SATransaction tx =
 conn.BeginTransaction( SAIsolationLevel.ReadUncommitted );
```

Related Information

SATransaction Class [page 335]
Transaction Property [page 90]
SAIsolationLevel Enumeration [page 350]

1.7.3.3 BeginTransaction(SAIsolationLevel) Method

Returns a transaction object.

Syntax

Visual Basic

```
Public Function BeginTransaction (ByVal isolationLevel As
SAIsolationLevel) As SATransaction
```

C#

```
public SATransaction BeginTransaction (SAIsolationLevel isolationLevel)
```

Parameters

isolationLevel A member of the SAIsolationLevel enumeration. The default value is ReadCommitted.

Returns

An SATransaction object representing the new transaction.

Remarks

Commands associated with a transaction object are executed as a single transaction. The transaction is terminated with a call to the Commit or Rollback methods.

To associate a command with a transaction object, use the SACommand.Transaction property.

For more information, see "Transaction processing".

For more information, see "Typical types of inconsistency".

Related Information

SATransaction Class [page 335]
Transaction Property [page 90]
SAIsolationLevel Enumeration [page 350]
Commit() Method [page 337]
Rollback() Method [page 338]
Rollback(string) Method [page 339]

1.7.4 ChangeDatabase(string) Method

Changes the current database for an open SAConnection.

Syntax

Visual Basic

```
Public Overrides Sub ChangeDatabase (ByVal database As String)
```

C#

```
public override void ChangeDatabase (string database)
```

Parameters

database The name of the database to use instead of the current database.

1.7.5 ChangePassword(string, string) Method

Changes the password to the supplied new password for the user indicated in the connection string.

Syntax

Visual Basic

```
Public Shared Sub ChangePassword (
    ByVal connectionString As String,
    ByVal newPassword As String
)
```

C#

```
public static void ChangePassword (
    string connectionString,
    string newPassword
)
```

Parameters

connectionString The connection string that contains enough information to connect to the database server that you want. The connection string may contain the user ID and the current password.

newPassword The new password to set. This password must comply with any password security policy set on the database server, including minimum length, requirements for specific characters, and so on.

Exceptions

ArgumentNullException Either the connectionString or the newPassword parameter is null.

ArgumentException The connection string includes the option to use integrated security.

1.7.6 ClearAllPools() Method

Empties all connection pools.

Syntax

Visual Basic

```
Public Shared Sub ClearAllPools ()
```

C#

```
public static void ClearAllPools ()
```

1.7.7 ClearPool(SAConnection) Method

Empties the connection pool associated with the specified connection.

Syntax

Visual Basic

```
Public Shared Sub ClearPool (ByVal connection As SAConnection)
```

C#

```
public static void ClearPool (SAConnection connection)
```

Parameters

connection The SAConnection object to be cleared from the pool.

Related Information

SAConnection Class [page 118]

1.7.8 Close() Method

Closes a database connection.

Syntax

Visual Basic

```
Public Overrides Sub Close ()
```

C#

```
public override void Close ()
```

Remarks

The Close method rolls back any pending transactions. It then releases the connection to the connection pool, or closes the connection if connection pooling is disabled. If Close is called while handling a StateChange event, then no additional StateChange events are fired. An application can call Close multiple times.

1.7.9 CreateCommand() Method

Initializes an SACommand object.

Syntax

Visual Basic

```
Public Shadows Function CreateCommand () As SACommand
```

C#

```
public new SACommand CreateCommand ()
```

Returns

An SACommand object.

Remarks

The command object is associated with the SAConnection object.

Related Information

SACommand Class [page 53]
SAConnection Class [page 118]

1.7.10 CreateDbCommand() Method

Creates and returns a System.Data.Common.DbCommand object associated with the current connection.

Syntax

Visual Basic

```
Protected Overrides Function CreateDbCommand () As DbCommand
```

C#

```
protected override DbCommand CreateDbCommand ()
```

Returns

A System.Data.Common.DbCommand object.

1.7.11 Dispose(bool) Method

Frees the resources associated with the object.

Syntax

Visual Basic

```
Protected Overrides Sub Dispose (ByVal disposing As Boolean)
```

C#

```
protected override void Dispose (bool disposing)
```

1.7.12 EnlistDistributedTransaction(System.EnterpriseServices.ITransaction) Method

Enlists in the specified transaction as a distributed transaction.

Syntax

Visual Basic

```
Public Sub EnlistDistributedTransaction (ByVal transaction As
System.EnterpriseServices.ITransaction)
```

C#

```
public void EnlistDistributedTransaction
(System.EnterpriseServices.ITransaction transaction)
```

Parameters

transaction A reference to an existing System.EnterpriseServices.ITransaction in which to enlist.

1.7.13 EnlistTransaction(System.Transactions.Transaction) Method

Enlists in the specified transaction as a distributed transaction.

Syntax

Visual Basic

```
Public Overrides Sub EnlistTransaction (ByVal transaction As
System.Transactions.Transaction)
```

C#

```
public override void EnlistTransaction (System.Transactions.Transaction
transaction)
```

Parameters

transaction A reference to an existing System.Transactions.Transaction in which to enlist.

1.7.14 GetSchema Method

Returns the list of supported schema collections.

Overload list

Modifier and Type	Overload name	Description
public override DataTable	GetSchema() [page 134]	Returns the list of supported schema collections.
public override DataTable	GetSchema(string) [page 135]	Returns information for the specified metadata collection for this SAConnection object.
public override DataTable	GetSchema(string, string[]) [page 136]	Returns schema information for the data source of this SAConnection object and, if specified, uses the specified string for the schema name and the specified string array for the restriction values.

In this section:

GetSchema() Method [page 134]
Returns the list of supported schema collections.

GetSchema(string) Method [page 135]
Returns information for the specified metadata collection for this SAConnection object.

GetSchema(string, string[]) Method [page 136]
Returns schema information for the data source of this SAConnection object and, if specified, uses the specified string for the schema name and the specified string array for the restriction values.

1.7.14.1 GetSchema() Method

Returns the list of supported schema collections.

Syntax

Visual Basic

```
Public Overrides Function GetSchema () As DataTable
```

C#

```
public override DataTable GetSchema ()
```

Remarks

See GetSchema(string,string[]) for a description of the available metadata.

1.7.14.2 GetSchema(string) Method

Returns information for the specified metadata collection for this SAConnection object.

Syntax

Visual Basic

```
Public Overrides Function GetSchema (ByVal collection As String) As
DataTable
```

C#

```
public override DataTable GetSchema (string collection)
```

Parameters

collection Name of the metadata collection. If a name is not provided, then MetaDataCollections is used.

Remarks

See GetSchema(string,string[]) for a description of the available metadata.

Related Information

SAConnection Class [page 118]

1.7.14.3 GetSchema(string, string[]) Method

Returns schema information for the data source of this SAConnection object and, if specified, uses the specified string for the schema name and the specified string array for the restriction values.

Syntax

Visual Basic

```
Public Overrides Function GetSchema (
    ByVal collection As String,
    ByVal restrictions As String()
) As DataTable
```

C#

```
public override DataTable GetSchema (
    string collection,
    string[] restrictions
)
```

Returns

A DataTable that contains schema information.

Remarks

These methods are used to query the database server for various metadata. Each type of metadata is given a collection name, which must be passed to receive that data. The default collection name is MetaDataCollections.

Query the .NET Data Provider to determine the list of supported schema collections by calling the GetSchema method with no arguments, or with the schema collection name *MetaDataCollections*. This will return a DataTable with a list of the supported schema collections (CollectionName), the number of restrictions that they each support (NumberOfRestrictions), and the number of identifier parts that they use (NumberOfIdentifierParts).

Collection	Metadata
Columns	Returns information on all columns in the database.
DataSourceInformation	Returns information about the database server.
DataTypes	Returns a list of supported data types.
ForeignKeys	Returns information on all foreign keys in the database.
IndexColumns	Returns information on all index columns in the database.
Indexes	Returns information on all indexes in the database.

Collection	Metadata
MetaDataCollections	Returns a list of all collection names.
ProcedureParameters	Returns information on all procedure parameters in the database.
Procedures	Returns information on all procedures in the database.
ReservedWords	Returns a list of reserved words used by the database server.
Restrictions	Returns information on restrictions used in GetSchema.
Tables	Returns information on all tables in the database.
UserDefinedTypes	Returns information on all user-defined data types in the database.
Users	Returns information on all users in the database.
ViewColumns	Returns information on all columns in views in the database.
Views	Returns information on all views in the database.

These collection names are also available as read-only properties in the SAMetaDataCollectionNames class.

The results returned can be filtered by specifying an array of restrictions in the call to GetSchema.

The restrictions available with each collection can be queried by calling:

```
GetSchema( "Restrictions" )
```

If the collection requires four restrictions, then the restrictions parameter must be either NULL, or a string with four values.

To filter on a particular restriction, place the string to filter by in its place in the array and leave any unused places NULL. For example, the Tables collection has three restrictions: Owner, Table, and TableType.

To filter the Table collection by table_name:

```
GetSchema( "Tables", new string[ ] { NULL, "my_table", NULL } )
```

This example returns information on all tables named my_table.

```
GetSchema( "Tables", new string[ ] { "DBA", "my_table", NULL } )
```

This example returns information on all tables named my_table owned by the user DBA.

The following list summarizes the columns returned by each collection. If the number of rows returned in a collection can be reduced by specifying a restriction on a column, then the restriction name for that column is shown in parentheses. The order in which restrictions are specified is the order in which they are presented in the lists below.

Columns collection

- TABLE_SCHEMA (Owner)
- TABLE_NAME (Table)
- COLUMN_NAME (Column)
- ORDINAL_POSITION
- DATA_TYPE

- COLUMN_DEFAULT
- IS_NULLABLE
- NUMERIC_PRECISION
- NUMERIC_SCALE
- CHARACTER_MAXIMUM_LENGTH
- DATETIME_PRECISION

DataSourceInformation collection

- CompositeIdentifierSeparatorPattern
- DataSourceProductName
- DataSourceProductVersion
- DataSourceProductVersionNormalized
- GroupByBehavior
- IdentifierPattern
- IdentifierCase
- OrderByColumnsInSelect
- ParameterMarkerFormat
- ParameterMarkerPattern
- ParameterNameMaxLength
- ParameterNamePattern
- QuotedIdentifierPattern
- QuotedIdentifierCase
- StatementSeparatorPattern
- StringLiteralPattern
- SupportedJoinOperators

DataTypes collection

- TypeName
- ProviderDbType
- ColumnSize
- CreateFormat
- CreateParameters
- DataType
- IsAutoIncrementable
- IsBestMatch
- IsCaseSensitive
- IsFixedLength
- IsFixedPrecisionScale
- IsLong
- IsNullable
- IsSearchable
- IsSearchableWithLike
- IsUnsigned

- MaximumScale
- MinimumScale
- IsConcurrencyType
- IsLiteralSupported
- LiteralPrefix
- LiteralSuffix

ForeignKeys collection

- TABLE_SCHEMA (Owner)
- TABLE_NAME (Table)
- COLUMN_NAME (Column)

IndexColumns collection

- TABLE_SCHEMA (Owner)
- TABLE_NAME (Table)
- INDEX_NAME (Name)
- COLUMN_NAME (Column)
- ORDER

Indexes collection

- TABLE_SCHEMA (Owner)
- TABLE_NAME (Table)
- INDEX_NAME (Name)
- PRIMARY_KEY
- IS_UNIQUE

MetaDataCollections collection

- CollectionName
- NumberOfRestrictions
- NumberOfIdentifierParts

ProcedureParameters collection

- PROCEDURE_SCHEMA (Owner)
- PROCEDURE_NAME (Name)
- PARAMETER_NAME (Parameter)
- DATA_TYPE
- PARAMETER_TYPE
- IS_INPUT
- IS_OUTPUT

Procedures collection

- PROCEDURE_SCHEMA (Owner)
- PROCEDURE_NAME (Name)

ReservedWords collection

- reserved_word

Restrictions collection

- CollectionName
- RestrictionName
- RestrictionDefault
- RestrictionNumber

Tables collection

- TABLE_SCHEMA (Owner)
- TABLE_NAME (Table)
- TABLE_TYPE (TableType) either "BASE", "VIEW", "MAT VIEW", "LCL TEMP", "GBL TEMP", "TEXT", or "TEXT GBL TEMP"

UserDefinedTypes collection

- DATA_TYPE
- DEFAULT
- PRECISION
- SCALE

Users collection

- USER_NAME (UserName)
- RESOURCE_AUTH
- DATABASE_AUTH
- SCHEDULE_AUTH
- USER_GROUP

ViewColumns collection

- VIEW_SCHEMA (Owner)
- VIEW_NAME (Name)
- COLUMN_NAME (Column)

Views collection

- VIEW_SCHEMA (Owner)
- VIEW_NAME (Name)

Related Information

SAConnection Class [page 118]

1.7.15 Open() Method

Opens a database connection with the property settings specified by the SAConnection.ConnectionString.

Syntax

Visual Basic

```
Public Overrides Sub Open ()
```

C#

```
public override unsafe void Open ()
```

Related Information

ConnectionString Property [page 141]

1.7.16 ConnectionString Property

Provides the database connection string.

Syntax

Visual Basic

```
Public Overrides Property ConnectionString As String
```

C#

```
public override string ConnectionString {get;set;}
```

Remarks

The ConnectionString is designed to match the database server connection string format as closely as possible with the following exception: when the Persist Security Info value is set to false (the default), the connection string that is returned is the same as the user-set ConnectionString minus security information. The .NET Data Provider does not persist the password in a returned connection string unless you set Persist Security Info to true.

Use the ConnectionString property to connect to a variety of data sources.

You can set the ConnectionString property only when the connection is closed. Many of the connection string values have corresponding read-only properties. When the connection string is set, all of these properties are

updated, unless an error is detected. If an error is detected, then none of the properties are updated. SAConnection properties return only those settings contained in the ConnectionString.

If you reset the ConnectionString on a closed connection, then all connection string values and related properties are reset, including the password.

When the property is set, a preliminary validation of the connection string is performed. When an application calls the Open method, the connection string is fully validated. A runtime exception is generated if the connection string contains invalid or unsupported properties.

Values can be delimited by single or double quotes. Either single or double quotes may be used within a connection string by using the other delimiter, for example, name="value's" or name= 'value"s', but not name='value's' or name= ""value"". Blank characters are ignored unless they are placed within a value or within quotes. keyword=value pairs must be separated by a semicolon. If a semicolon is part of a value, then it must also be delimited by quotes. Escape sequences are not supported, and the value type is irrelevant. Names are not case sensitive. If a property name occurs more than once in the connection string, then the value associated with the last occurrence is used.

Use caution when constructing a connection string based on user input, such as when retrieving a user ID and password from a window, and appending it to the connection string. The application should not allow a user to embed extra connection string parameters in these values.

The default value of connection pooling is true (pooling=true).

Example

The following statements set a connection string for an ODBC data source and open the connection.

```
SAConnection conn = new SAConnection();
conn.ConnectionString = "DSN=SQL Anywhere 17 Demo";
conn.Open();
```

Related Information

SAConnection Class [page 118]
Open() Method [page 141]

1.7.17 ConnectionTimeout Property

Gets the number of seconds before a connection attempt times out with an error.

Syntax

Visual Basic

```
Public ReadOnly Overrides Property ConnectionTimeout As Integer
```

C#

```
public override int ConnectionTimeout {get;}
```

Remarks

The default ConnectionTimeout value is 15 seconds.

Example

The following statement displays the value of the ConnectionTimeout.

```
MessageBox.Show( conn.ConnectionTimeout.ToString( ) );
```

1.7.18 Credential Property

Gets or sets the SACredential object for this connection.

Syntax

Visual Basic

```
Public Property Credential As SACredential
```

C#

```
public SACredential Credential {get;set;}
```

Returns

The SACredential object for this connection.

Related Information

SAConnection Class [page 118]
Open() Method [page 141]

1.7.19 Database Property

Gets the name of the current database.

Syntax

Visual Basic

```
Public ReadOnly Overrides Property Database As String
```

C#

```
public override string Database {get;}
```

Remarks

If the connection is open, then the SAConnection object returns the name of the current database. Otherwise, SAConnection looks in the connection string in the following order: DatabaseName, DBN, DataSourceName, Data Source, DSN, DatabaseFile, DBF, FileDataSourceName, FileDSN.

1.7.20 DataSource Property

Gets the name of the database server.

Syntax

Visual Basic

```
Public ReadOnly Overrides Property DataSource As String
```

C#

```
public override string DataSource {get;}
```

Remarks

If the connection is open, then the SAConnection object returns the ServerName server property. Otherwise, the SAConnection object looks in the connection string for the ServerName connection parameter.

Related Information

SAConnection Class [page 118]

1.7.21 InitString Property

A command that is executed immediately after the connection is established.

Syntax

Visual Basic

```
Public Property InitString As String
```

C#

```
public string InitString {get;set;}
```

Remarks

The InitString is executed immediately after the connection is opened.

1.7.22 ServerVersion Property

Gets a string that contains the version of the instance of the database server to which the client is connected.

Syntax

Visual Basic

```
Public ReadOnly Overrides Property ServerVersion As String
```

C#

```
public override string ServerVersion {get;}
```

Returns

The version of the instance of the database server.

Remarks

The version is ##.##.####, where the first two digits are the major version, the next two digits are the minor version, and the last four digits are the release version. The appended string is of the form major.minor.build, where major and minor are two digits, and build is four digits.

1.7.23 State Property

Indicates the state of the SAConnection object.

Syntax

Visual Basic

```
Public ReadOnly Overrides Property State As ConnectionState
```

C#

```
public override ConnectionState State {get;}
```

Returns

A System.Data.ConnectionState enumeration.

1.7.24 InfoMessage Event

Occurs when the database server returns a warning or informational message.

Syntax

Visual Basic

```
Public Event InfoMessage  As SAInfoMessageEventHandler
```

C#

```
public SAInfoMessageEventHandler InfoMessage;
```

Remarks

The event handler receives an argument of type SAInfoMessageEventArgs containing data related to this event. The following SAInfoMessageEventArgs properties provide information specific to this event: NativeError, Errors, Message, MessageType, and Source.

For more information, see the .NET Framework documentation for OleDbConnection.InfoMessage Event.

1.7.25 StateChange Event

Occurs when the state of the SAConnection object changes.

Syntax

Visual Basic

```
Public Event StateChange  As StateChangeEventHandler
```

C#

```
public override StateChangeEventHandler StateChange;
```

Remarks

The event handler receives an argument of type StateChangeEventArgs with data related to this event. The following StateChangeEventArgs properties provide information specific to this event: CurrentState and OriginalState.

For more information, see the .NET Framework documentation for OleDbConnection.StateChange Event.

1.8 SAConnectionStringBuilder Class

Provides a simple way to create and manage the contents of connection strings used by the SAConnection class.

Syntax

Visual Basic

```
Public NotInheritable Class SAConnectionStringBuilder Inherits
SAConnectionStringBuilderBase
```

C#

```
public sealed class SAConnectionStringBuilder :
SAConnectionStringBuilderBase
```

Members

All members of SAConnectionStringBuilder, including inherited members.

Constructors

Modifier and Type	Constructor	Description
public	SAConnectionStringBuilder [page 153]	Initializes a new instance of the SAConnectionStringBuilder class.

Properties

Modifier and Type	Property	Description
public string	AppInfo [page 155]	Gets or sets the AppInfo connection property.
public string	AutoStart [page 155]	Gets or sets the AutoStart connection property.
public string	AutoStop [page 155]	Gets or sets the AutoStop connection property.
public string	Charset [page 156]	Gets or sets the Charset connection property.
public int	CommBufferSize [page 156]	Gets or sets the CommBufferSize connection property.
public string	CommLinks [page 156]	Gets or sets the CommLinks connection property.
public string	Compress [page 157]	Gets or sets the Compress connection property.
public int	CompressionThreshold [page 157]	Gets or sets the CompressionThreshold connection property.
public int	ConnectionLifetime [page 157]	Gets or sets the ConnectionLifetime connection property.
public string	ConnectionName [page 158]	Gets or sets the ConnectionName connection property.
public string	ConnectionPool [page 158]	Gets or sets the ConnectionPool connection property.
public bool	ConnectionReset [page 158]	Gets or sets the ConnectionReset connection property.
public int	ConnectionTimeout [page 159]	Gets or sets the ConnectionTimeout connection property.
public string	DatabaseFile [page 159]	Gets or sets the DatabaseFile connection property.
public string	DatabaseKey [page 159]	Gets or sets the DatabaseKey connection property.
public string	DatabaseName [page 160]	Gets or sets the DatabaseName connection property.
public string	DatabaseSwitches [page 160]	Gets or sets the DatabaseSwitches connection property.
public string	DataSourceName [page 160]	Gets or sets the DataSourceName connection property.

Modifier and Type	Property	Description
public string	DisableMultiRowFetch [page 161]	Gets or sets the DisableMultiRowFetch connection property.
public string	Elevate [page 161]	Gets or sets the Elevate connection property.
public string	EncryptedPassword [page 161]	Gets or sets the EncodedPassword connection property.
public string	Encryption [page 162]	Gets or sets the Encryption connection property.
public bool	Enlist [page 162]	Gets or sets the Enlist connection property.
public string	FileDataSourceName [page 162]	Gets or sets the FileDataSourceName connection property.
public string	ForceStart [page 163]	Gets or sets the ForceStart connection property.
public string	Host [page 163]	Gets or sets the Host connection property.
public int	IdleTimeout [page 163]	Gets or sets the IdleTimeout connection property.
public string	InitString [page 164]	Gets or sets the InitString connection property.
public string	Language [page 165]	Gets or sets the Language connection property.
public string	LazyClose [page 165]	Gets or sets the LazyClose connection property.
public int	LivenessTimeout [page 165]	Gets or sets the LivenessTimeout connection property.
public string	LogFile [page 166]	Gets or sets the LogFile connection property.
public int	MaxPoolSize [page 166]	Gets or sets the MaxPoolSize connection property.
public int	MinPoolSize [page 166]	Gets or sets the MinPoolSize connection property.
public string	NewPassword [page 167]	Gets or sets the NewPassword connection property.
public string	NodeType [page 167]	Gets or sets the NodeType connection property.
public string	Password [page 167]	Gets or sets the Password connection property.
public bool	PersistSecurityInfo [page 168]	Gets or sets the PersistSecurityInfo connection property.
public bool	Pooling [page 168]	Gets or sets the Pooling connection property.

Modifier and Type	Property	Description
public int	PrefetchBuffer [page 168]	Gets or sets the PrefetchBuffer connection property.
public int	PrefetchRows [page 169]	Gets or sets the PrefetchRows connection property.
public int	RetryConnectionTimeout [page 169]	Gets or sets the RetryConnectionTimeout connection property.
public string	ServerName [page 169]	Gets or sets the ServerName connection property.
public string	StartLine [page 170]	Gets or sets the StartLine connection property.
public string	Unconditional [page 170]	Gets or sets the Unconditional connection property.
public string	UserID [page 170]	Gets or sets the UserID connection property.

Inherited members from SAConnectionStringBuilderBase

Modifier and Type	Member	Description
public override bool	ContainsKey(string) [page 172]	Determines whether the SAConnectionStringBuilder object contains a specific keyword.
public string	GetKeyword(string) [page 173]	Gets the keyword for the specified SAConnectionStringBuilder property.
public bool	GetUseLongNameAsKeyword() [page 174]	Gets a boolean value that indicates whether long connection parameter names are used in the connection string.
public override ICollection	Keys [page 177]	Gets an System.Collections.ICollection that contains the keys in the SAConnectionStringBuilder.
public override bool	Remove(string) [page 174]	Removes the entry with the specified key from the SAConnectionStringBuilder instance.
public void	SetUseLongNameAsKeyword(bool) [page 175]	Sets a boolean value that indicates whether long connection parameter names are used in the connection string.
public override bool	ShouldSerialize(string) [page 176]	Indicates whether the specified key exists in this SAConnectionStringBuilder instance.
public override object	this[string keyword] [page 177]	Gets or sets the value of the connection keyword.
public override bool	TryGetValue(string, out object) [page 176]	Retrieves a value corresponding to the supplied key from this SAConnectionStringBuilder.

Remarks

The SAConnectionStringBuilder class inherits SAConnectionStringBuilderBase, which inherits DbConnectionStringBuilder.

For a list of connection parameters, see ".NET connection parameters" and "Connection parameters".

In this section:

DatabaseSwitches Property [page 160]
Gets or sets the DatabaseSwitches connection property.

DataSourceName Property [page 160]
Gets or sets the DataSourceName connection property.

DisableMultiRowFetch Property [page 161]
Gets or sets the DisableMultiRowFetch connection property.

Elevate Property [page 161]
Gets or sets the Elevate connection property.

EncryptedPassword Property [page 161]
Gets or sets the EncodedPassword connection property.

Encryption Property [page 162]
Gets or sets the Encryption connection property.

Enlist Property [page 162]
Gets or sets the Enlist connection property.

FileDataSourceName Property [page 162]
Gets or sets the FileDataSourceName connection property.

ForceStart Property [page 163]
Gets or sets the ForceStart connection property.

Host Property [page 163]
Gets or sets the Host connection property.

IdleTimeout Property [page 163]
Gets or sets the IdleTimeout connection property.

InitString Property [page 164]
Gets or sets the InitString connection property.

Integrated Property [page 164]
Gets or sets the Integrated connection property.

Kerberos Property [page 164]
Gets or sets the Kerberos connection property.

Language Property [page 165]
Gets or sets the Language connection property.

LazyClose Property [page 165]
Gets or sets the LazyClose connection property.

LivenessTimeout Property [page 165]
Gets or sets the LivenessTimeout connection property.

LogFile Property [page 166]
Gets or sets the LogFile connection property.

MaxPoolSize Property [page 166]
Gets or sets the MaxPoolSize connection property.

MinPoolSize Property [page 166]
Gets or sets the MinPoolSize connection property.

NewPassword Property [page 167]
Gets or sets the NewPassword connection property.

NodeType Property [page 167]
Gets or sets the NodeType connection property.

Password Property [page 167]
Gets or sets the Password connection property.

PersistSecurityInfo Property [page 168]
Gets or sets the PersistSecurityInfo connection property.

Pooling Property [page 168]
Gets or sets the Pooling connection property.

PrefetchBuffer Property [page 168]
Gets or sets the PrefetchBuffer connection property.

PrefetchRows Property [page 169]
Gets or sets the PrefetchRows connection property.

RetryConnectionTimeout Property [page 169]
Gets or sets the RetryConnectionTimeout connection property.

ServerName Property [page 169]
Gets or sets the ServerName connection property.

StartLine Property [page 170]
Gets or sets the StartLine connection property.

Unconditional Property [page 170]
Gets or sets the Unconditional connection property.

UserID Property [page 170]
Gets or sets the UserID connection property.

1.8.1 SAConnectionStringBuilder Constructor

Initializes a new instance of the SAConnectionStringBuilder class.

Overload list

Modifier and Type	Overload name	Description
public	SAConnectionStringBuilder() [page 154]	Initializes a new instance of the SAConnectionStringBuilder class.
public	SAConnectionStringBuilder(string) [page 154]	Initializes a new instance of the SAConnectionStringBuilder class.

In this section:

SAConnectionStringBuilder() Constructor [page 154]
Initializes a new instance of the SAConnectionStringBuilder class.

SAConnectionStringBuilder(string) Constructor [page 154]

Initializes a new instance of the SAConnectionStringBuilder class.

1.8.1.1 SAConnectionStringBuilder() Constructor

Initializes a new instance of the SAConnectionStringBuilder class.

Syntax

Visual Basic

```
Public Sub SAConnectionStringBuilder ()
```

C#

```
public SAConnectionStringBuilder ()
```

1.8.1.2 SAConnectionStringBuilder(string) Constructor

Initializes a new instance of the SAConnectionStringBuilder class.

Syntax

Visual Basic

```
Public Sub SAConnectionStringBuilder (ByVal connectionString As String)
```

C#

```
public SAConnectionStringBuilder (string connectionString)
```

Parameters

connectionString The basis for the object's internal connection information. Parsed into keyword=value pairs. For a list of connection parameters, see "Connection parameters".

Example

The following statement initializes an SAConnection object for a connection to a database named policies running on a database server named hr. The connection uses the user ID admin and the password money.

```
SAConnectionStringBuilder conn = new SAConnectionStringBuilder(
 "UID=admin;PWD=money;SERVER=hr;DBN=policies" );
```

1.8.2 AppInfo Property

Gets or sets the AppInfo connection property.

Syntax

Visual Basic

```
Public Property AppInfo As String
```

C#

```
public string AppInfo {get;set;}
```

1.8.3 AutoStart Property

Gets or sets the AutoStart connection property.

Syntax

Visual Basic

```
Public Property AutoStart As String
```

C#

```
public string AutoStart {get;set;}
```

1.8.4 AutoStop Property

Gets or sets the AutoStop connection property.

Syntax

Visual Basic

```
Public Property AutoStop As String
```

C#

```
public string AutoStop {get;set;}
```

1.8.5 Charset Property

Gets or sets the Charset connection property.

Syntax

Visual Basic

```
Public Property Charset As String
```

C#

```
public string Charset {get;set;}
```

1.8.6 CommBufferSize Property

Gets or sets the CommBufferSize connection property.

Syntax

Visual Basic

```
Public Property CommBufferSize As Integer
```

C#

```
public int CommBufferSize {get;set;}
```

1.8.7 CommLinks Property

Gets or sets the CommLinks connection property.

Syntax

Visual Basic

```
Public Property CommLinks As String
```

C#

```
public string CommLinks {get;set;}
```

1.8.8 Compress Property

Gets or sets the Compress connection property.

Syntax

Visual Basic

```
Public Property Compress As String
```

C#

```
public string Compress {get;set;}
```

1.8.9 CompressionThreshold Property

Gets or sets the CompressionThreshold connection property.

Syntax

Visual Basic

```
Public Property CompressionThreshold As Integer
```

C#

```
public int CompressionThreshold {get;set;}
```

1.8.10 ConnectionLifetime Property

Gets or sets the ConnectionLifetime connection property.

Syntax

Visual Basic

```
Public Property ConnectionLifetime As Integer
```

C#

```
public int ConnectionLifetime {get;set;}
```

1.8.11 ConnectionName Property

Gets or sets the ConnectionName connection property.

Syntax

Visual Basic

```
Public Property ConnectionName As String
```

C#

```
public string ConnectionName {get;set;}
```

1.8.12 ConnectionPool Property

Gets or sets the ConnectionPool connection property.

Syntax

Visual Basic

```
Public Property ConnectionPool As String
```

C#

```
public string ConnectionPool {get;set;}
```

1.8.13 ConnectionReset Property

Gets or sets the ConnectionReset connection property.

Syntax

Visual Basic

```
Public Property ConnectionReset As Boolean
```

C#

```
public bool ConnectionReset {get;set;}
```

1.8.14 ConnectionTimeout Property

Gets or sets the ConnectionTimeout connection property.

Syntax

Visual Basic

```
Public Property ConnectionTimeout As Integer
```

C#

```
public int ConnectionTimeout {get;set;}
```

Example

The following statement displays the value of the ConnectionTimeout property.

```
MessageBox.Show( connString.ConnectionTimeout.ToString() );
```

1.8.15 DatabaseFile Property

Gets or sets the DatabaseFile connection property.

Syntax

Visual Basic

```
Public Property DatabaseFile As String
```

C#

```
public string DatabaseFile {get;set;}
```

1.8.16 DatabaseKey Property

Gets or sets the DatabaseKey connection property.

Syntax

Visual Basic

```
Public Property DatabaseKey As String
```

C#

```
public string DatabaseKey {get;set;}
```

1.8.17 DatabaseName Property

Gets or sets the DatabaseName connection property.

Syntax

Visual Basic

```
Public Property DatabaseName As String
```

C#

```
public string DatabaseName {get;set;}
```

1.8.18 DatabaseSwitches Property

Gets or sets the DatabaseSwitches connection property.

Syntax

Visual Basic

```
Public Property DatabaseSwitches As String
```

C#

```
public string DatabaseSwitches {get;set;}
```

1.8.19 DataSourceName Property

Gets or sets the DataSourceName connection property.

Syntax

Visual Basic

```
Public Property DataSourceName As String
```

C#

```
public string DataSourceName {get;set;}
```

1.8.20 DisableMultiRowFetch Property

Gets or sets the DisableMultiRowFetch connection property.

Syntax

Visual Basic

```
Public Property DisableMultiRowFetch As String
```

C#

```
public string DisableMultiRowFetch {get;set;}
```

1.8.21 Elevate Property

Gets or sets the Elevate connection property.

Syntax

Visual Basic

```
Public Property Elevate As String
```

C#

```
public string Elevate {get;set;}
```

1.8.22 EncryptedPassword Property

Gets or sets the EncodedPassword connection property.

Syntax

Visual Basic

```
Public Property EncryptedPassword As String
```

C#

```
public string EncryptedPassword {get;set;}
```

1.8.23 Encryption Property

Gets or sets the Encryption connection property.

Syntax

Visual Basic

```
Public Property Encryption As String
```

C#

```
public string Encryption {get;set;}
```

1.8.24 Enlist Property

Gets or sets the Enlist connection property.

Syntax

Visual Basic

```
Public Property Enlist As Boolean
```

C#

```
public bool Enlist {get;set;}
```

1.8.25 FileDataSourceName Property

Gets or sets the FileDataSourceName connection property.

Syntax

Visual Basic

```
Public Property FileDataSourceName As String
```

C#

```
public string FileDataSourceName {get;set;}
```

1.8.26 ForceStart Property

Gets or sets the ForceStart connection property.

Syntax

Visual Basic

```
Public Property ForceStart As String
```

C#

```
public string ForceStart {get;set;}
```

1.8.27 Host Property

Gets or sets the Host connection property.

Syntax

Visual Basic

```
Public Property Host As String
```

C#

```
public string Host {get;set;}
```

1.8.28 IdleTimeout Property

Gets or sets the IdleTimeout connection property.

Syntax

Visual Basic

```
Public Property IdleTimeout As Integer
```

C#

```
public int IdleTimeout {get;set;}
```

1.8.29 InitString Property

Gets or sets the InitString connection property.

Syntax

Visual Basic

```
Public Property InitString As String
```

C#

```
public string InitString {get;set;}
```

1.8.30 Integrated Property

Gets or sets the Integrated connection property.

Syntax

Visual Basic

```
Public Property Integrated As String
```

C#

```
public string Integrated {get;set;}
```

1.8.31 Kerberos Property

Gets or sets the Kerberos connection property.

Syntax

Visual Basic

```
Public Property Kerberos As String
```

C#

```
public string Kerberos {get;set;}
```

1.8.32 Language Property

Gets or sets the Language connection property.

Syntax

Visual Basic

```
Public Property Language As String
```

C#

```
public string Language {get;set;}
```

1.8.33 LazyClose Property

Gets or sets the LazyClose connection property.

Syntax

Visual Basic

```
Public Property LazyClose As String
```

C#

```
public string LazyClose {get;set;}
```

1.8.34 LivenessTimeout Property

Gets or sets the LivenessTimeout connection property.

Syntax

Visual Basic

```
Public Property LivenessTimeout As Integer
```

C#

```
public int LivenessTimeout {get;set;}
```

1.8.35 LogFile Property

Gets or sets the LogFile connection property.

Syntax

Visual Basic

```
Public Property LogFile As String
```

C#

```
public string LogFile {get;set;}
```

1.8.36 MaxPoolSize Property

Gets or sets the MaxPoolSize connection property.

Syntax

Visual Basic

```
Public Property MaxPoolSize As Integer
```

C#

```
public int MaxPoolSize {get;set;}
```

1.8.37 MinPoolSize Property

Gets or sets the MinPoolSize connection property.

Syntax

Visual Basic

```
Public Property MinPoolSize As Integer
```

C#

```
public int MinPoolSize {get;set;}
```

1.8.38 NewPassword Property

Gets or sets the NewPassword connection property.

Syntax

Visual Basic

```
Public Property NewPassword As String
```

C#

```
public string NewPassword {get;set;}
```

1.8.39 NodeType Property

Gets or sets the NodeType connection property.

Syntax

Visual Basic

```
Public Property NodeType As String
```

C#

```
public string NodeType {get;set;}
```

1.8.40 Password Property

Gets or sets the Password connection property.

Syntax

Visual Basic

```
Public Property Password As String
```

C#

```
public string Password {get;set;}
```

1.8.41 PersistSecurityInfo Property

Gets or sets the PersistSecurityInfo connection property.

Syntax

Visual Basic

```
Public Property PersistSecurityInfo As Boolean
```

C#

```
public bool PersistSecurityInfo {get;set;}
```

1.8.42 Pooling Property

Gets or sets the Pooling connection property.

Syntax

Visual Basic

```
Public Property Pooling As Boolean
```

C#

```
public bool Pooling {get;set;}
```

1.8.43 PrefetchBuffer Property

Gets or sets the PrefetchBuffer connection property.

Syntax

Visual Basic

```
Public Property PrefetchBuffer As Integer
```

C#

```
public int PrefetchBuffer {get;set;}
```

1.8.44 PrefetchRows Property

Gets or sets the PrefetchRows connection property.

Syntax

Visual Basic

```
Public Property PrefetchRows As Integer
```

C#

```
public int PrefetchRows {get;set;}
```

Remarks

The default value is 200.

1.8.45 RetryConnectionTimeout Property

Gets or sets the RetryConnectionTimeout connection property.

Syntax

Visual Basic

```
Public Property RetryConnectionTimeout As Integer
```

C#

```
public int RetryConnectionTimeout {get;set;}
```

1.8.46 ServerName Property

Gets or sets the ServerName connection property.

Syntax

Visual Basic

```
Public Property ServerName As String
```

C#

```
public string ServerName {get;set;}
```

1.8.47 StartLine Property

Gets or sets the StartLine connection property.

Syntax

Visual Basic

```
Public Property StartLine As String
```

C#

```
public string StartLine {get;set;}
```

1.8.48 Unconditional Property

Gets or sets the Unconditional connection property.

Syntax

Visual Basic

```
Public Property Unconditional As String
```

C#

```
public string Unconditional {get;set;}
```

1.8.49 UserID Property

Gets or sets the UserID connection property.

Syntax

Visual Basic

```
Public Property UserID As String
```

C#

```
public string UserID {get;set;}
```

1.9 SAConnectionStringBuilderBase Class

Base class of the SAConnectionStringBuilder class.

Syntax

Visual Basic

```
Public MustInherit Class SAConnectionStringBuilderBase Inherits
System.Data.Common.DbConnectionStringBuilder
```

C#

```
public abstract class SAConnectionStringBuilderBase :
System.Data.Common.DbConnectionStringBuilder
```

Members

All members of SAConnectionStringBuilderBase, including inherited members.

Methods

Modifier and Type	Method	Description
public override bool	ContainsKey(string) [page 172]	Determines whether the SAConnection-StringBuilder object contains a specific keyword.
public string	GetKeyword(string) [page 173]	Gets the keyword for the specified SA-ConnectionStringBuilder property.
public bool	GetUseLongNameAsKeyword() [page 174]	Gets a boolean value that indicates whether long connection parameter names are used in the connection string.
public override bool	Remove(string) [page 174]	Removes the entry with the specified key from the SAConnectionString-Builder instance.
public void	SetUseLongNameAsKeyword(bool) [page 175]	Sets a boolean value that indicates whether long connection parameter names are used in the connection string.
public override bool	ShouldSerialize(string) [page 176]	Indicates whether the specified key ex-ists in this SAConnectionStringBuilder instance.
public override bool	TryGetValue(string, out object) [page 176]	Retrieves a value corresponding to the supplied key from this SAConnection-StringBuilder.

Properties

Modifier and Type	Property	Description
public override ICollection	Keys [page 177]	Gets an System.Collections.ICollection that contains the keys in the SAConnectionStringBuilder.
public override object	this[string keyword] [page 177]	Gets or sets the value of the connection keyword.

In this section:

ContainsKey(string) Method [page 172]
Determines whether the SAConnectionStringBuilder object contains a specific keyword.

GetKeyword(string) Method [page 173]
Gets the keyword for the specified SAConnectionStringBuilder property.

GetUseLongNameAsKeyword() Method [page 174]
Gets a boolean value that indicates whether long connection parameter names are used in the connection string.

Remove(string) Method [page 174]
Removes the entry with the specified key from the SAConnectionStringBuilder instance.

SetUseLongNameAsKeyword(bool) Method [page 175]
Sets a boolean value that indicates whether long connection parameter names are used in the connection string.

ShouldSerialize(string) Method [page 176]
Indicates whether the specified key exists in this SAConnectionStringBuilder instance.

TryGetValue(string, out object) Method [page 176]
Retrieves a value corresponding to the supplied key from this SAConnectionStringBuilder.

Keys Property [page 177]
Gets an System.Collections.ICollection that contains the keys in the SAConnectionStringBuilder.

this[string keyword] Property [page 177]
Gets or sets the value of the connection keyword.

1.9.1 ContainsKey(string) Method

Determines whether the SAConnectionStringBuilder object contains a specific keyword.

Syntax

Visual Basic

```
Public Overrides Function ContainsKey (ByVal keyword As String) As Boolean
```

C#

```
public override bool ContainsKey (string keyword)
```

Parameters

keyword The keyword to locate in the SAConnectionStringBuilder.

Returns

True if the value associated with keyword has been set; false otherwise.

Example

The following statement determines whether the SAConnectionStringBuilder object contains the UserID keyword.

```
connectString.ContainsKey("UserID")
```

1.9.2 GetKeyword(string) Method

Gets the keyword for the specified SAConnectionStringBuilder property.

Syntax

Visual Basic

```
Public Function GetKeyword (ByVal propName As String) As String
```

C#

```
public string GetKeyword (string propName)
```

Parameters

propName The name of the SAConnectionStringBuilder property.

Returns

The keyword for the specified SAConnectionStringBuilder property.

1.9.3 GetUseLongNameAsKeyword() Method

Gets a boolean value that indicates whether long connection parameter names are used in the connection string.

Syntax

Visual Basic

```
Public Function GetUseLongNameAsKeyword () As Boolean
```

C#

```
public bool GetUseLongNameAsKeyword ()
```

Returns

True if long connection parameter names are used to build connection strings; false otherwise.

Remarks

Connection parameters have both long and short forms of their names. For example, to specify the name of an ODBC data source in your connection string, use either of the following values: DataSourceName or DSN. By default, long connection parameter names are used to build connection strings.

Related Information

SetUseLongNameAsKeyword(bool) Method [page 175]

1.9.4 Remove(string) Method

Removes the entry with the specified key from the SAConnectionStringBuilder instance.

Syntax

Visual Basic

```
Public Overrides Function Remove (ByVal keyword As String) As Boolean
```

C#

```
public override bool Remove (string keyword)
```

Parameters

keyword The key of the key/value pair to be removed from the connection string in this SAConnectionStringBuilder.

Returns

True if the key existed within the connection string and was removed; false if the key did not exist.

1.9.5 SetUseLongNameAsKeyword(bool) Method

Sets a boolean value that indicates whether long connection parameter names are used in the connection string.

Syntax

Visual Basic

```
Public Sub SetUseLongNameAsKeyword (ByVal useLongNameAsKeyword As Boolean)
```

C#

```
public void SetUseLongNameAsKeyword (bool useLongNameAsKeyword)
```

Parameters

useLongNameAsKeyword A boolean value that indicates whether the long connection parameter name is used in the connection string.

Remarks

Long connection parameter names are used by default.

Related Information

GetUseLongNameAsKeyword() Method [page 174]

1.9.6 ShouldSerialize(string) Method

Indicates whether the specified key exists in this SAConnectionStringBuilder instance.

Syntax

Visual Basic

```
Public Overrides Function ShouldSerialize (ByVal keyword As String) As
Boolean
```

C#

```
public override bool ShouldSerialize (string keyword)
```

Parameters

keyword The key to locate in the SAConnectionStringBuilder.

Returns

True if the SAConnectionStringBuilder contains an entry with the specified key; false otherwise.

1.9.7 TryGetValue(string, out object) Method

Retrieves a value corresponding to the supplied key from this SAConnectionStringBuilder.

Syntax

Visual Basic

```
Public Overrides Function TryGetValue (
    ByVal keyword As String,
    ByVal value As Object
) As Boolean
```

C#

```
public override bool TryGetValue (
    string keyword,
    out object value
)
```

Parameters

keyword The key of the item to retrieve.
value The value corresponding to keyword.

Returns

True if keyword was found within the connection string; false otherwise.

1.9.8 Keys Property

Gets an System.Collections.ICollection that contains the keys in the SAConnectionStringBuilder.

Syntax

Visual Basic

```
Public ReadOnly Overrides Property Keys As ICollection
```

C#

```
public override ICollection Keys {get;}
```

Returns

An System.Collections.ICollection that contains the keys in the SAConnectionStringBuilder.

1.9.9 this[string keyword] Property

Gets or sets the value of the connection keyword.

Syntax

Visual Basic

```
Public Overrides Property Item (ByVal keyword As String) As Object
```

C#

```
public override object this[string keyword] {get;set;}
```

Remarks

An object representing the value of the specified connection keyword.

If the keyword or type is invalid, then an exception is raised. The parameter value is case insensitive.

When setting the value, passing NULL clears the value.

1.10 SACredential Class

SACredential provides a more secure way to specify the password for a login attempt using database server authentication.

Syntax

Visual Basic

```
Public NotInheritable Class SACredential
```

C#

```
public sealed class SACredential
```

Members

All members of SACredential, including inherited members.

Constructors

Modifier and Type	Constructor	Description
public	SACredential(string, SecureString) [page 179]	Initializes an SACredential object.

Properties

Modifier and Type	Property	Description
public SecureString	Password [page 180]	Returns the password of the SACredential object.
public string	UserId [page 180]	Returns the user ID of the SACredential object.

Remarks

SACredential is comprised of a user ID and a password that are used for database server authentication. The password in a SACredential object is of type System.Security.SecureString. SACredential cannot be inherited.

In this section:

SACredential(string, SecureString) Constructor [page 179]
Initializes an SACredential object.

Password Property [page 180]
Returns the password of the SACredential object.

UserId Property [page 180]
Returns the user ID of the SACredential object.

1.10.1 SACredential(string, SecureString) Constructor

Initializes an SACredential object.

Syntax

Visual Basic

```
Public Sub SACredential (
    ByVal userId As String,
    ByVal password As SecureString
)
```

C#

```
public SACredential (
    string userId,
    SecureString password
)
```

Parameters

userId The user ID.

password The password; a System.Security.SecureString value marked as read-only. Passing a read/write System.Security.SecureString parameter will raise an System.ArgumentException.

Related Information

SAConnection Class [page 118]

1.10.2 Password Property

Returns the password of the SACredential object.

Syntax

Visual Basic

```
Public ReadOnly Property Password As SecureString
```

C#

```
public SecureString Password {get;}
```

Returns

The password of the SACredential object.

1.10.3 UserId Property

Returns the user ID of the SACredential object.

Syntax

Visual Basic

```
Public ReadOnly Property UserId As String
```

C#

```
public string UserId {get;}
```

Returns

The user ID of the SACredential object.

1.11 SADataAdapter Class

Represents a set of commands and a database connection used to fill a System.Data.DataSet and to update a database.

Syntax

Visual Basic

```
Public NotInheritable Class SADataAdapter Inherits
System.Data.Common.DbDataAdapter Implements System.ICloneable
```

C#

```
public sealed class SADataAdapter : System.Data.Common.DbDataAdapter,
System.ICloneable
```

Members

All members of SADataAdapter, including inherited members.

Constructors

Modifier and Type	Constructor	Description
public	SADataAdapter [page 184]	Initializes an SADataAdapter object.

Methods

Modifier and Type	Method	Description
protected override void	ClearBatch() [page 188]	Removes all SACommand objects from the batch.
protected override RowUpdatedEventArgs	CreateRowUpdatedEvent(DataRow, IDbCommand, StatementType, DataTableMapping) [page 188]	Initializes a new instance of the System.Data.Common.RowUpdatedEventArgs class.
protected override RowUpdatingEventArgs	CreateRowUpdatingEvent(DataRow, IDbCommand, StatementType, DataTableMapping) [page 189]	Initializes a new instance of the System.Data.Common.RowUpdatingEventArgs class.
protected override void	Dispose(bool) [page 190]	Releases the unmanaged resources used by the SADataAdapter object and optionally releases the managed resources.
protected override int	Fill [page 191]	Adds or refreshes rows in a System.Data.DataSet or System.Data.DataTable object with data from the database.

Modifier and Type	Method	Description
protected override DataTable[]	FillSchema [page 194]	Adds a System.Data.DataTable to a System.Data.DataSet and configures the schema to match the schema in the data source.
public new SAParameter[]	GetFillParameters() [page 197]	Returns the parameters set by you when executing a SELECT statement.
protected override void	InitializeBatching() [page 197]	Initializes batching for the SADataAdapter object.
protected override void	OnRowUpdated(RowUpdatedEventArgs) [page 198]	Raises the RowUpdated event of a .NET Framework data provider.
protected override void	OnRowUpdating(RowUpdatingEventArgs) [page 198]	Raises the RowUpdating event of a .NET Framework data provider.
protected override void	TerminateBatching() [page 199]	Ends batching for the SADataAdapter object.
protected override int	Update(DataRow[], DataTableMapping) [page 199]	Updates the tables in a database with the changes made to the DataSet.

Properties

Modifier and Type	Property	Description
public new SACommand	DeleteCommand [page 200]	Specifies an SACommand object that is executed against the database when the Update method is called to delete rows in the database that correspond to deleted rows in the DataSet.
public new SACommand	InsertCommand [page 201]	Specifies an SACommand that is executed against the database when the Update method is called that adds rows to the database to correspond to rows that were inserted in the DataSet.
public new SACommand	SelectCommand [page 201]	Specifies an SACommand that is used during Fill or FillSchema to obtain a result set from the database for copying into a DataSet.
public new DataTableMappingCollection	TableMappings [page 202]	Specifies a collection that provides the master mapping between a source table and a DataTable.
public override int	UpdateBatchSize [page 202]	Gets or sets the number of rows that are processed in each round-trip to the database server.
public new SACommand	UpdateCommand [page 203]	Specifies an SACommand that is executed against the database when the Update method is called to update rows in the database that correspond to updated rows in the DataSet.

Events

Modifier and Type	Event	Description
public SARowUpdatedEventHandler	RowUpdated [page 204]	Occurs during an update after a command is executed against the data source.
public SARowUpdatingEventHandler	RowUpdating [page 204]	Occurs during an update before a command is executed against the data source.

Remarks

The System.Data.DataSet provides a way to work with data offline. The SADataAdapter provides methods to associate a DataSet with a set of SQL statements.

Implements: IDbDataAdapter, IDataAdapter, ICloneable

For more information, see "Data access and manipulation".

In this section:

SADataAdapter Constructor [page 184]
Initializes an SADataAdapter object.

ClearBatch() Method [page 188]
Removes all SACommand objects from the batch.

CreateRowUpdatedEvent(DataRow, IDbCommand, StatementType, DataTableMapping) Method [page 188]
Initializes a new instance of the System.Data.Common.RowUpdatedEventArgs class.

CreateRowUpdatingEvent(DataRow, IDbCommand, StatementType, DataTableMapping) Method [page 189]
Initializes a new instance of the System.Data.Common.RowUpdatingEventArgs class.

Dispose(bool) Method [page 190]
Releases the unmanaged resources used by the SADataAdapter object and optionally releases the managed resources.

Fill Method [page 191]
Adds or refreshes rows in a System.Data.DataSet or System.Data.DataTable object with data from the database.

FillSchema Method [page 194]
Adds a System.Data.DataTable to a System.Data.DataSet and configures the schema to match the schema in the data source.

GetFillParameters() Method [page 197]
Returns the parameters set by you when executing a SELECT statement.

InitializeBatching() Method [page 197]
Initializes batching for the SADataAdapter object.

OnRowUpdated(RowUpdatedEventArgs) Method [page 198]
Raises the RowUpdated event of a .NET Framework data provider.

OnRowUpdating(RowUpdatingEventArgs) Method [page 198]
Raises the RowUpdating event of a .NET Framework data provider.

TerminateBatching() Method [page 199]
Ends batching for the SADataAdapter object.

Update(DataRow[], DataTableMapping) Method [page 199]
Updates the tables in a database with the changes made to the DataSet.

DeleteCommand Property [page 200]
Specifies an SACommand object that is executed against the database when the Update method is called to delete rows in the database that correspond to deleted rows in the DataSet.

InsertCommand Property [page 201]
Specifies an SACommand that is executed against the database when the Update method is called that adds rows to the database to correspond to rows that were inserted in the DataSet.

SelectCommand Property [page 201]
Specifies an SACommand that is used during Fill or FillSchema to obtain a result set from the database for copying into a DataSet.

TableMappings Property [page 202]
Specifies a collection that provides the master mapping between a source table and a DataTable.

UpdateBatchSize Property [page 202]
Gets or sets the number of rows that are processed in each round-trip to the database server.

UpdateCommand Property [page 203]
Specifies an SACommand that is executed against the database when the Update method is called to update rows in the database that correspond to updated rows in the DataSet.

RowUpdated Event [page 204]
Occurs during an update after a command is executed against the data source.

RowUpdating Event [page 204]
Occurs during an update before a command is executed against the data source.

1.11.1 SADataAdapter Constructor

Initializes an SADataAdapter object.

Overload list

Modifier and Type	Overload name	Description
public	SADataAdapter() [page 185]	Initializes an SADataAdapter object.
public	SADataAdapter(SACommand) [page 186]	Initializes an SADataAdapter object with the specified SELECT statement.

Modifier and Type	Overload name	Description
public	SADataAdapter(string, SAConnection) [page 186]	Initializes an SADataAdapter object with the specified SELECT statement and connection.
public	SADataAdapter(string, string) [page 187]	Initializes an SADataAdapter object with the specified SELECT statement and connection string.

In this section:

SADataAdapter() Constructor [page 185]
Initializes an SADataAdapter object.

SADataAdapter(SACommand) Constructor [page 186]
Initializes an SADataAdapter object with the specified SELECT statement.

SADataAdapter(string, SAConnection) Constructor [page 186]
Initializes an SADataAdapter object with the specified SELECT statement and connection.

SADataAdapter(string, string) Constructor [page 187]
Initializes an SADataAdapter object with the specified SELECT statement and connection string.

1.11.1.1 SADataAdapter() Constructor

Initializes an SADataAdapter object.

Syntax

Visual Basic

```
Public Sub SADataAdapter ()
```

C#

```
public SADataAdapter ()
```

Related Information

SADataAdapter(SACommand) Constructor [page 186]
SADataAdapter(string, SAConnection) Constructor [page 186]
SADataAdapter(string, string) Constructor [page 187]

1.11.1.2 SADataAdapter(SACommand) Constructor

Initializes an SADataAdapter object with the specified SELECT statement.

Syntax

Visual Basic

```
Public Sub SADataAdapter (ByVal selectCommand As SACommand)
```

C#

```
public SADataAdapter (SACommand selectCommand)
```

Parameters

selectCommand An SACommand object that is used during System.Data.Common.DbDataAdapter.Fill(System.Data.DataSet) to select records from the data source for placement in the System.Data.DataSet.

Related Information

SADataAdapter() Constructor [page 185]
SADataAdapter(string, SAConnection) Constructor [page 186]
SADataAdapter(string, string) Constructor [page 187]

1.11.1.3 SADataAdapter(string, SAConnection) Constructor

Initializes an SADataAdapter object with the specified SELECT statement and connection.

Syntax

Visual Basic

```
Public Sub SADataAdapter (
    ByVal selectCommandText As String,
    ByVal selectConnection As SAConnection
)
```

C#

```
public SADataAdapter (
    string selectCommandText,
    SAConnection selectConnection
)
```

Parameters

selectCommandText A SELECT statement to be used to set the SADataAdapter.SelectCommand property of the SADataAdapter object.
selectConnection An SAConnection object that defines a connection to a database.

Related Information

SADataAdapter() Constructor [page 185]
SADataAdapter(SACommand) Constructor [page 186]
SADataAdapter(string, string) Constructor [page 187]
SelectCommand Property [page 201]
SAConnection Class [page 118]

1.11.1.4 SADataAdapter(string, string) Constructor

Initializes an SADataAdapter object with the specified SELECT statement and connection string.

Syntax

Visual Basic

```
Public Sub SADataAdapter (
    ByVal selectCommandText As String,
    ByVal selectConnectionString As String
)
```

C#

```
public SADataAdapter (
    string selectCommandText,
    string selectConnectionString
)
```

Parameters

selectCommandText A SELECT statement to be used to set the SADataAdapter.SelectCommand property of the SADataAdapter object.
selectConnectionString A connection string for a database.

Related Information

SADataAdapter() Constructor [page 185]
SADataAdapter(SACommand) Constructor [page 186]
SADataAdapter(string, SAConnection) Constructor [page 186]
SelectCommand Property [page 201]

1.11.2 ClearBatch() Method

Removes all SACommand objects from the batch.

Syntax

Visual Basic

```
Protected Overrides Sub ClearBatch ()
```

C#

```
protected override void ClearBatch ()
```

Related Information

SACommand Class [page 53]

1.11.3 CreateRowUpdatedEvent(DataRow, IDbCommand, StatementType, DataTableMapping) Method

Initializes a new instance of the System.Data.Common.RowUpdatedEventArgs class.

Syntax

Visual Basic

```
Protected Overrides Function CreateRowUpdatedEvent (
    ByVal dataRow As DataRow,
    ByVal command As IDbCommand,
    ByVal statementType As StatementType,
    ByVal tableMapping As DataTableMapping
) As RowUpdatedEventArgs
```

C#

```
protected override RowUpdatedEventArgs CreateRowUpdatedEvent (
    DataRow dataRow,
    IDbCommand command,
```

```
    StatementType statementType,
    DataTableMapping tableMapping
)
```

Parameters

dataRow The System.Data.DataRow used to update the data source.

command The System.Data.IDbCommand executed during the System.Data.IDataAdapter.Update(System.Data.DataSet).

statementType Whether the command is an UPDATE, INSERT, DELETE, or SELECT statement.

tableMapping A System.Data.Common.DataTableMapping object.

Returns

A new instance of the System.Data.Common.RowUpdatedEventArgs class.

1.11.4 CreateRowUpdatingEvent(DataRow, IDbCommand, StatementType, DataTableMapping) Method

Initializes a new instance of the System.Data.Common.RowUpdatingEventArgs class.

Syntax

Visual Basic

```
Protected Overrides Function CreateRowUpdatingEvent (
    ByVal dataRow As DataRow,
    ByVal command As IDbCommand,
    ByVal statementType As StatementType,
    ByVal tableMapping As DataTableMapping
) As RowUpdatingEventArgs
```

C#

```
protected override RowUpdatingEventArgs CreateRowUpdatingEvent (
    DataRow dataRow,
    IDbCommand command,
    StatementType statementType,
    DataTableMapping tableMapping
)
```

Parameters

dataRow The System.Data.DataRow used to update the data source.
command The System.Data.IDbCommand executed during the System.Data.IDataAdapter.Update(System.Data.DataSet).
statementType Whether the command is an UPDATE, INSERT, DELETE, or SELECT statement.
tableMapping A System.Data.Common.DataTableMapping object.

Returns

A new instance of the System.Data.Common.RowUpdatingEventArgs class.

1.11.5 Dispose(bool) Method

Releases the unmanaged resources used by the SADataAdapter object and optionally releases the managed resources.

Syntax

Visual Basic

```
Protected Overrides Sub Dispose (ByVal disposing As Boolean)
```

C#

```
protected override void Dispose (bool disposing)
```

Parameters

disposing True releases both managed and unmanaged resources; false releases only unmanaged resources.

Related Information

SADataAdapter Class [page 181]

1.11.6 Fill Method

Adds or refreshes rows in a System.Data.DataSet or System.Data.DataTable object with data from the database.

Overload list

Modifier and Type	Overload name	Description
protected override int	Fill(DataSet, int, int, string, IDbCommand, CommandBehavior) [page 191]	Adds or refreshes rows in a System.Data.DataSet or System.Data.DataTable object with data from the database.
protected override int	Fill(DataTable[], int, int, IDbCommand, CommandBehavior) [page 193]	Adds or refreshes rows in a specified range in the System.Data.DataSet to match those in the data source using the System.Data.DataSet and System.Data.DataTable names.

In this section:

Fill(DataSet, int, int, string, IDbCommand, CommandBehavior) Method [page 191]
Adds or refreshes rows in a System.Data.DataSet or System.Data.DataTable object with data from the database.

Fill(DataTable[], int, int, IDbCommand, CommandBehavior) Method [page 193]
Adds or refreshes rows in a specified range in the System.Data.DataSet to match those in the data source using the System.Data.DataSet and System.Data.DataTable names.

1.11.6.1 Fill(DataSet, int, int, string, IDbCommand, CommandBehavior) Method

Adds or refreshes rows in a System.Data.DataSet or System.Data.DataTable object with data from the database.

Syntax

Visual Basic

```
Protected Overrides Function Fill (
    ByVal dataSet As DataSet,
    ByVal startRecord As Integer,
    ByVal maxRecords As Integer,
    ByVal srcTable As String,
    ByVal command As IDbCommand,
    ByVal behavior As CommandBehavior
) As Integer
```

C#

```
protected override int Fill (
    DataSet dataSet,
    int startRecord,
    int maxRecords,
    string srcTable,
    IDbCommand command,
    CommandBehavior behavior
)
```

Parameters

dataSet A System.Data.DataSet to fill with records and optionally, schema.
startRecord The zero-based record number with which to start.
maxRecords The maximum number of records to be read into the System.Data.DataSet.
srcTable The name of the source table to use for table mapping.
command The SQL SELECT statement used to retrieve rows from the data source.
behavior One of the CommandBehavior values.

Returns

The number of rows successfully added or refreshed in the System.Data.DataSet.

Remarks

Even if you use the startRecord argument to limit the number of records that are copied to the DataSet, all records in the SADataAdapter query are fetched from the database to the client. For large result sets, this can have a significant performance impact.

An alternative is to use an SADataReader when a read-only, forward-only result set is sufficient, perhaps with SQL statements (ExecuteNonQuery) to undertake modifications. Another alternative is to write a stored procedure that returns only the result you need.

If SelectCommand does not return any rows, then no tables are added to the DataSet and no exception is raised.

For more information, see "Data access and manipulation".

1.11.6.2 Fill(DataTable[], int, int, IDbCommand, CommandBehavior) Method

Adds or refreshes rows in a specified range in the System.Data.DataSet to match those in the data source using the System.Data.DataSet and System.Data.DataTable names.

Syntax

Visual Basic

```
Protected Overrides Function Fill (
    ByVal dataTables As DataTable(),
    ByVal startRecord As Integer,
    ByVal maxRecords As Integer,
    ByVal command As IDbCommand,
    ByVal behavior As CommandBehavior
) As Integer
```

C#

```
protected override int Fill (
    DataTable[] dataTables,
    int startRecord,
    int maxRecords,
    IDbCommand command,
    CommandBehavior behavior
)
```

Parameters

dataTables The System.Data.DataTable objects to fill from the data source.
startRecord The zero-based record number to start with.
maxRecords The maximum number of records to retrieve.
command The System.Data.IDbCommand executed to fill the System.Data.DataTable objects.
behavior One of the System.Data.CommandBehavior values.

Returns

The number of rows added to or refreshed in the data tables.

Remarks

Even if you use the startRecord argument to limit the number of records that are copied to the DataSet, all records in the SADataAdapter query are fetched from the database to the client. For large result sets, this can have a significant performance impact.

An alternative is to use an SADataReader when a read-only, forward-only result set is sufficient, perhaps with SQL statements (ExecuteNonQuery) to undertake modifications. Another alternative is to write a stored procedure that returns only the result you need.

If SelectCommand does not return any rows, then no tables are added to the DataSet and no exception is raised.

For more information, see "Data access and manipulation".

1.11.7 FillSchema Method

Adds a System.Data.DataTable to a System.Data.DataSet and configures the schema to match the schema in the data source.

Overload list

Modifier and Type	Overload name	Description
protected override DataTable[]	FillSchema(DataSet, SchemaType, IDbCommand, string, CommandBehavior) [page 195]	Adds a System.Data.DataTable to a System.Data.DataSet and configures the schema to match the schema in the data source.
protected override DataTable	FillSchema(DataTable, SchemaType, IDbCommand, CommandBehavior) [page 196]	Adds a System.Data.DataTable to a System.Data.DataSet and configures the schema to match the schema in the data source.

In this section:

FillSchema(DataSet, SchemaType, IDbCommand, string, CommandBehavior) Method [page 195]
Adds a System.Data.DataTable to a System.Data.DataSet and configures the schema to match the schema in the data source.

FillSchema(DataTable, SchemaType, IDbCommand, CommandBehavior) Method [page 196]
Adds a System.Data.DataTable to a System.Data.DataSet and configures the schema to match the schema in the data source.

1.11.7.1 FillSchema(DataSet, SchemaType, IDbCommand, string, CommandBehavior) Method

Adds a System.Data.DataTable to a System.Data.DataSet and configures the schema to match the schema in the data source.

Syntax

Visual Basic

```
Protected Overrides Function FillSchema (
    ByVal dataSet As DataSet,
    ByVal schemaType As SchemaType,
    ByVal command As IDbCommand,
    ByVal srcTable As String,
    ByVal behavior As CommandBehavior
) As DataTable()
```

C#

```
protected override DataTable[] FillSchema (
    DataSet dataSet,
    SchemaType schemaType,
    IDbCommand command,
    string srcTable,
    CommandBehavior behavior
)
```

Parameters

dataSet A System.Data.DataSet to fill with the schema.
schemaType One of the System.Data.SchemaType values that specify how to insert the schema.
command The SQL SELECT statement used to retrieve rows from the data source.
srcTable The name of the source table to use for table mapping.
behavior One of the System.Data.CommandBehavior values.

Returns

A reference to a collection of System.Data.DataTable objects that were added to the System.Data.DataSet.

Remarks

For more information, see System.Data.IDataAdapter.FillSchema and "Data access and manipulation".

1.11.7.2 FillSchema(DataTable, SchemaType, IDbCommand, CommandBehavior) Method

Adds a System.Data.DataTable to a System.Data.DataSet and configures the schema to match the schema in the data source.

Syntax

Visual Basic

```
Protected Overrides Function FillSchema (
    ByVal dataTable As DataTable,
    ByVal schemaType As SchemaType,
    ByVal command As IDbCommand,
    ByVal behavior As CommandBehavior
) As DataTable
```

C#

```
protected override DataTable FillSchema (
    DataTable dataTable,
    SchemaType schemaType,
    IDbCommand command,
    CommandBehavior behavior
)
```

Parameters

dataTable A System.Data.DataTable to fill with the schema.
schemaType One of the System.Data.SchemaType values that specify how to insert the schema.
command The SQL SELECT statement used to retrieve rows from the data source.
behavior One of the System.Data.CommandBehavior values.

Returns

A reference to the System.Data.DataTable object that contains the schema.

Remarks

For more information, see System.Data.Common.DbDataAdapter.FillSchema(System.Data.DataTable,System.Data.SchemaType) and "Data access and manipulation".

1.11.8 GetFillParameters() Method

Returns the parameters set by you when executing a SELECT statement.

Syntax

Visual Basic

```
Public Shadows Function GetFillParameters () As SAParameter()
```

C#

```
public new SAParameter[] GetFillParameters ()
```

Returns

An array of IDataParameter objects that contains the parameters set by the user.

1.11.9 InitializeBatching() Method

Initializes batching for the SADataAdapter object.

Syntax

Visual Basic

```
Protected Overrides Sub InitializeBatching ()
```

C#

```
protected override void InitializeBatching ()
```

Related Information

SADataAdapter Class [page 181]

1.11.10 OnRowUpdated(RowUpdatedEventArgs) Method

Raises the RowUpdated event of a .NET Framework data provider.

Syntax

Visual Basic

```
Protected Overrides Sub OnRowUpdated (ByVal value As RowUpdatedEventArgs)
```

C#

```
protected override void OnRowUpdated (RowUpdatedEventArgs value)
```

Parameters

value A System.Data.Common.RowUpdatedEventArgs that contains the event data.

1.11.11 OnRowUpdating(RowUpdatingEventArgs) Method

Raises the RowUpdating event of a .NET Framework data provider.

Syntax

Visual Basic

```
Protected Overrides Sub OnRowUpdating (ByVal value As RowUpdatingEventArgs)
```

C#

```
protected override void OnRowUpdating (RowUpdatingEventArgs value)
```

Parameters

value A System.Data.Common.RowUpdatingEventArgs that contains the event data.

1.11.12 TerminateBatching() Method

Ends batching for the SADataAdapter object.

Syntax

Visual Basic

```
Protected Overrides Sub TerminateBatching ()
```

C#

```
protected override void TerminateBatching ()
```

Related Information

SADataAdapter Class [page 181]

1.11.13 Update(DataRow[], DataTableMapping) Method

Updates the tables in a database with the changes made to the DataSet.

Syntax

Visual Basic

```
Protected Overrides Function Update (
    ByVal dataRows As DataRow(),
    ByVal tableMapping As DataTableMapping
) As Integer
```

C#

```
protected override int Update (
    DataRow[] dataRows,
    DataTableMapping tableMapping
)
```

Parameters

dataRows An array of System.Data.DataRow to update from.
tableMapping The System.Data.IDataAdapter.TableMappings collection to use.

Returns

The number of rows successfully updated from the System.Data.DataRow array.

Remarks

The Update is carried out using the InsertCommand, UpdateCommand, and DeleteCommand on each row in the data set that has been inserted, updated, or deleted.

For more information, see "Data access and manipulation".

Related Information

DeleteCommand Property [page 200]
InsertCommand Property [page 201]
UpdateCommand Property [page 203]

1.11.14 DeleteCommand Property

Specifies an SACommand object that is executed against the database when the Update method is called to delete rows in the database that correspond to deleted rows in the DataSet.

Syntax

Visual Basic

```
Public Shadows Property DeleteCommand As SACommand
```

C#

```
public new SACommand DeleteCommand {get;set;}
```

Remarks

If this property is not set and primary key information is present in the DataSet during Update, then DeleteCommand can be generated automatically by setting SelectCommand and using the SACommandBuilder. In that case, the SACommandBuilder generates any additional commands that you do not set. This generation logic requires key column information to be present in the SelectCommand.

When DeleteCommand is assigned to an existing SACommand object, the SACommand object is not cloned. The DeleteCommand maintains a reference to the existing SACommand.

Related Information

SelectCommand Property [page 201]

1.11.15 InsertCommand Property

Specifies an SACommand that is executed against the database when the Update method is called that adds rows to the database to correspond to rows that were inserted in the DataSet.

Syntax

Visual Basic

```
Public Shadows Property InsertCommand As SACommand
```

C#

```
public new SACommand InsertCommand {get;set;}
```

Remarks

The SACommandBuilder does not require key columns to generate InsertCommand.

When InsertCommand is assigned to an existing SACommand object, the SACommand is not cloned. The InsertCommand maintains a reference to the existing SACommand.

If this command returns rows, then the rows may be added to the DataSet depending on how you set the UpdatedRowSource property of the SACommand object.

1.11.16 SelectCommand Property

Specifies an SACommand that is used during Fill or FillSchema to obtain a result set from the database for copying into a DataSet.

Syntax

Visual Basic

```
Public Shadows Property SelectCommand As SACommand
```

C#

```
public new SACommand SelectCommand {get;set;}
```

Remarks

When SelectCommand is assigned to a previously created SACommand, the SACommand is not cloned. The SelectCommand maintains a reference to the previously created SACommand object.

If the SelectCommand does not return any rows, then no tables are added to the DataSet, and no exception is raised.

The SELECT statement can also be specified in the SADataAdapter constructor.

1.11.17 TableMappings Property

Specifies a collection that provides the master mapping between a source table and a DataTable.

Syntax

Visual Basic

```
Public ReadOnly Shadows Property TableMappings As
DataTableMappingCollection
```

C#

```
public new DataTableMappingCollection TableMappings {get;}
```

Remarks

The default value is an empty collection.

When reconciling changes, the SADataAdapter uses the DataTableMappingCollection collection to associate the column names used by the data source with the column names used by the DataSet.

1.11.18 UpdateBatchSize Property

Gets or sets the number of rows that are processed in each round-trip to the database server.

Syntax

Visual Basic

```
Public Overrides Property UpdateBatchSize As Integer
```

C#

```
public override int UpdateBatchSize {get;set;}
```

Remarks

The default value is 1.

Setting the value to something greater than 1 causes SADataAdapter.Update to execute all the insert statements in batches. The deletions and updates are executed sequentially as before, but insertions are executed afterward in batches of size equal to the value of UpdateBatchSize. Setting the value to 0 causes Update to send the insert statements in a single batch.

Setting the value to something greater than 1 causes SADataAdapter.Fill to execute all the insert statements in batches. The deletions and updates are executed sequentially as before, but insertions are executed afterward in batches of size equal to the value of UpdateBatchSize.

Setting the value to 0 causes Fill to send the insert statements in a single batch.

Setting it less than 0 is an error.

If UpdateBatchSize is set to something other than one, and the InsertCommand property is set to something that is not an INSERT statement, then an exception is thrown when calling Fill.

This behavior is different from SqlDataAdapter. It batches all types of commands.

1.11.19 UpdateCommand Property

Specifies an SACommand that is executed against the database when the Update method is called to update rows in the database that correspond to updated rows in the DataSet.

Syntax

Visual Basic

```
Public Shadows Property UpdateCommand As SACommand
```

C#

```
public new SACommand UpdateCommand {get;set;}
```

Remarks

During Update, if this property is not set and primary key information is present in the SelectCommand, then the UpdateCommand can be generated automatically if you set the SelectCommand property and use the SACommandBuilder. Then, any additional commands that you do not set are generated by the SACommandBuilder. This generation logic requires key column information to be present in the SelectCommand.

When UpdateCommand is assigned to a previously created SACommand, the SACommand is not cloned. The UpdateCommand maintains a reference to the previously created SACommand object.

If execution of this command returns rows, then these rows can be merged with the DataSet depending on how you set the UpdatedRowSource property of the SACommand object.

1.11.20 RowUpdated Event

Occurs during an update after a command is executed against the data source.

Syntax

Visual Basic

```
Public Event RowUpdated  As SARowUpdatedEventHandler
```

C#

```
public SARowUpdatedEventHandler RowUpdated;
```

Remarks

When an attempt to update is made, the event fires.

The event handler receives an argument of type SARowUpdatedEventArgs containing data related to this event.

For more information, see the .NET Framework documentation for OleDbDataAdapter.RowUpdated Event.

1.11.21 RowUpdating Event

Occurs during an update before a command is executed against the data source.

Syntax

Visual Basic

```
Public Event RowUpdating  As SARowUpdatingEventHandler
```

C#

```
public SARowUpdatingEventHandler RowUpdating;
```

Remarks

When an attempt to update is made, the event fires.

The event handler receives an argument of type SARowUpdatingEventArgs containing data related to this event.

For more information, see the .NET Framework documentation for OleDbDataAdapter.RowUpdating Event.

1.12 SADataReader Class

A read-only, forward-only result set from a query or stored procedure.

Syntax

Visual Basic

```
Public NotInheritable Class SADataReader Inherits
System.Data.Common.DbDataReader Implements
System.ComponentModel.IListSource
```

C#

```
public sealed class SADataReader : System.Data.Common.DbDataReader,
System.ComponentModel.IListSource
```

Members

All members of SADataReader, including inherited members.

Methods

Modifier and Type	Method	Description
public override void	Close() [page 210]	Closes the SADataReader.
public override bool	GetBoolean(int) [page 210]	Returns the value of the specified column as a Boolean.
public override byte	GetByte(int) [page 211]	Returns the value of the specified column as a Byte.
public override unsafe long	GetBytes(int, long, byte[], int, int) [page 212]	Reads a stream of bytes from the specified column offset into the buffer as an array, starting at the given buffer offset.
public override char	GetChar(int) [page 213]	Returns the value of the specified column as a character.
public override unsafe long	GetChars(int, long, char[], int, int) [page 214]	Reads a stream of characters from the specified column offset into the buffer as an array starting at the given buffer offset.
public new IDataReader	GetData(int) [page 215]	This method is not supported.
public override string	GetDataTypeName(int) [page 215]	Returns the name of the source data type.
public override DateTime	GetDateTime(int) [page 216]	Returns the value of the specified column as a DateTime object.
public DateTimeOffset	GetDateTimeOffset(int) [page 217]	Returns the value of the specified column as a DateTimeOffset object.

Modifier and Type	Method	Description
public override decimal	GetDecimal(int) [page 218]	Returns the value of the specified column as a Decimal object.
public override double	GetDouble(int) [page 219]	Returns the value of the specified column as a double-precision floating-point number.
public override IEnumerator	GetEnumerator() [page 220]	Returns a System.Collections.IEnumerator that iterates through the SADataReader object.
public override Type	GetFieldType(int) [page 220]	Returns the Type that is the data type of the object.
public override float	GetFloat(int) [page 221]	Returns the value of the specified column as a single-precision floating-point number.
public override Guid	GetGuid(int) [page 222]	Returns the value of the specified column as a global unique identifier (GUID).
public override short	GetInt16(int) [page 223]	Returns the value of the specified column as a 16-bit signed integer.
public override int	GetInt32(int) [page 223]	Returns the value of the specified column as a 32-bit signed integer.
public override long	GetInt64(int) [page 224]	Returns the value of the specified column as a 64-bit signed integer.
public override string	GetName(int) [page 225]	Returns the name of the specified column.
public override int	GetOrdinal(string) [page 225]	Returns the column ordinal, given the column name.
public override unsafe DataTable	GetSchemaTable() [page 226]	Returns a DataTable that describes the column metadata of the SADataReader.
public override string	GetString(int) [page 228]	Returns the value of the specified column as a string.
public TimeSpan	GetTimeSpan(int) [page 229]	Returns the value of the specified column as a TimeSpan object.
public ushort	GetUInt16(int) [page 230]	Returns the value of the specified column as a 16-bit unsigned integer.
public uint	GetUInt32(int) [page 231]	Returns the value of the specified column as a 32-bit unsigned integer.
public ulong	GetUInt64(int) [page 231]	Returns the value of the specified column as a 64-bit unsigned integer.
public override object	GetValue [page 232]	Returns the value of the specified column as an Object.
public override unsafe int	GetValues(object[]) [page 234]	Gets all the columns in the current row.
public override bool	IsDBNull(int) [page 235]	Returns a value indicating whether the column contains NULL values.

Modifier and Type	Method	Description
public void	myDispose() [page 236]	Frees the resources associated with the object.
public override bool	NextResult() [page 236]	Advances the SADataReader to the next result set when processing queries that return multiple result sets.
public override unsafe bool	Read() [page 237]	Reads the next row of the result set and moves the SADataReader to that row.

Properties

Modifier and Type	Property	Description
public override int	Depth [page 238]	Gets a value indicating the depth of nesting for the current row.
public override int	FieldCount [page 238]	Gets the number of columns in the result set.
public override bool	HasRows [page 239]	Gets a value that indicates whether the SADataReader contains one or more rows.
public override bool	IsClosed [page 239]	Gets a values that indicates whether the SADataReader is closed.
public override int	RecordsAffected [page 240]	The number of rows changed, inserted, or deleted by the execution of the SQL statement.
public override object	this [page 240]	Returns the value of a column in its native format.

Remarks

There is no constructor for SADataReader. To get an SADataReader object, execute an SACommand:

```
SACommand cmd = new SACommand(
   "SELECT EmployeeID FROM Employees", conn );
SADataReader reader = cmd.ExecuteReader();
```

You can only move forward through an SADataReader. If you need a more flexible object to manipulate results, then use an SADataAdapter.

The SADataReader retrieves rows as needed, whereas the SADataAdapter must retrieve all rows of a result set before you carry out any action on the object. For large result sets, this difference gives the SADataReader a much faster response time.

Implements: IDataReader, IDisposable, IDataRecord, IListSource

For more information, see "Data access and manipulation".

In this section:

Close() Method [page 210]
Closes the SADataReader.

GetBoolean(int) Method [page 210]
Returns the value of the specified column as a Boolean.

GetByte(int) Method [page 211]
Returns the value of the specified column as a Byte.

GetBytes(int, long, byte[], int, int) Method [page 212]
Reads a stream of bytes from the specified column offset into the buffer as an array, starting at the given buffer offset.

GetChar(int) Method [page 213]
Returns the value of the specified column as a character.

GetChars(int, long, char[], int, int) Method [page 214]
Reads a stream of characters from the specified column offset into the buffer as an array starting at the given buffer offset.

GetData(int) Method [page 215]
This method is not supported.

GetDataTypeName(int) Method [page 215]
Returns the name of the source data type.

GetDateTime(int) Method [page 216]
Returns the value of the specified column as a DateTime object.

GetDateTimeOffset(int) Method [page 217]
Returns the value of the specified column as a DateTimeOffset object.

GetDecimal(int) Method [page 218]
Returns the value of the specified column as a Decimal object.

GetDouble(int) Method [page 219]
Returns the value of the specified column as a double-precision floating-point number.

GetEnumerator() Method [page 220]
Returns a System.Collections.IEnumerator that iterates through the SADataReader object.

GetFieldType(int) Method [page 220]
Returns the Type that is the data type of the object.

GetFloat(int) Method [page 221]
Returns the value of the specified column as a single-precision floating-point number.

GetGuid(int) Method [page 222]
Returns the value of the specified column as a global unique identifier (GUID).

GetInt16(int) Method [page 223]
Returns the value of the specified column as a 16-bit signed integer.

GetInt32(int) Method [page 223]
Returns the value of the specified column as a 32-bit signed integer.

GetInt64(int) Method [page 224]
Returns the value of the specified column as a 64-bit signed integer.

GetName(int) Method [page 225]
Returns the name of the specified column.

Related Information

ExecuteReader() Method [page 77]

1.12.1 Close() Method

Closes the SADataReader.

Syntax

Visual Basic

```
Public Overrides Sub Close ()
```

C#

```
public override void Close ()
```

Remarks

Explicitly call the Close method when you are finished using the SADataReader.

When running in autocommit mode, a COMMIT is issued as a side effect of closing the SADataReader.

1.12.2 GetBoolean(int) Method

Returns the value of the specified column as a Boolean.

Syntax

Visual Basic

```
Public Overrides Function GetBoolean (ByVal ordinal As Integer) As Boolean
```

C#

```
public override bool GetBoolean (int ordinal)
```

Parameters

ordinal An ordinal number indicating the column from which the value is obtained. The numbering is zero-based.

Returns

The value of the column.

Remarks

No conversions are performed, so the data that is being retrieved must already be a Boolean.

Related Information

GetOrdinal(string) Method [page 225]
GetFieldType(int) Method [page 220]

1.12.3 GetByte(int) Method

Returns the value of the specified column as a Byte.

Syntax

Visual Basic

```
Public Overrides Function GetByte (ByVal ordinal As Integer) As Byte
```

C#

```
public override byte GetByte (int ordinal)
```

Parameters

ordinal An ordinal number indicating the column from which the value is obtained. The numbering is zero-based.

Returns

The value of the column.

Remarks

No conversions are performed, so the data that is being retrieved must already be a byte.

1.12.4 GetBytes(int, long, byte[], int, int) Method

Reads a stream of bytes from the specified column offset into the buffer as an array, starting at the given buffer offset.

Syntax

Visual Basic

```
Public Overrides Function GetBytes (
    ByVal ordinal As Integer,
    ByVal dataIndex As Long,
    ByVal buffer As Byte(),
    ByVal bufferIndex As Integer,
    ByVal length As Integer
) As Long
```

C#

```
public override unsafe long GetBytes (
    int ordinal,
    long dataIndex,
    byte[] buffer,
    int bufferIndex,
    int length
)
```

Parameters

ordinal An ordinal number indicating the column from which the value is obtained. The numbering is zero-based.
dataIndex The index within the column value from which to read bytes.
buffer An array in which to store the data.
bufferIndex The index in the array to start copying data.
length The maximum length to copy into the specified buffer.

Returns

The number of bytes read.

Remarks

GetBytes returns the number of available bytes in the field. In most cases this is the exact length of the field. However, the number returned may be less than the true length of the field if GetBytes has already been used to obtain bytes from the field. This may be the case, for example, when the SADataReader is reading a large data structure into a buffer.

If you pass a buffer that is a null reference (Nothing in Visual Basic), then GetBytes returns the length of the field in bytes.

No conversions are performed, so the data that is being retrieved must already be a byte array.

1.12.5 GetChar(int) Method

Returns the value of the specified column as a character.

Syntax

Visual Basic

```
Public Overrides Function GetChar (ByVal ordinal As Integer) As Char
```

C#

```
public override char GetChar (int ordinal)
```

Parameters

ordinal An ordinal number indicating the column from which the value is obtained. The numbering is zero-based.

Returns

The value of the column.

Remarks

No conversions are performed, so the data that is being retrieved must already be a character.

Call the SADataReader.IsDBNull method to check for null values before calling this method.

Related Information

IsDBNull(int) Method [page 235]
IsDBNull(int) Method [page 235]

1.12.6 GetChars(int, long, char[], int, int) Method

Reads a stream of characters from the specified column offset into the buffer as an array starting at the given buffer offset.

Syntax

Visual Basic

```
Public Overrides Function GetChars (
    ByVal ordinal As Integer,
    ByVal dataIndex As Long,
    ByVal buffer As Char(),
    ByVal bufferIndex As Integer,
    ByVal length As Integer
) As Long
```

C#

```
public override unsafe long GetChars (
    int ordinal,
    long dataIndex,
    char[] buffer,
    int bufferIndex,
    int length
)
```

Parameters

ordinal The zero-based column ordinal.
dataIndex The index within the row from which to begin the read operation.
buffer The buffer into which to copy data.
bufferIndex The index for buffer to begin the read operation.
length The number of characters to read.

Returns

The actual number of characters read.

Remarks

GetChars returns the number of available characters in the field. In most cases this is the exact length of the field. However, the number returned may be less than the true length of the field if GetChars has already been used to obtain characters from the field. This may be the case, for example, when the SADataReader is reading a large data structure into a buffer.

If you pass a buffer that is a null reference (Nothing in Visual Basic), then GetChars returns the length of the field in characters.

No conversions are performed, so the data that is being retrieved must already be a character array.

For information about handling BLOBs, see "Data access and manipulation".

1.12.7 GetData(int) Method

This method is not supported.

Syntax

Visual Basic

```
Public Shadows Function GetData (ByVal i As Integer) As IDataReader
```

C#

```
public new IDataReader GetData (int i)
```

Remarks

When called, it throws an InvalidOperationException.

1.12.8 GetDataTypeName(int) Method

Returns the name of the source data type.

Syntax

Visual Basic

```
Public Overrides Function GetDataTypeName (ByVal index As Integer) As
String
```

C#

```
public override string GetDataTypeName (int index)
```

Parameters

index The zero-based column ordinal.

Returns

The name of the back-end data type.

1.12.9 GetDateTime(int) Method

Returns the value of the specified column as a DateTime object.

Syntax

Visual Basic

```
Public Overrides Function GetDateTime (ByVal ordinal As Integer) As Date
```

C#

```
public override DateTime GetDateTime (int ordinal)
```

Parameters

ordinal The zero-based column ordinal.

Returns

The value of the specified column.

Remarks

No conversions are performed, so the data that is being retrieved must already be a DateTime object.

Call the SADataReader.IsDBNull method to check for null values before calling this method.

Related Information

IsDBNull(int) Method [page 235]

1.12.10 GetDateTimeOffset(int) Method

Returns the value of the specified column as a DateTimeOffset object.

Syntax

Visual Basic

```
Public Function GetDateTimeOffset (ByVal ordinal As Integer) As
DateTimeOffset
```

C#

```
public DateTimeOffset GetDateTimeOffset (int ordinal)
```

Parameters

ordinal The zero-based column ordinal.

Returns

The value of the specified column.

Remarks

No conversions are performed, so the data that is being retrieved must already be a DateTimeOffset object.

Call the SADataReader.IsDBNull method to check for null values before calling this method.

Related Information

IsDBNull(int) Method [page 235]

1.12.11 GetDecimal(int) Method

Returns the value of the specified column as a Decimal object.

Syntax

Visual Basic

```
Public Overrides Function GetDecimal (ByVal ordinal As Integer) As Decimal
```

C#

```
public override decimal GetDecimal (int ordinal)
```

Parameters

ordinal An ordinal number indicating the column from which the value is obtained. The numbering is zero-based.

Returns

The value of the specified column.

Remarks

No conversions are performed, so the data that is being retrieved must already be a Decimal object.

Call the SADataReader.IsDBNull method to check for null values before calling this method.

Related Information

IsDBNull(int) Method [page 235]

1.12.12 GetDouble(int) Method

Returns the value of the specified column as a double-precision floating-point number.

Syntax

Visual Basic

```
Public Overrides Function GetDouble (ByVal ordinal As Integer) As Double
```

C#

```
public override double GetDouble (int ordinal)
```

Parameters

ordinal An ordinal number indicating the column from which the value is obtained. The numbering is zero-based.

Returns

The value of the specified column.

Remarks

No conversions are performed, so the data that is being retrieved must already be a double-precision floating-point number.

Call the SADataReader.IsDBNull method to check for null values before calling this method.

Related Information

IsDBNull(int) Method [page 235]

1.12.13 GetEnumerator() Method

Returns a System.Collections.IEnumerator that iterates through the SADataReader object.

Syntax

Visual Basic

```
Public Overrides Function GetEnumerator () As
System.Collections.IEnumerator
```

C#

```
public override IEnumerator GetEnumerator ()
```

Returns

A System.Collections.IEnumerator for the SADataReader object.

Related Information

SADataReader Class [page 205]

1.12.14 GetFieldType(int) Method

Returns the Type that is the data type of the object.

Syntax

Visual Basic

```
Public Overrides Function GetFieldType (ByVal index As Integer) As Type
```

C#

```
public override Type GetFieldType (int index)
```

Parameters

index The zero-based column ordinal.

Returns

The type that is the data type of the object.

1.12.15 GetFloat(int) Method

Returns the value of the specified column as a single-precision floating-point number.

Syntax

Visual Basic

```
Public Overrides Function GetFloat (ByVal ordinal As Integer) As Single
```

C#

```
public override float GetFloat (int ordinal)
```

Parameters

ordinal An ordinal number indicating the column from which the value is obtained. The numbering is zero-based.

Returns

The value of the specified column.

Remarks

No conversions are performed, so the data that is being retrieved must already be a single-precision floating-point number.

Call the SADataReader.IsDBNull method to check for null values before calling this method.

Related Information

IsDBNull(int) Method [page 235]

1.12.16 GetGuid(int) Method

Returns the value of the specified column as a global unique identifier (GUID).

Syntax

Visual Basic

```
Public Overrides Function GetGuid (ByVal ordinal As Integer) As Guid
```

C#

```
public override Guid GetGuid (int ordinal)
```

Parameters

ordinal An ordinal number indicating the column from which the value is obtained. The numbering is zero-based.

Returns

The value of the specified column.

Remarks

The data that is being retrieved must already be a globally-unique identifier or binary(16).

Call the SADataReader.IsDBNull method to check for null values before calling this method.

Related Information

IsDBNull(int) Method [page 235]

1.12.17 GetInt16(int) Method

Returns the value of the specified column as a 16-bit signed integer.

Syntax

Visual Basic

```
Public Overrides Function GetInt16 (ByVal ordinal As Integer) As Short
```

C#

```
public override short GetInt16 (int ordinal)
```

Parameters

ordinal An ordinal number indicating the column from which the value is obtained. The numbering is zero-based.

Returns

The value of the specified column.

Remarks

No conversions are performed, so the data that is being retrieved must already be a 16-bit signed integer.

1.12.18 GetInt32(int) Method

Returns the value of the specified column as a 32-bit signed integer.

Syntax

Visual Basic

```
Public Overrides Function GetInt32 (ByVal ordinal As Integer) As Integer
```

C#

```
public override int GetInt32 (int ordinal)
```

Parameters

ordinal An ordinal number indicating the column from which the value is obtained. The numbering is zero-based.

Returns

The value of the specified column.

Remarks

No conversions are performed, so the data that is being retrieved must already be a 32-bit signed integer.

1.12.19 GetInt64(int) Method

Returns the value of the specified column as a 64-bit signed integer.

Syntax

Visual Basic

```
Public Overrides Function GetInt64 (ByVal ordinal As Integer) As Long
```

C#

```
public override long GetInt64 (int ordinal)
```

Parameters

ordinal An ordinal number indicating the column from which the value is obtained. The numbering is zero-based.

Returns

The value of the specified column.

Remarks

No conversions are performed, so the data that is being retrieved must already be a 64-bit signed integer.

1.12.20 GetName(int) Method

Returns the name of the specified column.

Syntax

Visual Basic

```
Public Overrides Function GetName (ByVal index As Integer) As String
```

C#

```
public override string GetName (int index)
```

Parameters

index The zero-based index of the column.

Returns

The name of the specified column.

1.12.21 GetOrdinal(string) Method

Returns the column ordinal, given the column name.

Syntax

Visual Basic

```
Public Overrides Function GetOrdinal (ByVal name As String) As Integer
```

C#

```
public override int GetOrdinal (string name)
```

Parameters

name The column name.

Returns

The zero-based column ordinal.

Remarks

GetOrdinal performs a case-sensitive lookup first. If it fails, then a second case-insensitive search is made.

GetOrdinal is Japanese kana-width insensitive.

Because ordinal-based lookups are more efficient than named lookups, it is inefficient to call GetOrdinal within a loop. Save time by calling GetOrdinal once and assigning the results to an integer variable for use within the loop.

1.12.22 GetSchemaTable() Method

Returns a DataTable that describes the column metadata of the SADataReader.

Syntax

Visual Basic

```
Public Overrides Function GetSchemaTable () As DataTable
```

C#

```
public override unsafe DataTable GetSchemaTable ()
```

Returns

A DataTable that describes the column metadata.

Remarks

This method returns metadata about each column in the following order:

DataTable column	Description
ColumnName	The name of the column or a null reference (Nothing in Visual Basic) if the column has no name. If the column is aliased in the SQL query, then the alias is returned. In result sets, not all columns have names and not all column names are unique.
ColumnOrdinal	The ID of the column. The value is in the range [0, FieldCount -1].
ColumnSize	For sized columns, the maximum length of a value in the column. For other columns, this is the size in bytes of the data type.
NumericPrecision	The precision of a numeric column or DBNull if the column is not numeric.
NumericScale	The scale of a numeric column or DBNull if the column is not numeric.
IsUnique	True if the column is a non-computed unique column in the table (BaseTableName) it is taken from.
IsKey	True if the column is one of a set of columns in the result set that taken together from a unique key for the result set. The set of columns with IsKey set to true does not need to be the minimal set that uniquely identifies a row in the result set.
BaseServerName	The name of the database server used by the SADataReader.
BaseCatalogName	The name of the catalog in the database that contains the column. This value is always DBNull.
BaseColumnName	The original name of the column in the table BaseTableName of the database or DBNull if the column is computed or if this information cannot be determined.
BaseSchemaName	The name of the schema in the database that contains the column.
BaseTableName	The name of the table in the database that contains the column, or DBNull if column is computed or if this information cannot be determined.
DataType	The .NET data type that is most appropriate for this type of column.
AllowDBNull	True if the column is nullable; false if the column is not nullable or if this information cannot be determined.
ProviderType	The type of the column.
IsAliased	True if the column name is an alias; false if it is not an alias.
IsExpression	True if the column is an expression; false if it is a column value.

DataTable column	Description
IsIdentity	True if the column is an identity column; false if it is not an identity column.
IsAutoIncrement	True if the column is an autoincrement or global autoincrement column; false otherwise (or if this information cannot be determined).
IsRowVersion	True if the column contains a persistent row identifier that cannot be written to, and has no meaningful value except to identify the row.
IsHidden	True if the column is hidden; false otherwise.
IsLong	True if the column is a long varchar, long nvarchar, or a long binary column; false otherwise.
IsReadOnly	True if the column is read-only; false if the column is modifiable or if its access cannot be determined.

For more information about these columns, see the .NET Framework documentation for SqlDataReader.GetSchemaTable.

For more information, see "Data access and manipulation".

1.12.23 GetString(int) Method

Returns the value of the specified column as a string.

Syntax

Visual Basic

```
Public Overrides Function GetString (ByVal ordinal As Integer) As String
```

C#

```
public override string GetString (int ordinal)
```

Parameters

ordinal An ordinal number indicating the column from which the value is obtained. The numbering is zero-based.

Returns

The value of the specified column.

Remarks

No conversions are performed, so the data that is being retrieved must already be a string.

Call the SADataReader.IsDBNull method to check for NULL values before calling this method.

Related Information

IsDBNull(int) Method [page 235]

1.12.24 GetTimeSpan(int) Method

Returns the value of the specified column as a TimeSpan object.

Syntax

Visual Basic

```
Public Function GetTimeSpan (ByVal ordinal As Integer) As TimeSpan
```

C#

```
public TimeSpan GetTimeSpan (int ordinal)
```

Parameters

ordinal An ordinal number indicating the column from which the value is obtained. The numbering is zero-based.

Returns

The value of the specified column.

Remarks

The column must be a TIME data type. The data is converted to TimeSpan. The Days property of TimeSpan is always set to 0.

Call SADataReader.IsDBNull method to check for NULL values before calling this method.

For more information, see "Data access and manipulation".

Related Information

IsDBNull(int) Method [page 235]

1.12.25 GetUInt16(int) Method

Returns the value of the specified column as a 16-bit unsigned integer.

Syntax

Visual Basic

```
Public Function GetUInt16 (ByVal ordinal As Integer) As UShort
```

C#

```
public ushort GetUInt16 (int ordinal)
```

Parameters

ordinal An ordinal number indicating the column from which the value is obtained. The numbering is zero-based.

Returns

The value of the specified column.

Remarks

No conversions are performed, so the data that is being retrieved must already be a 16-bit unsigned integer.

1.12.26 GetUInt32(int) Method

Returns the value of the specified column as a 32-bit unsigned integer.

Syntax

Visual Basic

```
Public Function GetUInt32 (ByVal ordinal As Integer) As UInteger
```

C#

```
public uint GetUInt32 (int ordinal)
```

Parameters

ordinal An ordinal number indicating the column from which the value is obtained. The numbering is zero-based.

Returns

The value of the specified column.

Remarks

No conversions are performed, so the data that is being retrieved must already be a 32-bit unsigned integer.

1.12.27 GetUInt64(int) Method

Returns the value of the specified column as a 64-bit unsigned integer.

Syntax

Visual Basic

```
Public Function GetUInt64 (ByVal ordinal As Integer) As ULong
```

C#

```
public ulong GetUInt64 (int ordinal)
```

Parameters

ordinal An ordinal number indicating the column from which the value is obtained. The numbering is zero-based.

Returns

The value of the specified column.

Remarks

No conversions are performed, so the data that is being retrieved must already be a 64-bit unsigned integer.

1.12.28 GetValue Method

Returns the value of the specified column as an Object.

Overload list

Modifier and Type	Overload name	Description
public override object	GetValue(int) [page 233]	Returns the value of the specified column as an Object.
public object	GetValue(int, long, int) [page 233]	Returns a substring of the value of the specified column as an Object.

In this section:

GetValue(int) Method [page 233]
Returns the value of the specified column as an Object.

GetValue(int, long, int) Method [page 233]
Returns a substring of the value of the specified column as an Object.

1.12.28.1 GetValue(int) Method

Returns the value of the specified column as an Object.

Syntax

Visual Basic

```
Public Overrides Function GetValue (ByVal ordinal As Integer) As Object
```

C#

```
public override object GetValue (int ordinal)
```

Parameters

ordinal An ordinal number indicating the column from which the value is obtained. The numbering is zero-based.

Returns

The value of the specified column as an object.

Remarks

This method returns DBNull for NULL database columns.

1.12.28.2 GetValue(int, long, int) Method

Returns a substring of the value of the specified column as an Object.

Syntax

Visual Basic

```
Public Function GetValue (
    ByVal ordinal As Integer,
    ByVal index As Long,
    ByVal length As Integer
) As Object
```

C#

```
public object GetValue (
```

```
    int ordinal,
    long index,
    int length
)
```

Parameters

ordinal An ordinal number indicating the column from which the value is obtained. The numbering is zero-based.

index A zero-based index of the substring of the value to be obtained.

length The length of the substring of the value to be obtained.

Returns

The substring value is returned as an object.

Remarks

This method returns DBNull for NULL database columns.

1.12.29 GetValues(object[]) Method

Gets all the columns in the current row.

Syntax

Visual Basic

```
Public Overrides Function GetValues (ByVal values As Object()) As Integer
```

C#

```
public override unsafe int GetValues (object[] values)
```

Parameters

values An array of objects that holds an entire row of the result set.

Returns

The number of objects in the array.

Remarks

For most applications, the GetValues method provides an efficient means for retrieving all columns, rather than retrieving each column individually.

You can pass an Object array that contains fewer than the number of columns contained in the resulting row. Only the amount of data the Object array holds is copied to the array. You can also pass an Object array whose length is more than the number of columns contained in the resulting row.

This method returns DBNull for NULL database columns.

1.12.30 IsDBNull(int) Method

Returns a value indicating whether the column contains NULL values.

Syntax

Visual Basic

```
Public Overrides Function IsDBNull (ByVal ordinal As Integer) As Boolean
```

C#

```
public override bool IsDBNull (int ordinal)
```

Parameters

ordinal The zero-based column ordinal.

Returns

True if the specified column value is equivalent to DBNull; false otherwise.

Remarks

Call this method to check for NULL column values before calling the typed get methods (for example, GetByte, GetChar, and so on) to avoid raising an exception.

1.12.31 myDispose() Method

Frees the resources associated with the object.

Syntax

Visual Basic

```
Public Sub myDispose ()
```

C#

```
public void myDispose ()
```

1.12.32 NextResult() Method

Advances the SADataReader to the next result set when processing queries that return multiple result sets.

Syntax

Visual Basic

```
Public Overrides Function NextResult () As Boolean
```

C#

```
public override bool NextResult ()
```

Returns

True if there are more result sets; false otherwise.

Remarks

Used to process multiple result sets, which can be generated by executing batch SQL statements or stored procedures.

By default, the data reader is positioned on the first result set.

1.12.33 Read() Method

Reads the next row of the result set and moves the SADataReader to that row.

Syntax

Visual Basic

```
Public Overrides Function Read () As Boolean
```

C#

```
public override unsafe bool Read ()
```

Returns

True if there are more rows; false otherwise.

Remarks

The default position of the SADataReader is prior to the first record. Call Read to begin accessing any data.

Example

The following code fills a listbox with the values in a single column of results.

```
while( reader.Read() ) {
   listResults.Items.Add( reader.GetValue( 0 ).ToString() );
}
listResults.EndUpdate();
reader.Close();
```

1.12.34 Depth Property

Gets a value indicating the depth of nesting for the current row.

Syntax

Visual Basic

```
Public ReadOnly Overrides Property Depth As Integer
```

C#

```
public override int Depth {get;}
```

Remarks

The outermost table has a depth of zero.

The depth of nesting for the current row.

1.12.35 FieldCount Property

Gets the number of columns in the result set.

Syntax

Visual Basic

```
Public ReadOnly Overrides Property FieldCount As Integer
```

C#

```
public override int FieldCount {get;}
```

Remarks

The number of columns in the current record.

1.12.36 HasRows Property

Gets a value that indicates whether the SADataReader contains one or more rows.

Syntax

Visual Basic

```
Public ReadOnly Overrides Property HasRows As Boolean
```

C#

```
public override bool HasRows {get;}
```

Remarks

True if the SADataReader contains one or more rows; false otherwise.

1.12.37 IsClosed Property

Gets a values that indicates whether the SADataReader is closed.

Syntax

Visual Basic

```
Public ReadOnly Overrides Property IsClosed As Boolean
```

C#

```
public override bool IsClosed {get;}
```

Remarks

True if the SADataReader is closed; false otherwise.

IsClosed and RecordsAffected are the only properties that you can use after the SADataReader is closed.

1.12.38 RecordsAffected Property

The number of rows changed, inserted, or deleted by the execution of the SQL statement.

Syntax

Visual Basic

```
Public ReadOnly Overrides Property RecordsAffected As Integer
```

C#

```
public override int RecordsAffected {get;}
```

Remarks

The number of rows changed, inserted, or deleted. This value is 0 if no rows were affected or the statement failed, and -1 for SELECT statements.

The value of this property is cumulative. For example, if two records are inserted in batch mode, then the value of RecordsAffected is 2.

IsClosed and RecordsAffected are the only properties that you can use after the SADataReader is closed.

1.12.39 this Property

Returns the value of a column in its native format.

Overload list

Modifier and Type	Overload name	Description
public override object	this[int index] [page 241]	Returns the value of a column in its native format.
public override object	this[string name] [page 241]	Returns the value of a column in its native format.

In this section:

this[int index] Property [page 241]
Returns the value of a column in its native format.

this[string name] Property [page 241]
Returns the value of a column in its native format.

1.12.39.1 this[int index] Property

Returns the value of a column in its native format.

Syntax

Visual Basic

```
Public ReadOnly Overrides Property Item (ByVal index As Integer) As Object
```

C#

```
public override object this[int index] {get;}
```

Remarks

In C#, this property is the indexer for the SADataReader class.

1.12.39.2 this[string name] Property

Returns the value of a column in its native format.

Syntax

Visual Basic

```
Public ReadOnly Overrides Property Item (ByVal name As String) As Object
```

C#

```
public override object this[string name] {get;}
```

Remarks

In C#, this property is the indexer for the SADataReader class.

1.13 SADataSourceEnumerator Class

Provides a mechanism for enumerating all available instances of database servers within the local network.

Syntax

Visual Basic

```
Public NotInheritable Class SADataSourceEnumerator Inherits
System.Data.Common.DbDataSourceEnumerator
```

C#

```
public sealed class SADataSourceEnumerator :
System.Data.Common.DbDataSourceEnumerator
```

Members

All members of SADataSourceEnumerator, including inherited members.

Methods

Modifier and Type	Method	Description
public unsafe override DataTable	GetDataSources() [page 243]	Retrieves a DataTable containing information about all visible database servers.

Properties

Modifier and Type	Property	Description
public SADataSourceEnumerator	Instance [page 243]	Gets an instance of SADataSourceEnumerator, which can be used to retrieve information about all visible database servers.

Remarks

There is no constructor for SADataSourceEnumerator.

In this section:

GetDataSources() Method [page 243]
Retrieves a DataTable containing information about all visible database servers.

Instance Property [page 243]
Gets an instance of SADataSourceEnumerator, which can be used to retrieve information about all visible database servers.

1.13.1 GetDataSources() Method

Retrieves a DataTable containing information about all visible database servers.

Syntax

Visual Basic

```
Public Overrides Function GetDataSources () As DataTable
```

C#

```
public unsafe override DataTable GetDataSources ()
```

Remarks

The returned table has four columns: ServerName, IPAddress, PortNumber, and DataBaseNames. There is a row in the table for each available database server.

Example

The following code fills a DataTable with information for each database server that is available.

```
DataTable servers = SADataSourceEnumerator.Instance.GetDataSources();
```

1.13.2 Instance Property

Gets an instance of SADataSourceEnumerator, which can be used to retrieve information about all visible database servers.

Syntax

Visual Basic

```
Public Shared ReadOnly Property Instance As SADataSourceEnumerator
```

C#

```
public SADataSourceEnumerator Instance {get;}
```

1.14 SADefault Class

Represents a parameter with a default value.

Syntax

Visual Basic

```
Public NotInheritable Class SADefault
```

C#

```
public sealed class SADefault
```

Members

All members of SADefault, including inherited members.

Variables

Modifier and Type	Variable	Description
public static readonly SADefault	Value	Gets the value for a default parameter. This field is read-only and static.

Remarks

There is no constructor for SADefault.

```
SAParameter parm = new SAParameter();
parm.Value = SADefault.Value;
```

1.15 SAError Class

Collects information relevant to a warning or error returned by the data source.

Syntax

Visual Basic

```
Public NotInheritable Class SAError
```

C#

```
public sealed class SAError
```

Members

All members of SAError, including inherited members.

Methods

Modifier and Type	Method	Description
public override string	ToString() [page 246]	The complete text of the error message.

Properties

Modifier and Type	Property	Description
public string	Message [page 246]	Returns a short description of the error.
public int	NativeError [page 246]	Returns database-specific error information.
public string	Source [page 247]	Returns the name of the provider that generated the error.
public string	SqlState [page 247]	The five-character SQLSTATE following the ANSI SQL standard.

Remarks

There is no constructor for SAError.

For information about error handling, see ".NET error handling".

In this section:

ToString() Method [page 246]
The complete text of the error message.

Message Property [page 246]
Returns a short description of the error.

NativeError Property [page 246]
Returns database-specific error information.

Source Property [page 247]
Returns the name of the provider that generated the error.

SqlState Property [page 247]
The five-character SQLSTATE following the ANSI SQL standard.

1.15.1 ToString() Method

The complete text of the error message.

Syntax

Visual Basic

```
Public Overrides Function ToString () As String
```

C#

```
public override string ToString ()
```

Example

The return value is a string is in the form *SAError:*, followed by the message. For example:

```
SAError:UserId or Password not valid.
```

1.15.2 Message Property

Returns a short description of the error.

Syntax

Visual Basic

```
Public ReadOnly Property Message As String
```

C#

```
public string Message {get;}
```

1.15.3 NativeError Property

Returns database-specific error information.

Syntax

Visual Basic

```
Public ReadOnly Property NativeError As Integer
```

C#

```
public int NativeError {get;}
```

1.15.4 Source Property

Returns the name of the provider that generated the error.

Syntax

Visual Basic

```
Public ReadOnly Property Source As String
```

C#

```
public string Source {get;}
```

1.15.5 SqlState Property

The five-character SQLSTATE following the ANSI SQL standard.

Syntax

Visual Basic

```
Public ReadOnly Property SqlState As String
```

C#

```
public string SqlState {get;}
```

1.16 SAErrorCollection Class

Collects all errors generated by the .NET Data Provider.

Syntax

Visual Basic

```
Public NotInheritable Class SAErrorCollection Implements
System.Collections.ICollection, System.Collections.IEnumerable
```

C#

```
public sealed class SAErrorCollection : System.Collections.ICollection,
System.Collections.IEnumerable
```

Members

All members of SAErrorCollection, including inherited members.

Methods

Modifier and Type	Method	Description
public void	CopyTo(Array, int) [page 249]	Copies the elements of the SAErrorCollection into an array, starting at the given index within the array.
public IEnumerator	GetEnumerator() [page 249]	Returns an enumerator that iterates through the SAErrorCollection.

Properties

Modifier and Type	Property	Description
public int	Count [page 250]	Returns the number of errors in the collection.
public SAError	this[int index] [page 250]	Returns the error at the specified index.

Remarks

There is no constructor for SAErrorCollection. Typically, an SAErrorCollection is obtained from the SAException.Errors property.

Implements: ICollection, IEnumerable

For information about error handling, see ".NET error handling".

In this section:

CopyTo(Array, int) Method [page 249]
: Copies the elements of the SAErrorCollection into an array, starting at the given index within the array.

GetEnumerator() Method [page 249]
: Returns an enumerator that iterates through the SAErrorCollection.

Count Property [page 250]
: Returns the number of errors in the collection.

this[int index] Property [page 250]
: Returns the error at the specified index.

Related Information

Errors Property [page 253]

1.16.1 CopyTo(Array, int) Method

Copies the elements of the SAErrorCollection into an array, starting at the given index within the array.

Syntax

Visual Basic

```
Public Sub CopyTo (
    ByVal array As Array,
    ByVal index As Integer
)
```

C#

```
public void CopyTo (
    Array array,
    int index
)
```

Parameters

array The array into which to copy the elements.
index The starting index of the array.

1.16.2 GetEnumerator() Method

Returns an enumerator that iterates through the SAErrorCollection.

Syntax

Visual Basic

```
Public Function GetEnumerator () As System.Collections.IEnumerator
```

C#

```
public IEnumerator GetEnumerator ()
```

Returns

A System.Collections.IEnumerator for the SAErrorCollection.

1.16.3 Count Property

Returns the number of errors in the collection.

Syntax

Visual Basic

```
Public ReadOnly Property Count As Integer
```

C#

```
public int Count {get;}
```

1.16.4 this[int index] Property

Returns the error at the specified index.

Syntax

Visual Basic

```
Public ReadOnly Property Item (ByVal index As Integer) As SAError
```

C#

```
public SAError this[int index] {get;}
```

Remarks

An SAError object that contains the error at the specified index.

Related Information

SAError Class [page 244]

1.17 SAException Class

The exception that is thrown when the database server returns a warning or error.

Syntax

Visual Basic

```
Public Class SAException Inherits System.Exception
```

C#

```
public class SAException : System.Exception
```

Members

All members of SAException, including inherited members.

Methods

Modifier and Type	Method	Description
public override void	GetObjectData(SerializationInfo, StreamingContext) [page 252]	Sets the SerializationInfo with information about the exception.

Properties

Modifier and Type	Property	Description
public SAErrorCollection	Errors [page 253]	Returns a collection of one or more SAError objects.
public override string	Message [page 253]	Returns the text describing the error.
public int	NativeError [page 254]	Returns database-specific error information.
public override string	Source [page 254]	Returns the name of the provider that generated the error.

Remarks

There is no constructor for SAException. Typically, an SAException object is declared in a catch. For example:

```
...
catch( SAException ex )
{
   MessageBox.Show( ex.Errors[0].Message, "Error" );
}
```

For information about error handling, see ".NET error handling".

In this section:

1.17.1 GetObjectData(SerializationInfo, StreamingContext) Method

Sets the SerializationInfo with information about the exception.

Syntax

Visual Basic

```
Public Overrides Sub GetObjectData (
    ByVal info As SerializationInfo,
    ByVal context As StreamingContext
)
```

C#

```
public override void GetObjectData (
    SerializationInfo info,
    StreamingContext context
)
```

Parameters

info The SerializationInfo that holds the serialized object data about the exception being thrown.
context The StreamingContext that contains contextual information about the source or destination.

Remarks

Overrides Exception.GetObjectData.

1.17.2 Errors Property

Returns a collection of one or more SAError objects.

Syntax

Visual Basic

```
Public ReadOnly Property Errors As SAErrorCollection
```

C#

```
public SAErrorCollection Errors {get;}
```

Remarks

The SAErrorCollection object always contains at least one instance of the SAError object.

Related Information

SAErrorCollection Class [page 247]
SAError Class [page 244]

1.17.3 Message Property

Returns the text describing the error.

Syntax

Visual Basic

```
Public ReadOnly Overrides Property Message As String
```

C#

```
public override string Message {get;}
```

Remarks

This method returns a single string that contains a concatenation of all of the Message properties of all of the SAError objects in the Errors collection. Each message, except the last one, is followed by a carriage return.

Related Information

SAError Class [page 244]

1.17.4 NativeError Property

Returns database-specific error information.

Syntax

Visual Basic

```
Public ReadOnly Property NativeError As Integer
```

C#

```
public int NativeError {get;}
```

1.17.5 Source Property

Returns the name of the provider that generated the error.

Syntax

Visual Basic

```
Public ReadOnly Overrides Property Source As String
```

C#

```
public override string Source {get;}
```

1.18 SAFactory Class

Represents a set of methods for creating instances of the Sap.Data.SQLAnywhere provider's implementation of the data source classes.

Syntax

Visual Basic

```
Public NotInheritable Class SAFactory Inherits
System.Data.Common.DbProviderFactory
```

C#

```
public sealed class SAFactory : System.Data.Common.DbProviderFactory
```

Members

All members of SAFactory, including inherited members.

Variables

Modifier and Type	Variable	Description
public static readonly SAFactory	Instance	Represents the singleton instance of the SAFactory class. SAFactory is a singleton class, which means only this instance of this class can exist. Normally you would not use this field directly. Instead, you get a reference to this instance of SAFactory using System.Data.Common.DbProviderFactories.GetFactory(String). For an example, see the SAFactory description.

Methods

Modifier and Type	Method	Description
public override DbCommand	CreateCommand() [page 257]	Returns a strongly typed System.Data.Common.DbCommand instance.
public override DbCommandBuilder	CreateCommandBuilder() [page 257]	Returns a strongly typed System.Data.Common.DbCommandBuilder instance.
public override DbConnection	CreateConnection() [page 258]	Returns a strongly typed System.Data.Common.DbConnection instance.
public override DbConnectionStringBuilder	CreateConnectionStringBuilder() [page 259]	Returns a strongly typed System.Data.Common.DbConnectionStringBuilder instance.
public override DbDataAdapter	CreateDataAdapter() [page 259]	Returns a strongly typed System.Data.Common.DbDataAdapter instance.
public override DbDataSourceEnumerator	CreateDataSourceEnumerator() [page 260]	Returns a strongly typed System.Data.Common.DbDataSourceEnumerator instance.

Modifier and Type	Method	Description
public override DbParameter	CreateParameter() [page 260]	Returns a strongly typed System.Data.Common.DbParameter instance.
public override CodeAccessPermission	CreatePermission(PermissionState) [page 261]	Returns a strongly-typed CodeAccessPermission instance.

Properties

Modifier and Type	Property	Description
public override bool	CanCreateDataSourceEnumerator [page 262]	Always returns true, which indicates that an SADataSourceEnumerator object can be created.

Remarks

There is no constructor for SAFactory.

DbProviderFactories and DbProviderFactory make provider independent code easier to write. Specify Sap.Data.SQLAnywhere as the provider invariant name passed to GetFactory. For example:

```
' Visual Basic
Dim factory As DbProviderFactory = _
 DbProviderFactories.GetFactory( "Sap.Data.SQLAnywhere" )
Dim conn As DbConnection = _
 factory.CreateConnection()
// C#
DbProviderFactory factory =
        DbProviderFactories.GetFactory("Sap.Data.SQLAnywhere" );
DbConnection conn = factory.CreateConnection();
```

In this example, conn is created as an SAConnection object.

For an explanation of provider factories and generic programming in ADO.NET, see Generic Coding with the ADO.NET 2.0 Base Classes and Factories .

In this section:

CreateCommand() Method [page 257]
Returns a strongly typed System.Data.Common.DbCommand instance.

CreateCommandBuilder() Method [page 257]
Returns a strongly typed System.Data.Common.DbCommandBuilder instance.

CreateConnection() Method [page 258]
Returns a strongly typed System.Data.Common.DbConnection instance.

CreateConnectionStringBuilder() Method [page 259]
Returns a strongly typed System.Data.Common.DbConnectionStringBuilder instance.

CreateDataAdapter() Method [page 259]
Returns a strongly typed System.Data.Common.DbDataAdapter instance.

CreateDataSourceEnumerator() Method [page 260]

Returns a strongly typed System.Data.Common.DbDataSourceEnumerator instance.

CreateParameter() Method [page 260]
Returns a strongly typed System.Data.Common.DbParameter instance.

CreatePermission(PermissionState) Method [page 261]
Returns a strongly-typed CodeAccessPermission instance.

CanCreateDataSourceEnumerator Property [page 262]
Always returns true, which indicates that an SADataSourceEnumerator object can be created.

1.18.1 CreateCommand() Method

Returns a strongly typed System.Data.Common.DbCommand instance.

Syntax

Visual Basic

```
Public Overrides Function CreateCommand () As DbCommand
```

C#

```
public override DbCommand CreateCommand ()
```

Returns

A new SACommand object typed as DbCommand.

Related Information

SACommand Class [page 53]

1.18.2 CreateCommandBuilder() Method

Returns a strongly typed System.Data.Common.DbCommandBuilder instance.

Syntax

Visual Basic

```
Public Overrides Function CreateCommandBuilder () As DbCommandBuilder
```

C#

```
public override DbCommandBuilder CreateCommandBuilder ()
```

Returns

A new SACommandBuilder object typed as DbCommandBuilder.

Related Information

SACommandBuilder Class [page 91]

1.18.3 CreateConnection() Method

Returns a strongly typed System.Data.Common.DbConnection instance.

Syntax

Visual Basic

```
Public Overrides Function CreateConnection () As DbConnection
```

C#

```
public override DbConnection CreateConnection ()
```

Returns

A new SAConnection object typed as DbConnection.

Related Information

SAConnection Class [page 118]

1.18.4 CreateConnectionStringBuilder() Method

Returns a strongly typed System.Data.Common.DbConnectionStringBuilder instance.

Syntax

Visual Basic

```
Public Overrides Function CreateConnectionStringBuilder () As
DbConnectionStringBuilder
```

C#

```
public override DbConnectionStringBuilder CreateConnectionStringBuilder ()
```

Returns

A new SAConnectionStringBuilder object typed as DbConnectionStringBuilder.

Related Information

SAConnectionStringBuilder Class [page 147]

1.18.5 CreateDataAdapter() Method

Returns a strongly typed System.Data.Common.DbDataAdapter instance.

Syntax

Visual Basic

```
Public Overrides Function CreateDataAdapter () As DbDataAdapter
```

C#

```
public override DbDataAdapter CreateDataAdapter ()
```

Returns

A new SADataAdapter object typed as DbDataAdapter.

Related Information

SADataAdapter Class [page 181]

1.18.6 CreateDataSourceEnumerator() Method

Returns a strongly typed System.Data.Common.DbDataSourceEnumerator instance.

Syntax

Visual Basic

```
Public Overrides Function CreateDataSourceEnumerator () As
DbDataSourceEnumerator
```

C#

```
public override DbDataSourceEnumerator CreateDataSourceEnumerator ()
```

Returns

A new SADataSourceEnumerator object typed as DbDataSourceEnumerator.

Related Information

SADataSourceEnumerator Class [page 242]

1.18.7 CreateParameter() Method

Returns a strongly typed System.Data.Common.DbParameter instance.

Syntax

Visual Basic

```
Public Overrides Function CreateParameter () As DbParameter
```

C#

```
public override DbParameter CreateParameter ()
```

Returns

A new SAParameter object typed as DbParameter.

Related Information

SAParameter Class [page 268]

1.18.8 CreatePermission(PermissionState) Method

Returns a strongly-typed CodeAccessPermission instance.

Syntax

Visual Basic

```
Public Overrides Function CreatePermission (ByVal state As
PermissionState) As CodeAccessPermission
```

C#

```
public override CodeAccessPermission CreatePermission (PermissionState
state)
```

Parameters

state A member of the System.Security.Permissions.PermissionState enumeration.

Returns

A new SAPermission object typed as CodeAccessPermission.

Related Information

SACommand Class [page 53]

1.18.9 CanCreateDataSourceEnumerator Property

Always returns true, which indicates that an SADataSourceEnumerator object can be created.

Syntax

Visual Basic

```
Public ReadOnly Overrides Property CanCreateDataSourceEnumerator As Boolean
```

C#

```
public override bool CanCreateDataSourceEnumerator {get;}
```

Returns

True, indicating that an SADataSourceEnumerator object can be created.

Related Information

SADataSourceEnumerator Class [page 242]
SACommand Class [page 53]

1.19 SAInfoMessageEventArgs Class

Provides data for the InfoMessage event.

Syntax

Visual Basic

```
Public NotInheritable Class SAInfoMessageEventArgs Inherits
System.EventArgs
```

C#

```
public sealed class SAInfoMessageEventArgs : System.EventArgs
```

Members

All members of SAInfoMessageEventArgs, including inherited members.

Methods

Modifier and Type	Method	Description
public override string	ToString() [page 264]	Retrieves a string representation of the InfoMessage event.

Properties

Modifier and Type	Property	Description
public SAErrorCollection	Errors [page 264]	Returns the collection of messages sent from the data source.
public string	Message [page 264]	Returns the full text of the error sent from the data source.
public SAMessageType	MessageType [page 265]	Returns the type of the message.
public int	NativeError [page 265]	Returns the SQLCODE returned by the database.
public string	Source [page 265]	Returns the name of the .NET Data Provider.

Remarks

There is no constructor for SAInfoMessageEventArgs.

In this section:

ToString() Method [page 264]
Retrieves a string representation of the InfoMessage event.

Errors Property [page 264]
Returns the collection of messages sent from the data source.

Message Property [page 264]
Returns the full text of the error sent from the data source.

MessageType Property [page 265]
Returns the type of the message.

NativeError Property [page 265]
Returns the SQLCODE returned by the database.

Source Property [page 265]
Returns the name of the .NET Data Provider.

1.19.1 ToString() Method

Retrieves a string representation of the InfoMessage event.

Syntax

Visual Basic

```
Public Overrides Function ToString () As String
```

C#

```
public override string ToString ()
```

Returns

A string representing the InfoMessage event.

1.19.2 Errors Property

Returns the collection of messages sent from the data source.

Syntax

Visual Basic

```
Public ReadOnly Property Errors As SAErrorCollection
```

C#

```
public SAErrorCollection Errors {get;}
```

1.19.3 Message Property

Returns the full text of the error sent from the data source.

Syntax

Visual Basic

```
Public ReadOnly Property Message As String
```

C#

```
public string Message {get;}
```

1.19.4 MessageType Property

Returns the type of the message.

Syntax

Visual Basic

```
Public ReadOnly Property MessageType As SAMessageType
```

C#

```
public SAMessageType MessageType {get;}
```

Remarks

This can be one of: Action, Info, Status, or Warning.

1.19.5 NativeError Property

Returns the SQLCODE returned by the database.

Syntax

Visual Basic

```
Public ReadOnly Property NativeError As Integer
```

C#

```
public int NativeError {get;}
```

1.19.6 Source Property

Returns the name of the .NET Data Provider.

Syntax

Visual Basic

```
Public ReadOnly Property Source As String
```

C#

```
public string Source {get;}
```

1.20 SAMetaDataCollectionNames Class

Provides a list of constants for use with the SAConnection.GetSchema(string) method to retrieve metadata collections.

Syntax

Visual Basic

```
Public NotInheritable Class SAMetaDataCollectionNames
```

C#

```
public sealed class SAMetaDataCollectionNames
```

Members

All members of SAMetaDataCollectionNames, including inherited members.

Variables

Modifier and Type	Variable	Description
public static readonly string	Columns	Provides a constant for use with the SAConnection.GetSchema(string) method that represents the Columns collection.
public static readonly string	DataSourceInformation	Provides a constant for use with the SAConnection.GetSchema(string) method that represents the DataSourceInformation collection.
public static readonly string	DataTypes	Provides a constant for use with the SAConnection.GetSchema(string) method that represents the DataTypes collection.
public static readonly string	ForeignKeys	Provides a constant for use with the SAConnection.GetSchema(string) method that represents the ForeignKeys collection.
public static readonly string	IndexColumns	Provides a constant for use with the SAConnection.GetSchema(string) method that represents the IndexColumns collection.
public static readonly string	Indexes	Provides a constant for use with the SAConnection.GetSchema(string) method that represents the Indexes collection.

Modifier and Type	Variable	Description
public static readonly string	MetaDataCollections	Provides a constant for use with the SAConnection.GetSchema(string) method that represents the MetaDataCollections collection.
public static readonly string	ProcedureParameters	Provides a constant for use with the SAConnection.GetSchema(string) method that represents the ProcedureParameters collection.
public static readonly string	Procedures	Provides a constant for use with the SAConnection.GetSchema(string) method that represents the Procedures collection.
public static readonly string	ReservedWords	Provides a constant for use with the SAConnection.GetSchema(string) method that represents the ReservedWords collection.
public static readonly string	Restrictions	Provides a constant for use with the SAConnection.GetSchema(string) method that represents the Restrictions collection.
public static readonly string	Tables	Provides a constant for use with the SAConnection.GetSchema(string) method that represents the Tables collection.
public static readonly string	Users	Provides a constant for use with the SAConnection.GetSchema(string) method that represents the Users collection.
public static readonly string	UserDefinedTypes	Provides a constant for use with the SAConnection.GetSchema(string) method that represents the UserDefinedTypes collection.
public static readonly string	ViewColumns	Provides a constant for use with the SAConnection.GetSchema(string) method that represents the ViewColumns collection.
public static readonly string	Views	Provides a constant for use with the SAConnection.GetSchema(string) method that represents the Views collection.

Remarks

This field is constant and read-only.

Related Information

GetSchema(string) Method [page 135]

1.21 SAParameter Class

Represents a parameter to an SACommand, and optionally, its mapping to a DataSet column.

Syntax

Visual Basic

```
Public NotInheritable Class SAParameter Inherits
System.Data.Common.DbParameter Implements System.ICloneable
```

C#

```
public sealed class SAParameter : System.Data.Common.DbParameter,
System.ICloneable
```

Members

All members of SAParameter, including inherited members.

Constructors

Modifier and Type	Constructor	Description
public	SAParameter [page 270]	Initializes an SAParameter object with null (Nothing in Visual Basic) as its value.

Methods

Modifier and Type	Method	Description
public override void	ResetDbType() [page 277]	Resets the type (the values of DbType and SADbType) associated with this SAParameter.
public override string	ToString() [page 277]	Returns a string containing the ParameterName.

Properties

Modifier and Type	Property	Description
public override DbType	DbType [page 277]	Gets and sets the DbType of the parameter.

Modifier and Type	Property	Description
public override ParameterDirection	Direction [page 278]	Gets and sets a value indicating whether the parameter is input-only, output-only, bidirectional, or a stored procedure return value parameter.
public override bool	IsNullable [page 278]	Gets and sets a value indicating whether the parameter accepts null values.
public int	Offset [page 279]	Gets and sets the offset to the Value property.
public override string	ParameterName [page 279]	Gets and sets the name of the SAParameter.
public byte	Precision [page 280]	Gets and sets the maximum number of digits used to represent the Value property.
public SADbType	SADbType [page 280]	The SADbType of the parameter.
public byte	Scale [page 281]	Gets and sets the number of decimal places to which Value is resolved.
public override int	Size [page 281]	Gets and sets the maximum size, in bytes, of the data within the column.
public override string	SourceColumn [page 282]	Gets and sets the name of the source column mapped to the DataSet and used for loading or returning the value.
public override bool	SourceColumnNullMapping [page 283]	Gets and sets value that indicates whether the source column is nullable.
public override DataRowVersion	SourceVersion [page 283]	Gets and sets the DataRowVersion to use when loading Value.
public override object	Value [page 284]	Gets and sets the value of the parameter.

Remarks

Implements: IDbDataParameter, IDataParameter, ICloneable

In this section:

SAParameter Constructor [page 270]
Initializes an SAParameter object with null (Nothing in Visual Basic) as its value.

ResetDbType() Method [page 277]
Resets the type (the values of DbType and SADbType) associated with this SAParameter.

ToString() Method [page 277]
Returns a string containing the ParameterName.

DbType Property [page 277]
Gets and sets the DbType of the parameter.

Direction Property [page 278]
Gets and sets a value indicating whether the parameter is input-only, output-only, bidirectional, or a stored procedure return value parameter.

IsNullable Property [page 278]
Gets and sets a value indicating whether the parameter accepts null values.

Offset Property [page 279]
Gets and sets the offset to the Value property.

ParameterName Property [page 279]
Gets and sets the name of the SAParameter.

Precision Property [page 280]
Gets and sets the maximum number of digits used to represent the Value property.

SADbType Property [page 280]
The SADbType of the parameter.

Scale Property [page 281]
Gets and sets the number of decimal places to which Value is resolved.

Size Property [page 281]
Gets and sets the maximum size, in bytes, of the data within the column.

SourceColumn Property [page 282]
Gets and sets the name of the source column mapped to the DataSet and used for loading or returning the value.

SourceColumnNullMapping Property [page 283]
Gets and sets value that indicates whether the source column is nullable.

SourceVersion Property [page 283]
Gets and sets the DataRowVersion to use when loading Value.

Value Property [page 284]
Gets and sets the value of the parameter.

1.21.1 SAParameter Constructor

Initializes an SAParameter object with null (Nothing in Visual Basic) as its value.

Overload list

Modifier and Type	Overload name	Description
public	SAParameter() [page 271]	Initializes an SAParameter object with null (Nothing in Visual Basic) as its value.

Modifier and Type	Overload name	Description
public	SAParameter(string, SADbType) [page 272]	Initializes an SAParameter object with the specified parameter name and data type.
public	SAParameter(string, SADbType, int) [page 273]	Initializes an SAParameter object with the specified parameter name, data type, and size.
public	SAParameter(string, SADbType, int, ParameterDirection, bool, byte, byte, string, DataRowVersion, object) [page 274]	Initializes an SAParameter object with the specified parameter name, data type, size, direction, nullability, numeric precision, numeric scale, source column, source version, and value.
public	SAParameter(string, SADbType, int, string) [page 275]	Initializes an SAParameter object with the specified parameter name, data type, size, and source column.
public	SAParameter(string, object) [page 276]	Initializes an SAParameter object with the specified parameter name and value.

In this section:

SAParameter() Constructor [page 271]
Initializes an SAParameter object with null (Nothing in Visual Basic) as its value.

SAParameter(string, SADbType) Constructor [page 272]
Initializes an SAParameter object with the specified parameter name and data type.

SAParameter(string, SADbType, int) Constructor [page 273]
Initializes an SAParameter object with the specified parameter name, data type, and size.

SAParameter(string, SADbType, int, ParameterDirection, bool, byte, byte, string, DataRowVersion, object) Constructor [page 274]
Initializes an SAParameter object with the specified parameter name, data type, size, direction, nullability, numeric precision, numeric scale, source column, source version, and value.

SAParameter(string, SADbType, int, string) Constructor [page 275]
Initializes an SAParameter object with the specified parameter name, data type, size, and source column.

SAParameter(string, object) Constructor [page 276]
Initializes an SAParameter object with the specified parameter name and value.

1.21.1.1 SAParameter() Constructor

Initializes an SAParameter object with null (Nothing in Visual Basic) as its value.

Syntax

Visual Basic

```
Public Sub SAParameter ()
```

C#

```
public SAParameter ()
```

1.21.1.2 SAParameter(string, SADbType) Constructor

Initializes an SAParameter object with the specified parameter name and data type.

Syntax

Visual Basic

```
Public Sub SAParameter (
    ByVal parameterName As String,
    ByVal dbType As SADbType
)
```

C#

```
public SAParameter (
    string parameterName,
    SADbType dbType
)
```

Parameters

parameterName The name of the parameter.
dbType One of the SADbType values.

Example

The parameter name is "param" in the following example.

```
SELECT * FROM Customers WHERE ID = :param
```

Related Information

SADbType Property [page 280]

1.21.1.3 SAParameter(string, SADbType, int) Constructor

Initializes an SAParameter object with the specified parameter name, data type, and size.

Syntax

Visual Basic

```
Public Sub SAParameter (
    ByVal parameterName As String,
    ByVal dbType As SADbType,
    ByVal size As Integer
)
```

C#

```
public SAParameter (
    string parameterName,
    SADbType dbType,
    int size
)
```

Parameters

parameterName The name of the parameter.
dbType One of the SADbType values.
size The length of the parameter.

Example

The parameter name is "param" in the following example.

```
SELECT * FROM Customers WHERE ID = :param
```

1.21.1.4 SAParameter(string, SADbType, int, ParameterDirection, bool, byte, byte, string, DataRowVersion, object) Constructor

Initializes an SAParameter object with the specified parameter name, data type, size, direction, nullability, numeric precision, numeric scale, source column, source version, and value.

Syntax

Visual Basic

```
Public Sub SAParameter (
    ByVal parameterName As String,
    ByVal dbType As SADbType,
    ByVal size As Integer,
    ByVal direction As ParameterDirection,
    ByVal isNullable As Boolean,
    ByVal precision As Byte,
    ByVal scale As Byte,
    ByVal sourceColumn As String,
    ByVal sourceVersion As DataRowVersion,
    ByVal value As Object
)
```

C#

```
public SAParameter (
    string parameterName,
    SADbType dbType,
    int size,
    ParameterDirection direction,
    bool isNullable,
    byte precision,
    byte scale,
    string sourceColumn,
    DataRowVersion sourceVersion,
    object value
)
```

Parameters

parameterName The name of the parameter.
dbType One of the SADbType values.
size The length of the parameter.
direction One of the ParameterDirection values.
isNullable True if the value of the field can be null; false otherwise.
precision The total number of digits to the left and right of the decimal point to which Value is resolved.
scale The total number of decimal places to which Value is resolved.
sourceColumn The name of the source column to map.
sourceVersion One of the DataRowVersion values.
value An Object that is the value of the parameter.

Example

The name of the parameter. The parameter name is "param" in the following example.

```
SELECT * FROM Customers WHERE ID = :param
```

1.21.1.5 SAParameter(string, SADbType, int, string) Constructor

Initializes an SAParameter object with the specified parameter name, data type, size, and source column.

Syntax

Visual Basic

```
Public Sub SAParameter (
    ByVal parameterName As String,
    ByVal dbType As SADbType,
    ByVal size As Integer,
    ByVal sourceColumn As String
)
```

C#

```
public SAParameter (
    string parameterName,
    SADbType dbType,
    int size,
    string sourceColumn
)
```

Parameters

parameterName The name of the parameter.
dbType One of the SADbType values.
size The length of the parameter.
sourceColumn The name of the source column to map.

Example

The parameter name is "param" in the following example.

```
SELECT * FROM Customers WHERE ID = :param
```

1.21.1.6 SAParameter(string, object) Constructor

Initializes an SAParameter object with the specified parameter name and value.

Syntax

Visual Basic

```
Public Sub SAParameter (
    ByVal parameterName As String,
    ByVal value As Object
)
```

C#

```
public SAParameter (
    string parameterName,
    object value
)
```

Parameters

parameterName The name of the parameter.
value An Object that is the value of the parameter.

Remarks

This constructor is not recommended; it is provided for compatibility with other data providers.

Example

The parameter name is "param" in the following example.

```
SELECT * FROM Customers WHERE ID = :param
```

1.21.2 ResetDbType() Method

Resets the type (the values of DbType and SADbType) associated with this SAParameter.

Syntax

Visual Basic

```
Public Overrides Sub ResetDbType ()
```

C#

```
public override void ResetDbType ()
```

1.21.3 ToString() Method

Returns a string containing the ParameterName.

Syntax

Visual Basic

```
Public Overrides Function ToString () As String
```

C#

```
public override string ToString ()
```

Returns

The name of the parameter.

1.21.4 DbType Property

Gets and sets the DbType of the parameter.

Syntax

Visual Basic

```
Public Overrides Property DbType As DbType
```

C#

```
public override DbType DbType {get;set;}
```

Remarks

The SADbType and DbType are linked. Setting the DbType changes the SADbType to a supporting SADbType.

The value must be a member of the SADbType enumerator.

1.21.5 Direction Property

Gets and sets a value indicating whether the parameter is input-only, output-only, bidirectional, or a stored procedure return value parameter.

Syntax

Visual Basic

```
Public Overrides Property Direction As ParameterDirection
```

C#

```
public override ParameterDirection Direction {get;set;}
```

Remarks

One of the ParameterDirection values.

If the ParameterDirection is output, and execution of the associated SACommand does not return a value, then the SAParameter contains a null value. After the last row from the last result set is read, the Output, InputOut, and ReturnValue parameters are updated.

1.21.6 IsNullable Property

Gets and sets a value indicating whether the parameter accepts null values.

Syntax

Visual Basic

```
Public Overrides Property IsNullable As Boolean
```

C#

```
public override bool IsNullable {get;set;}
```

Remarks

This property is true if null values are accepted; otherwise, it is false. The default is false. Null values are handled using the DBNull class.

1.21.7 Offset Property

Gets and sets the offset to the Value property.

Syntax

Visual Basic

```
Public Property Offset As Integer
```

C#

```
public int Offset {get;set;}
```

Remarks

The offset to the value. The default is 0.

1.21.8 ParameterName Property

Gets and sets the name of the SAParameter.

Syntax

Visual Basic

```
Public Overrides Property ParameterName As String
```

C#

```
public override string ParameterName {get;set;}
```

Remarks

The default is an empty string.

Example

The parameter name is "param" in the following example.

```
SELECT * FROM Customers WHERE ID = :param
```

1.21.9 Precision Property

Gets and sets the maximum number of digits used to represent the Value property.

Syntax

Visual Basic

```
Public Property Precision As Byte
```

C#

```
public byte Precision {get;set;}
```

Remarks

The value of this property is the maximum number of digits used to represent the Value property. The default value is 0, which indicates that the data provider sets the precision for the Value property.

The Precision property is only used for decimal and numeric input parameters.

1.21.10 SADbType Property

The SADbType of the parameter.

Syntax

Visual Basic

```
Public Property SADbType As SADbType
```

C#

```
public SADbType SADbType {get;set;}
```

Remarks

The SADbType and DbType are linked. Setting the SADbType changes the DbType to a supporting DbType.

The value must be a member of the SADbType enumerator.

1.21.11 Scale Property

Gets and sets the number of decimal places to which Value is resolved.

Syntax

Visual Basic

```
Public Property Scale As Byte
```

C#

```
public byte Scale {get;set;}
```

Remarks

The number of decimal places to which Value is resolved. The default is 0.

The Scale property is only used for decimal and numeric input parameters.

1.21.12 Size Property

Gets and sets the maximum size, in bytes, of the data within the column.

Syntax

Visual Basic

```
Public Overrides Property Size As Integer
```

C#

```
public override int Size {get;set;}
```

Remarks

The value of this property is the maximum size, in bytes, of the data within the column. The default value is inferred from the parameter value.

The Size property is used for binary and string types.

For variable length data types, the Size property describes the maximum amount of data to transmit to the database server. For example, the Size property can be used to limit the amount of data sent to the database server for a string value to the first one hundred bytes.

If Size is not explicitly set, then it is inferred from the actual size of the specified parameter value. For fixed width data types, the value of Size is ignored. It can be retrieved for informational purposes, and returns the maximum amount of bytes the provider uses when transmitting the value of the parameter to the database server.

1.21.13 SourceColumn Property

Gets and sets the name of the source column mapped to the DataSet and used for loading or returning the value.

Syntax

Visual Basic

```
Public Overrides Property SourceColumn As String
```

C#

```
public override string SourceColumn {get;set;}
```

Remarks

A string specifying the name of the source column mapped to the DataSet and used for loading or returning the value.

When SourceColumn is set to anything other than an empty string, the value of the parameter is retrieved from the column with the SourceColumn name. If Direction is set to Input, then the value is taken from the DataSet. If Direction is set to Output, then the value is taken from the data source. A Direction of InputOutput is a combination of both.

1.21.14 SourceColumnNullMapping Property

Gets and sets value that indicates whether the source column is nullable.

Syntax

Visual Basic

```
Public Overrides Property SourceColumnNullMapping As Boolean
```

C#

```
public override bool SourceColumnNullMapping {get;set;}
```

Remarks

This property allows SACommandBuilder to generate Update statements for nullable columns correctly.

If the source column is nullable, then true is returned; otherwise, false is returned.

1.21.15 SourceVersion Property

Gets and sets the DataRowVersion to use when loading Value.

Syntax

Visual Basic

```
Public Overrides Property SourceVersion As DataRowVersion
```

C#

```
public override DataRowVersion SourceVersion {get;set;}
```

Remarks

Used by UpdateCommand during an Update operation to determine whether the parameter value is set to Current or Original. This property allows primary keys to be updated. This property is ignored by InsertCommand and DeleteCommand. This property is set to the version of the DataRow used by the Item property, or the GetChildRows method of the DataRow object.

1.21.16 Value Property

Gets and sets the value of the parameter.

Syntax

Visual Basic

```
Public Overrides Property Value As Object
```

C#

```
public override object Value {get;set;}
```

Remarks

An Object that specifies the value of the parameter.

For input parameters, the value is bound to the SACommand that is sent to the database server. For output and return value parameters, the value is set on completion of the SACommand and after the SADataReader is closed.

When sending a null parameter value to the database server, specify DBNull, not null. The null value in the system is an empty object that has no value. DBNull is used to represent null values.

If the application specifies the database type, then the bound value is converted to that type when the .NET Data Provider sends the data to the database server. The provider attempts to convert any type of value if it supports the IConvertible interface. Conversion errors may result if the specified type is not compatible with the value.

Both the DbType and SADbType properties can be inferred by setting the Value.

The Value property is overwritten by Update.

1.22 SAParameterCollection Class

Represents all parameters to an SACommand object and, optionally, their mapping to a DataSet column.

Syntax

Visual Basic

```
Public NotInheritable Class SAParameterCollection Inherits
System.Data.Common.DbParameterCollection
```

C#

```
public sealed class SAParameterCollection :
System.Data.Common.DbParameterCollection
```

Members

All members of SAParameterCollection, including inherited members.

Methods

Modifier and Type	Method	Description
public SAParameter	Add [page 287]	Adds an SAParameter object to this collection.
public override void	AddRange [page 294]	Adds an array of values to the end of the SAParameterCollection.
public SAParameter	AddWithValue(string, object) [page 295]	Adds a value to the end of this collection.
public override void	Clear() [page 296]	Removes all items from the collection.
public override bool	Contains [page 296]	Indicates whether an SAParameter object exists in the collection.
public override void	CopyTo(Array, int) [page 298]	Copies SAParameter objects from the SAParameterCollection to the specified array.
public override IEnumerator	GetEnumerator() [page 299]	Returns an enumerator that iterates through the SAParameterCollection.
protected override DbParameter	GetParameter [page 300]	Returns a parameter from the SAParameterCollection object.
public override int	IndexOf [page 302]	Returns the location of the SAParameter object in the collection.
public override void	Insert(int, object) [page 304]	Inserts an SAParameter object in the collection at the specified index.
public override void	Remove(object) [page 304]	Removes the specified SAParameter object from the collection.
public override void	RemoveAt [page 305]	Removes the specified SAParameter object from the collection.
protected override void	SetParameter [page 306]	Sets a parameter in the SAParameterCollection object.

Properties

Modifier and Type	Property	Description
public override int	Count [page 308]	Returns the number of SAParameter objects in the collection.
public override bool	IsFixedSize [page 309]	Gets a value that indicates whether the SAParameterCollection has a fixed size.
public override bool	IsReadOnly [page 309]	Gets a value that indicates whether the SAParameterCollection is read-only.
public override bool	IsSynchronized [page 310]	Gets a value that indicates whether the SAParameterCollection object is synchronized.

Modifier and Type	Property	Description
public override object	SyncRoot [page 310]	Gets an object that can be used to synchronize access to the SAParameterCollection.
public new SAParameter	this [page 311]	Gets and sets the SAParameter object at the specified index.

Remarks

There is no constructor for SAParameterCollection. You obtain an SAParameterCollection object from the SACommand.Parameters property of an SACommand object.

In this section:

Add Method [page 287]
Adds an SAParameter object to this collection.

AddRange Method [page 294]
Adds an array of values to the end of the SAParameterCollection.

AddWithValue(string, object) Method [page 295]
Adds a value to the end of this collection.

Clear() Method [page 296]
Removes all items from the collection.

Contains Method [page 296]
Indicates whether an SAParameter object exists in the collection.

CopyTo(Array, int) Method [page 298]
Copies SAParameter objects from the SAParameterCollection to the specified array.

GetEnumerator() Method [page 299]
Returns an enumerator that iterates through the SAParameterCollection.

GetParameter Method [page 300]
Returns a parameter from the SAParameterCollection object.

IndexOf Method [page 302]
Returns the location of the SAParameter object in the collection.

Insert(int, object) Method [page 304]
Inserts an SAParameter object in the collection at the specified index.

Remove(object) Method [page 304]
Removes the specified SAParameter object from the collection.

RemoveAt Method [page 305]
Removes the specified SAParameter object from the collection.

SetParameter Method [page 306]
Sets a parameter in the SAParameterCollection object.

Count Property [page 308]

Returns the number of SAParameter objects in the collection.

IsFixedSize Property [page 309]
Gets a value that indicates whether the SAParameterCollection has a fixed size.

IsReadOnly Property [page 309]
Gets a value that indicates whether the SAParameterCollection is read-only.

IsSynchronized Property [page 310]
Gets a value that indicates whether the SAParameterCollection object is synchronized.

SyncRoot Property [page 310]
Gets an object that can be used to synchronize access to the SAParameterCollection.

this Property [page 311]
Gets and sets the SAParameter object at the specified index.

Related Information

SACommand Class [page 53]
Parameters Property [page 89]
SAParameter Class [page 268]

1.22.1 Add Method

Adds an SAParameter object to this collection.

Overload list

Modifier and Type	Overload name	Description
public SAParameter	Add(SAParameter) [page 288]	Adds an SAParameter object to this collection.
public override int	Add(object) [page 289]	Adds an SAParameter object to this collection.
public SAParameter	Add(string, SADbType) [page 290]	Adds an SAParameter object to this collection, created using the specified parameter name and data type, to the collection.
public SAParameter	Add(string, SADbType, int) [page 291]	Adds an SAParameter object to this collection, created using the specified parameter name, data type, and length, to the collection.

Modifier and Type	Overload name	Description
public SAParameter	Add(string, SADbType, int, string) [page 292]	Adds an SAParameter object to this collection, created using the specified parameter name, data type, length, and source column name, to the collection.
public SAParameter	Add(string, object) [page 293]	Adds an SAParameter object to this collection, created using the specified parameter name and value, to the collection.

In this section:

Add(SAParameter) Method [page 288]
Adds an SAParameter object to this collection.

Add(object) Method [page 289]
Adds an SAParameter object to this collection.

Add(string, SADbType) Method [page 290]
Adds an SAParameter object to this collection, created using the specified parameter name and data type, to the collection.

Add(string, SADbType, int) Method [page 291]
Adds an SAParameter object to this collection, created using the specified parameter name, data type, and length, to the collection.

Add(string, SADbType, int, string) Method [page 292]
Adds an SAParameter object to this collection, created using the specified parameter name, data type, length, and source column name, to the collection.

Add(string, object) Method [page 293]
Adds an SAParameter object to this collection, created using the specified parameter name and value, to the collection.

1.22.1.1 Add(SAParameter) Method

Adds an SAParameter object to this collection.

Syntax

Visual Basic

```
Public Function Add (ByVal value As SAParameter) As SAParameter
```

C#

```
public SAParameter Add (SAParameter value)
```

Parameters

value The SAParameter object to add to the collection.

Returns

The new SAParameter object.

1.22.1.2 Add(object) Method

Adds an SAParameter object to this collection.

Syntax

Visual Basic

```
Public Overrides Function Add (ByVal value As Object) As Integer
```

C#

```
public override int Add (object value)
```

Parameters

value The SAParameter object to add to the collection.

Returns

The index of the new SAParameter object.

Related Information

SAParameter Class [page 268]

1.22.1.3 Add(string, SADbType) Method

Adds an SAParameter object to this collection, created using the specified parameter name and data type, to the collection.

Syntax

Visual Basic

```
Public Function Add (
    ByVal parameterName As String,
    ByVal saDbType As SADbType
) As SAParameter
```

C#

```
public SAParameter Add (
    string parameterName,
    SADbType saDbType
)
```

Parameters

parameterName The name of the parameter.
saDbType One of the SADbType values.

Returns

The new SAParameter object.

Related Information

SADbType Enumeration [page 346]
Add(SAParameter) Method [page 288]
Add(string, object) Method [page 293]

1.22.1.4 Add(string, SADbType, int) Method

Adds an SAParameter object to this collection, created using the specified parameter name, data type, and length, to the collection.

Syntax

Visual Basic

```
Public Function Add (
    ByVal parameterName As String,
    ByVal saDbType As SADbType,
    ByVal size As Integer
) As SAParameter
```

C#

```
public SAParameter Add (
    string parameterName,
    SADbType saDbType,
    int size
)
```

Parameters

parameterName The name of the parameter.
saDbType One of the SADbType values.
size The length of the parameter.

Returns

The new SAParameter object.

Related Information

SADbType Enumeration [page 346]
Add(SAParameter) Method [page 288]
Add(string, object) Method [page 293]

1.22.1.5 Add(string, SADbType, int, string) Method

Adds an SAParameter object to this collection, created using the specified parameter name, data type, length, and source column name, to the collection.

Syntax

Visual Basic

```
Public Function Add (
    ByVal parameterName As String,
    ByVal saDbType As SADbType,
    ByVal size As Integer,
    ByVal sourceColumn As String
) As SAParameter
```

C#

```
public SAParameter Add (
    string parameterName,
    SADbType saDbType,
    int size,
    string sourceColumn
)
```

Parameters

parameterName The name of the parameter.
saDbType One of the SADbType values.
size The length of the column.
sourceColumn The name of the source column to map.

Returns

The new SAParameter object.

Related Information

SADbType Enumeration [page 346]
Add(SAParameter) Method [page 288]
Add(string, object) Method [page 293]

1.22.1.6 Add(string, object) Method

Adds an SAParameter object to this collection, created using the specified parameter name and value, to the collection.

Syntax

Visual Basic

```
Public Function Add (
    ByVal parameterName As String,
    ByVal value As Object
) As SAParameter
```

C#

```
public SAParameter Add (
    string parameterName,
    object value
)
```

Parameters

parameterName The name of the parameter.
value The value of the parameter to add to the connection.

Returns

The new SAParameter object.

Remarks

Because of the special treatment of the 0 and 0.0 constants and the way overloaded methods are resolved, explicitly cast constant values to the desired object type when using this method.

Related Information

SAParameter Class [page 268]

1.22.2 AddRange Method

Adds an array of values to the end of the SAParameterCollection.

Overload list

Modifier and Type	Overload name	Description
public override void	AddRange(Array) [page 294]	Adds an array of values to the end of the SAParameterCollection.
public void	AddRange(SAParameter[]) [page 295]	Adds an array of values to the end of the SAParameterCollection.

In this section:

AddRange(Array) Method [page 294]
Adds an array of values to the end of the SAParameterCollection.

AddRange(SAParameter[]) Method [page 295]
Adds an array of values to the end of the SAParameterCollection.

1.22.2.1 AddRange(Array) Method

Adds an array of values to the end of the SAParameterCollection.

Syntax

Visual Basic

```
Public Overrides Sub AddRange (ByVal values As Array)
```

C#

```
public override void AddRange (Array values)
```

Parameters

values The values to add.

1.22.2.2 AddRange(SAParameter[]) Method

Adds an array of values to the end of the SAParameterCollection.

Syntax

Visual Basic

```
Public Sub AddRange (ByVal values As SAParameter())
```

C#

```
public void AddRange (SAParameter[] values)
```

Parameters

values An array of SAParameter objects to add to the end of this collection.

1.22.3 AddWithValue(string, object) Method

Adds a value to the end of this collection.

Syntax

Visual Basic

```
Public Function AddWithValue (
    ByVal parameterName As String,
    ByVal value As Object
) As SAParameter
```

C#

```
public SAParameter AddWithValue (
    string parameterName,
    object value
)
```

Parameters

parameterName The name of the parameter.
value The value to be added.

Returns

The new SAParameter object.

1.22.4 Clear() Method

Removes all items from the collection.

Syntax

Visual Basic

```
Public Overrides Sub Clear ()
```

C#

```
public override void Clear ()
```

1.22.5 Contains Method

Indicates whether an SAParameter object exists in the collection.

Overload list

Modifier and Type	Overload name	Description
public override bool	Contains(object) [page 297]	Indicates whether an SAParameter object exists in the collection.
public override bool	Contains(string) [page 297]	Indicates whether an SAParameter object exists in the collection.

In this section:

Contains(object) Method [page 297]
Indicates whether an SAParameter object exists in the collection.

Contains(string) Method [page 297]
Indicates whether an SAParameter object exists in the collection.

1.22.5.1 Contains(object) Method

Indicates whether an SAParameter object exists in the collection.

Syntax

Visual Basic

```
Public Overrides Function Contains (ByVal value As Object) As Boolean
```

C#

```
public override bool Contains (object value)
```

Parameters

value The SAParameter object to find.

Returns

True if the collection contains the SAParameter object; false otherwise.

Related Information

SAParameter Class [page 268]
Contains(string) Method [page 297]

1.22.5.2 Contains(string) Method

Indicates whether an SAParameter object exists in the collection.

Syntax

Visual Basic

```
Public Overrides Function Contains (ByVal value As String) As Boolean
```

C#

```
public override bool Contains (string value)
```

Parameters

value The name of the parameter to search for.

Returns

True if the collection contains the SAParameter object; false otherwise.

Related Information

SAParameter Class [page 268]
Contains(object) Method [page 297]

1.22.6 CopyTo(Array, int) Method

Copies SAParameter objects from the SAParameterCollection to the specified array.

Syntax

Visual Basic

```
Public Overrides Sub CopyTo (
    ByVal array As Array,
    ByVal index As Integer
)
```

C#

```
public override void CopyTo (
    Array array,
    int index
)
```

Parameters

array The array to copy the SAParameter objects into.
index The starting index of the array.

Related Information

SAParameter Class [page 268]
SAParameterCollection Class [page 284]

1.22.7 GetEnumerator() Method

Returns an enumerator that iterates through the SAParameterCollection.

Syntax

Visual Basic

```
Public Overrides Function GetEnumerator () As
System.Collections.IEnumerator
```

C#

```
public override IEnumerator GetEnumerator ()
```

Returns

A System.Collections.IEnumerator for the SAParameterCollection object.

Related Information

SAParameterCollection Class [page 284]

1.22.8 GetParameter Method

Returns a parameter from the SAParameterCollection object.

Overload list

Modifier and Type	Overload name	Description
protected override DbParameter	GetParameter(int) [page 300]	Returns a parameter from the SAParameterCollection object.
protected override DbParameter	GetParameter(string) [page 301]	Returns a parameter from the SAParameterCollection object.

In this section:

GetParameter(int) Method [page 300]
Returns a parameter from the SAParameterCollection object.

GetParameter(string) Method [page 301]
Returns a parameter from the SAParameterCollection object.

1.22.8.1 GetParameter(int) Method

Returns a parameter from the SAParameterCollection object.

Syntax

Visual Basic

```
Protected Overrides Function GetParameter (ByVal index As Integer) As
DbParameter
```

C#

```
protected override DbParameter GetParameter (int index)
```

Parameters

index The zero-based index of the parameter within the collection.

Returns

A System.Data.Common.DbParameter from SAParameterCollection object.

Related Information

SAParameterCollection Class [page 284]

1.22.8.2 GetParameter(string) Method

Returns a parameter from the SAParameterCollection object.

Syntax

Visual Basic

```
Protected Overrides Function GetParameter (ByVal parameterName As String)
As DbParameter
```

C#

```
protected override DbParameter GetParameter (string parameterName)
```

Parameters

parameterName The name of the parameter to locate.

Returns

A System.Data.Common.DbParameter from SAParameterCollection object.

Related Information

SAParameterCollection Class [page 284]

1.22.9 IndexOf Method

Returns the location of the SAParameter object in the collection.

Overload list

Modifier and Type	Overload name	Description
public override int	IndexOf(object) [page 302]	Returns the location of the SAParameter object in the collection.
public override int	IndexOf(string) [page 303]	Returns the location of the SAParameter object in the collection.

In this section:

IndexOf(object) Method [page 302]
Returns the location of the SAParameter object in the collection.

IndexOf(string) Method [page 303]
Returns the location of the SAParameter object in the collection.

1.22.9.1 IndexOf(object) Method

Returns the location of the SAParameter object in the collection.

Syntax

Visual Basic

```
Public Overrides Function IndexOf (ByVal value As Object) As Integer
```

C#

```
public override int IndexOf (object value)
```

Parameters

value The SAParameter object to locate.

Returns

The zero-based location of the SAParameter object in the collection.

Related Information

SAParameter Class [page 268]
IndexOf(string) Method [page 303]

1.22.9.2 IndexOf(string) Method

Returns the location of the SAParameter object in the collection.

Syntax

Visual Basic

```
Public Overrides Function IndexOf (ByVal parameterName As String) As
Integer
```

C#

```
public override int IndexOf (string parameterName)
```

Parameters

parameterName The name of the parameter to locate.

Returns

The zero-based index of the SAParameter object in the collection.

Related Information

SAParameter Class [page 268]
IndexOf(object) Method [page 302]

1.22.10 Insert(int, object) Method

Inserts an SAParameter object in the collection at the specified index.

Syntax

Visual Basic

```
Public Overrides Sub Insert (
    ByVal index As Integer,
    ByVal value As Object
)
```

C#

```
public override void Insert (
    int index,
    object value
)
```

Parameters

index The zero-based index where the parameter is to be inserted within the collection.
value The SAParameter object to add to the collection.

1.22.11 Remove(object) Method

Removes the specified SAParameter object from the collection.

Syntax

Visual Basic

```
Public Overrides Sub Remove (ByVal value As Object)
```

C#

```
public override void Remove (object value)
```

Parameters

value The SAParameter object to remove from the collection.

1.22.12 RemoveAt Method

Removes the specified SAParameter object from the collection.

Overload list

Modifier and Type	Overload name	Description
public override void	RemoveAt(int) [page 305]	Removes the specified SAParameter object from the collection.
public override void	RemoveAt(string) [page 306]	Removes the specified SAParameter object from the collection.

In this section:

1.22.12.1 RemoveAt(int) Method

Removes the specified SAParameter object from the collection.

Syntax

Visual Basic

```
Public Overrides Sub RemoveAt (ByVal index As Integer)
```

C#

```
public override void RemoveAt (int index)
```

Parameters

index The zero-based index of the parameter to remove.

Related Information

RemoveAt(string) Method [page 306]

1.22.12.2 RemoveAt(string) Method

Removes the specified SAParameter object from the collection.

Syntax

Visual Basic

```
Public Overrides Sub RemoveAt (ByVal parameterName As String)
```

C#

```
public override void RemoveAt (string parameterName)
```

Parameters

parameterName The name of the SAParameter object to remove.

Related Information

RemoveAt(int) Method [page 305]

1.22.13 SetParameter Method

Sets a parameter in the SAParameterCollection object.

Overload list

Modifier and Type	Overload name	Description
protected override void	SetParameter(int, DbParameter) [page 307]	Sets a parameter in the SAParameter-Collection object.

Modifier and Type	Overload name	Description
protected override void	SetParameter(string, DbParameter) [page 308]	Sets a parameter in the SAParameter-Collection object.

In this section:

SetParameter(int, DbParameter) Method [page 307]
Sets a parameter in the SAParameterCollection object.

SetParameter(string, DbParameter) Method [page 308]
Sets a parameter in the SAParameterCollection object.

1.22.13.1 SetParameter(int, DbParameter) Method

Sets a parameter in the SAParameterCollection object.

Syntax

Visual Basic

```
Protected Overrides Sub SetParameter (
    ByVal index As Integer,
    ByVal value As DbParameter
)
```

C#

```
protected override void SetParameter (
    int index,
    DbParameter value
)
```

Parameters

index The zero-based index of the parameter to set.
value A System.Data.Common.DbParameter to be inserted into the SAParameterCollection object.

Related Information

SAParameterCollection Class [page 284]

1.22.13.2 SetParameter(string, DbParameter) Method

Sets a parameter in the SAParameterCollection object.

Syntax

Visual Basic

```
Protected Overrides Sub SetParameter (
    ByVal parameterName As String,
    ByVal value As DbParameter
)
```

C#

```
protected override void SetParameter (
    string parameterName,
    DbParameter value
)
```

Parameters

parameterName The name of the parameter to set.
value A System.Data.Common.DbParameter to be inserted into the SAParameterCollection object.

Related Information

SAParameterCollection Class [page 284]

1.22.14 Count Property

Returns the number of SAParameter objects in the collection.

Syntax

Visual Basic

```
Public ReadOnly Overrides Property Count As Integer
```

C#

```
public override int Count {get;}
```

Remarks

The number of SAParameter objects in the collection.

Related Information

SAParameter Class [page 268]
SAParameterCollection Class [page 284]

1.22.15 IsFixedSize Property

Gets a value that indicates whether the SAParameterCollection has a fixed size.

Syntax

Visual Basic

```
Public ReadOnly Overrides Property IsFixedSize As Boolean
```

C#

```
public override bool IsFixedSize {get;}
```

Remarks

True if this collection has a fixed size; false otherwise.

1.22.16 IsReadOnly Property

Gets a value that indicates whether the SAParameterCollection is read-only.

Syntax

Visual Basic

```
Public ReadOnly Overrides Property IsReadOnly As Boolean
```

C#

```
public override bool IsReadOnly {get;}
```

Remarks

True if this collection is read-only; false otherwise.

1.22.17 IsSynchronized Property

Gets a value that indicates whether the SAParameterCollection object is synchronized.

Syntax

Visual Basic

```
Public ReadOnly Overrides Property IsSynchronized As Boolean
```

C#

```
public override bool IsSynchronized {get;}
```

Remarks

True if this collection is synchronized; false otherwise.

1.22.18 SyncRoot Property

Gets an object that can be used to synchronize access to the SAParameterCollection.

Syntax

Visual Basic

```
Public ReadOnly Overrides Property SyncRoot As Object
```

C#

```
public override object SyncRoot {get;}
```

1.22.19 this Property

Gets and sets the SAParameter object at the specified index.

Overload list

Modifier and Type	Overload name	Description
public new SAParameter	this[int index] [page 311]	Gets and sets the SAParameter object at the specified index.
public new SAParameter	this[string parameterName] [page 312]	Gets and sets the SAParameter object at the specified index.

In this section:

this[int index] Property [page 311]
Gets and sets the SAParameter object at the specified index.

this[string parameterName] Property [page 312]
Gets and sets the SAParameter object at the specified index.

1.22.19.1 this[int index] Property

Gets and sets the SAParameter object at the specified index.

Syntax

Visual Basic

```
Public Shadows Property Item (ByVal index As Integer) As SAParameter
```

C#

```
public new SAParameter this[int index] {get;set;}
```

Returns

The SAParameter at the specified index.

Remarks

An SAParameter object.

In C#, this property is the indexer for the SAParameterCollection object.

Related Information

SAParameter Class [page 268]
SAParameterCollection Class [page 284]

1.22.19.2 this[string parameterName] Property

Gets and sets the SAParameter object at the specified index.

Syntax

Visual Basic

```
Public Shadows Property Item (ByVal parameterNameAs String) As SAParameter
```

C#

```
public new SAParameter this[string parameterName] {get;set;}
```

Returns

The SAParameter object with the specified name.

Remarks

An SAParameter object.

In C#, this property is the indexer for the SAParameterCollection object.

Related Information

SAParameter Class [page 268]
SAParameterCollection Class [page 284]

GetOrdinal(string) Method [page 225]
GetValue(int) Method [page 233]
GetFieldType(int) Method [page 220]

1.23 SAPermission Class

Enables the .NET Data Provider to ensure that a user has a security level adequate to access a data source.

Syntax

Visual Basic

```
Public NotInheritable Class SAPermission Inherits
System.Data.Common.DBDataPermission
```

C#

```
public sealed class SAPermission : System.Data.Common.DBDataPermission
```

Members

All members of SAPermission, including inherited members.

Constructors

Modifier and Type	Constructor	Description
public	SAPermission(PermissionState) [page 314]	Initializes a new instance of the SAPermission class.

Methods

Modifier and Type	Method	Description
protected override DBDataPermission	CreateInstance() [page 314]	Creates a new instance of an SAPermission class.

In this section:

SAPermission(PermissionState) Constructor [page 314]
Initializes a new instance of the SAPermission class.

CreateInstance() Method [page 314]
Creates a new instance of an SAPermission class.

1.23.1 SAPermission(PermissionState) Constructor

Initializes a new instance of the SAPermission class.

Syntax

Visual Basic

```
Public Sub SAPermission (ByVal state As PermissionState)
```

C#

```
public SAPermission (PermissionState state)
```

Parameters

state One of the PermissionState values.

1.23.2 CreateInstance() Method

Creates a new instance of an SAPermission class.

Syntax

Visual Basic

```
Protected Overrides Function CreateInstance () As DBDataPermission
```

C#

```
protected override DBDataPermission CreateInstance ()
```

Returns

A new SAPermission object.

1.24 SAPermissionAttribute Class

Associates a security action with a custom security attribute.

Syntax

Visual Basic

```
Public NotInheritable Class SAPermissionAttribute Inherits
System.Data.Common.DBDataPermissionAttribute
```

C#

```
public sealed class SAPermissionAttribute :
System.Data.Common.DBDataPermissionAttribute
```

Members

All members of SAPermissionAttribute, including inherited members.

Constructors

Modifier and Type	Constructor	Description
public	SAPermissionAttribute(SecurityAction) [page 316]	Initializes a new instance of the SAPermissionAttribute class.

Methods

Modifier and Type	Method	Description
public override IPermission	CreatePermission() [page 316]	Returns an SAPermission object that is configured according to the attribute properties.

In this section:

SAPermissionAttribute(SecurityAction) Constructor [page 316]
Initializes a new instance of the SAPermissionAttribute class.

CreatePermission() Method [page 316]
Returns an SAPermission object that is configured according to the attribute properties.

1.24.1 SAPermissionAttribute(SecurityAction) Constructor

Initializes a new instance of the SAPermissionAttribute class.

Syntax

Visual Basic

```
Public Sub SAPermissionAttribute (ByVal action As SecurityAction)
```

C#

```
public SAPermissionAttribute (SecurityAction action)
```

Parameters

action One of the SecurityAction values representing an action that can be performed using declarative security.

Returns

An SAPermissionAttribute object.

1.24.2 CreatePermission() Method

Returns an SAPermission object that is configured according to the attribute properties.

Syntax

Visual Basic

```
Public Overrides Function CreatePermission () As IPermission
```

C#

```
public override IPermission CreatePermission ()
```

1.25 SARowsCopiedEventArgs Class

Represents the set of arguments passed to the SARowsCopiedEventHandler.

Syntax

Visual Basic

```
Public NotInheritable Class SARowsCopiedEventArgs
```

C#

```
public sealed class SARowsCopiedEventArgs
```

Members

All members of SARowsCopiedEventArgs, including inherited members.

Constructors

Modifier and Type	Constructor	Description
public	SARowsCopiedEventArgs(long) [page 318]	Creates a new instance of the SARowsCopiedEventArgs object.

Properties

Modifier and Type	Property	Description
public bool	Abort [page 318]	Gets or sets a value that indicates whether the bulk-copy operation should be aborted.
public long	RowsCopied [page 318]	Gets the number of rows copied during the current bulk-copy operation.

In this section:

SARowsCopiedEventArgs(long) Constructor [page 318]
Creates a new instance of the SARowsCopiedEventArgs object.

Abort Property [page 318]
Gets or sets a value that indicates whether the bulk-copy operation should be aborted.

RowsCopied Property [page 318]
Gets the number of rows copied during the current bulk-copy operation.

1.25.1 SARowsCopiedEventArgs(long) Constructor

Creates a new instance of the SARowsCopiedEventArgs object.

Syntax

Visual Basic

```
Public Sub SARowsCopiedEventArgs (ByVal rowsCopied As Long)
```

C#

```
public SARowsCopiedEventArgs (long rowsCopied)
```

Parameters

rowsCopied An 64-bit integer value that indicates the number of rows copied during the current bulk-copy operation.

1.25.2 Abort Property

Gets or sets a value that indicates whether the bulk-copy operation should be aborted.

Syntax

Visual Basic

```
Public Property Abort As Boolean
```

C#

```
public bool Abort {get;set;}
```

1.25.3 RowsCopied Property

Gets the number of rows copied during the current bulk-copy operation.

Syntax

Visual Basic

```
Public ReadOnly Property RowsCopied As Long
```

C#

```
public long RowsCopied {get;}
```

1.26 SARowUpdatedEventArgs Class

Provides data for the RowUpdated event.

Syntax

Visual Basic

```
Public NotInheritable Class SARowUpdatedEventArgs Inherits
System.Data.Common.RowUpdatedEventArgs
```

C#

```
public sealed class SARowUpdatedEventArgs :
System.Data.Common.RowUpdatedEventArgs
```

Members

All members of SARowUpdatedEventArgs, including inherited members.

Constructors

Modifier and Type	Constructor	Description
public	SARowUpdatedEventArgs(DataRow, IDbCommand, StatementType, DataTableMapping) [page 320]	Initializes a new instance of the SARowUpdatedEventArgs class.

Properties

Modifier and Type	Property	Description
public new SACommand	Command [page 320]	Gets the SACommand that is executed when DataAdapter.Update is called.
public new int	RecordsAffected [page 321]	Returns the number of rows changed, inserted, or deleted by the execution of the SQL statement.

In this section:

SARowUpdatedEventArgs(DataRow, IDbCommand, StatementType, DataTableMapping) Constructor [page 320]
Initializes a new instance of the SARowUpdatedEventArgs class.

Command Property [page 320]
Gets the SACommand that is executed when DataAdapter.Update is called.

RecordsAffected Property [page 321]
Returns the number of rows changed, inserted, or deleted by the execution of the SQL statement.

1.26.1 SARowUpdatedEventArgs(DataRow, IDbCommand, StatementType, DataTableMapping) Constructor

Initializes a new instance of the SARowUpdatedEventArgs class.

Syntax

Visual Basic

```
Public Sub SARowUpdatedEventArgs (
    ByVal row As DataRow,
    ByVal command As IDbCommand,
    ByVal statementType As StatementType,
    ByVal tableMapping As DataTableMapping
)
```

C#

```
public SARowUpdatedEventArgs (
    DataRow row,
    IDbCommand command,
    StatementType statementType,
    DataTableMapping tableMapping
)
```

Parameters

row The DataRow sent through an Update.
command The IDbCommand executed when Update is called.
statementType One of the StatementType values that specifies the type of query executed.
tableMapping The DataTableMapping sent through an Update.

1.26.2 Command Property

Gets the SACommand that is executed when DataAdapter.Update is called.

Syntax

Visual Basic

```
Public ReadOnly Shadows Property Command As SACommand
```

C#

```
public new SACommand Command {get;}
```

1.26.3 RecordsAffected Property

Returns the number of rows changed, inserted, or deleted by the execution of the SQL statement.

Syntax

Visual Basic

```
Public ReadOnly Shadows Property RecordsAffected As Integer
```

C#

```
public new int RecordsAffected {get;}
```

Remarks

The number of rows changed, inserted, or deleted; 0 if no rows were affected or the statement failed; and -1 for SELECT statements.

1.27 SARowUpdatingEventArgs Class

Provides data for the RowUpdating event.

Syntax

Visual Basic

```
Public NotInheritable Class SARowUpdatingEventArgs Inherits
System.Data.Common.RowUpdatingEventArgs
```

C#

```
public sealed class SARowUpdatingEventArgs :
System.Data.Common.RowUpdatingEventArgs
```

Members

All members of SARowUpdatingEventArgs, including inherited members.

Constructors

Modifier and Type	Constructor	Description
public	SARowUpdatingEventArgs(DataRow, IDbCommand, StatementType, DataTableMapping) [page 322]	Initializes a new instance of the SARowUpdatingEventArgs class.

Properties

Modifier and Type	Property	Description
public new SACommand	Command [page 323]	Specifies the SACommand to execute when performing the Update.

In this section:

SARowUpdatingEventArgs(DataRow, IDbCommand, StatementType, DataTableMapping) Constructor [page 322]
Initializes a new instance of the SARowUpdatingEventArgs class.

Command Property [page 323]
Specifies the SACommand to execute when performing the Update.

1.27.1 SARowUpdatingEventArgs(DataRow, IDbCommand, StatementType, DataTableMapping) Constructor

Initializes a new instance of the SARowUpdatingEventArgs class.

Syntax

Visual Basic

```
Public Sub SARowUpdatingEventArgs (
    ByVal row As DataRow,
    ByVal command As IDbCommand,
    ByVal statementType As StatementType,
    ByVal tableMapping As DataTableMapping
)
```

C#

```
public SARowUpdatingEventArgs (
    DataRow row,
    IDbCommand command,
    StatementType statementType,
    DataTableMapping tableMapping
)
```

Parameters

row The DataRow to update.

command The IDbCommand to execute during update.
statementType One of the StatementType values that specifies the type of query executed.
tableMapping The DataTableMapping sent through an Update.

1.27.2 Command Property

Specifies the SACommand to execute when performing the Update.

Syntax

Visual Basic

```
Public Shadows Property Command As SACommand
```

C#

```
public new SACommand Command {get;set;}
```

1.28 SAServerSideConnection Class

Represents a server-side connection to a database.

Syntax

Visual Basic

```
Public NotInheritable Class SAServerSideConnection
```

C#

```
public sealed class SAServerSideConnection
```

Members

All members of SAServerSideConnection, including inherited members.

Properties

Modifier and Type	Property	Description
public SAConnection	Connection [page 324]	Provides the SAConnection object required to execute SQL on the database server.

Remarks

This class supports the execution of server-side SQL from the CLR external environment. A CLR function or method called from the database server can call back into the database server. A connection is already established to the database server. It is not necessary to use SAConnection to establish a connection.

In this section:

Connection Property [page 324]
Provides the SAConnection object required to execute SQL on the database server.

1.28.1 Connection Property

Provides the SAConnection object required to execute SQL on the database server.

Syntax

Visual Basic

```
Public Shared ReadOnly Property Connection As SAConnection
```

C#

```
public SAConnection Connection {get;}
```

Remarks

The connection is already established to the database server. It is not necessary to use SAConnection to establish a connection. Obtain the connection object using a statement similar to the following:

SAConnection _conn = SAServerSideConnection.Connection;

Example

The following example obtains the SAConnection object and then executes a SQL statement to create a table.

```
using Sap.Data.SQLAnywhere;
using Sap.SQLAnywhere.Server;
SAConnection _conn = SAServerSideConnection.Connection;
SACommand cmd = _conn.CreateCommand();
cmd.CommandText = "CREATE TABLE Tab( c1 int, c2 char(128), c3 smallint, c4
double, c5 numeric(30,6) )";
cmd.ExecuteNonQuery();
cmd.Dispose();
```

Related Information

SAConnection Class [page 118]

1.29 SATcpOptionsBuilder Class

Provides a simple way to create and manage the TCP options portion of connection strings used by the SAConnection object.

Syntax

Visual Basic

```
Public NotInheritable Class SATcpOptionsBuilder Inherits
SAConnectionStringBuilderBase
```

C#

```
public sealed class SATcpOptionsBuilder : SAConnectionStringBuilderBase
```

Members

All members of SATcpOptionsBuilder, including inherited members.

Constructors

Modifier and Type	Constructor	Description
public	SATcpOptionsBuilder [page 328]	Initializes an SATcpOptionsBuilder object.

Methods

Modifier and Type	Method	Description
public override string	ToString() [page 329]	Converts the SATcpOptionsBuilder object to a string representation.

Properties

Modifier and Type	Property	Description
public string	Broadcast [page 330]	Gets or sets the Broadcast option.
public string	BroadcastListener [page 330]	Gets or sets the BroadcastListener option.
public string	ClientPort [page 331]	Gets or sets the ClientPort option.

Modifier and Type	Property	Description
public string	DoBroadcast [page 331]	Gets or sets the DoBroadcast option.
public string	Host [page 331]	Gets or sets the Host option.
public string	IPV6 [page 332]	Gets or sets the IPV6 option.
public string	LDAP [page 332]	Gets or sets the LDAP option.
public string	LocalOnly [page 332]	Gets or sets the LocalOnly option.
public string	MyIP [page 333]	Gets or sets the MyIP option.
public int	ReceiveBufferSize [page 333]	Gets or sets the ReceiveBufferSize option.
public int	SendBufferSize [page 333]	Gets or sets the Send BufferSize option.
public string	ServerPort [page 334]	Gets or sets the ServerPort option.
public string	TDS [page 334]	Gets or sets the TDS option.
public int	Timeout [page 334]	Gets or sets the Timeout option.
public string	VerifyServerName [page 335]	Gets or sets the VerifyServerName option.

Inherited members from SAConnectionStringBuilderBase

Modifier and Type	Member	Description
public override bool	ContainsKey(string) [page 172]	Determines whether the SAConnectionStringBuilder object contains a specific keyword.
public string	GetKeyword(string) [page 173]	Gets the keyword for the specified SAConnectionStringBuilder property.
public bool	GetUseLongNameAsKeyword() [page 174]	Gets a boolean value that indicates whether long connection parameter names are used in the connection string.
public override ICollection	Keys [page 177]	Gets an System.Collections.ICollection that contains the keys in the SAConnectionStringBuilder.
public override bool	Remove(string) [page 174]	Removes the entry with the specified key from the SAConnectionStringBuilder instance.
public void	SetUseLongNameAsKeyword(bool) [page 175]	Sets a boolean value that indicates whether long connection parameter names are used in the connection string.
public override bool	ShouldSerialize(string) [page 176]	Indicates whether the specified key exists in this SAConnectionStringBuilder instance.
public override object	this[string keyword] [page 177]	Gets or sets the value of the connection keyword.

Modifier and Type	Member	Description
public override bool	TryGetValue(string, out object) [page 176]	Retrieves a value corresponding to the supplied key from this SAConnection-StringBuilder.

In this section:

SATcpOptionsBuilder Constructor [page 328]
Initializes an SATcpOptionsBuilder object.

ToString() Method [page 329]
Converts the SATcpOptionsBuilder object to a string representation.

Broadcast Property [page 330]
Gets or sets the Broadcast option.

BroadcastListener Property [page 330]
Gets or sets the BroadcastListener option.

ClientPort Property [page 331]
Gets or sets the ClientPort option.

DoBroadcast Property [page 331]
Gets or sets the DoBroadcast option.

Host Property [page 331]
Gets or sets the Host option.

IPV6 Property [page 332]
Gets or sets the IPV6 option.

LDAP Property [page 332]
Gets or sets the LDAP option.

LocalOnly Property [page 332]
Gets or sets the LocalOnly option.

MyIP Property [page 333]
Gets or sets the MyIP option.

ReceiveBufferSize Property [page 333]
Gets or sets the ReceiveBufferSize option.

SendBufferSize Property [page 333]
Gets or sets the Send BufferSize option.

ServerPort Property [page 334]
Gets or sets the ServerPort option.

TDS Property [page 334]
Gets or sets the TDS option.

Timeout Property [page 334]
Gets or sets the Timeout option.

VerifyServerName Property [page 335]
Gets or sets the VerifyServerName option.

Related Information

SAConnection Class [page 118]

1.29.1 SATcpOptionsBuilder Constructor

Initializes an SATcpOptionsBuilder object.

Overload list

Modifier and Type	Overload name	Description
public	SATcpOptionsBuilder() [page 328]	Initializes an SATcpOptionsBuilder object.
public	SATcpOptionsBuilder(string) [page 329]	Initializes an SATcpOptionsBuilder object.

In this section:

SATcpOptionsBuilder() Constructor [page 328]
Initializes an SATcpOptionsBuilder object.

SATcpOptionsBuilder(string) Constructor [page 329]
Initializes an SATcpOptionsBuilder object.

1.29.1.1 SATcpOptionsBuilder() Constructor

Initializes an SATcpOptionsBuilder object.

Syntax

Visual Basic

```
Public Sub SATcpOptionsBuilder ()
```

C#

```
public SATcpOptionsBuilder ()
```

Example

The following statement initializes an SATcpOptionsBuilder object.

```
SATcpOptionsBuilder options = new SATcpOptionsBuilder( );
```

1.29.1.2 SATcpOptionsBuilder(string) Constructor

Initializes an SATcpOptionsBuilder object.

Syntax

Visual Basic

```
Public Sub SATcpOptionsBuilder (ByVal options As String)
```

C#

```
public SATcpOptionsBuilder (string options)
```

Parameters

options A database server TCP connection parameter options string. For a list of connection parameters, see "Connection parameters".

Example

The following statement initializes an SATcpOptionsBuilder object.

```
SATcpOptionsBuilder options = new SATcpOptionsBuilder("localonly=yes;port=6873");
```

1.29.2 ToString() Method

Converts the SATcpOptionsBuilder object to a string representation.

Syntax

Visual Basic

```
Public Overrides Function ToString () As String
```

C#

```
public override string ToString ()
```

Returns

The options string being built.

1.29.3 Broadcast Property

Gets or sets the Broadcast option.

Syntax

Visual Basic

```
Public Property Broadcast As String
```

C#

```
public string Broadcast {get;set;}
```

1.29.4 BroadcastListener Property

Gets or sets the BroadcastListener option.

Syntax

Visual Basic

```
Public Property BroadcastListener As String
```

C#

```
public string BroadcastListener {get;set;}
```

1.29.5 ClientPort Property

Gets or sets the ClientPort option.

Syntax

Visual Basic

```
Public Property ClientPort As String
```

C#

```
public string ClientPort {get;set;}
```

1.29.6 DoBroadcast Property

Gets or sets the DoBroadcast option.

Syntax

Visual Basic

```
Public Property DoBroadcast As String
```

C#

```
public string DoBroadcast {get;set;}
```

1.29.7 Host Property

Gets or sets the Host option.

Syntax

Visual Basic

```
Public Property Host As String
```

C#

```
public string Host {get;set;}
```

1.29.8 IPV6 Property

Gets or sets the IPV6 option.

Syntax

Visual Basic

```
Public Property IPV6 As String
```

C#

```
public string IPV6 {get;set;}
```

1.29.9 LDAP Property

Gets or sets the LDAP option.

Syntax

Visual Basic

```
Public Property LDAP As String
```

C#

```
public string LDAP {get;set;}
```

1.29.10 LocalOnly Property

Gets or sets the LocalOnly option.

Syntax

Visual Basic

```
Public Property LocalOnly As String
```

C#

```
public string LocalOnly {get;set;}
```

1.29.11 MyIP Property

Gets or sets the MyIP option.

Syntax

Visual Basic

```
Public Property MyIP As String
```

C#

```
public string MyIP {get;set;}
```

1.29.12 ReceiveBufferSize Property

Gets or sets the ReceiveBufferSize option.

Syntax

Visual Basic

```
Public Property ReceiveBufferSize As Integer
```

C#

```
public int ReceiveBufferSize {get;set;}
```

1.29.13 SendBufferSize Property

Gets or sets the Send BufferSize option.

Syntax

Visual Basic

```
Public Property SendBufferSize As Integer
```

C#

```
public int SendBufferSize {get;set;}
```

1.29.14 ServerPort Property

Gets or sets the ServerPort option.

Syntax

Visual Basic

```
Public Property ServerPort As String
```

C#

```
public string ServerPort {get;set;}
```

1.29.15 TDS Property

Gets or sets the TDS option.

Syntax

Visual Basic

```
Public Property TDS As String
```

C#

```
public string TDS {get;set;}
```

1.29.16 Timeout Property

Gets or sets the Timeout option.

Syntax

Visual Basic

```
Public Property Timeout As Integer
```

C#

```
public int Timeout {get;set;}
```

1.29.17 VerifyServerName Property

Gets or sets the VerifyServerName option.

Syntax

Visual Basic

```
Public Property VerifyServerName As String
```

C#

```
public string VerifyServerName {get;set;}
```

1.30 SATransaction Class

Represents a SQL transaction.

Syntax

Visual Basic

```
Public NotInheritable Class SATransaction Inherits
System.Data.Common.DbTransaction
```

C#

```
public sealed class SATransaction : System.Data.Common.DbTransaction
```

Members

All members of SATransaction, including inherited members.

Methods

Modifier and Type	Method	Description
public override void	Commit() [page 337]	Commits the database transaction.
protected override void	Dispose(bool) [page 337]	
public override void	Rollback [page 338]	Rolls back a transaction from a pending state.
public unsafe void	Save(string) [page 339]	Creates a savepoint in the transaction that can be used to roll back a portion of the transaction, and specifies the savepoint name.

Properties

Modifier and Type	Property	Description
public new SAConnection	Connection [page 340]	The SAConnection object associated with the transaction, or a null reference (Nothing in Visual Basic) if the transaction is no longer valid.
protected override DbConnection	DbConnection [page 340]	Specifies the System.Data.Common.DbConnection object associated with the transaction.
public override System.Data.IsolationLevel	IsolationLevel [page 341]	Specifies the isolation level for this transaction.
public SAIsolationLevel	SAIsolationLevel [page 341]	Specifies the extended isolation level for this transaction.

Remarks

There is no constructor for SATransaction. To obtain an SATransaction object, use one of the BeginTransaction methods. To associate a command with a transaction, use the SACommand.Transaction property.

For more information, see "Transaction processing" and "Data access and manipulation".

In this section:

Commit() Method [page 337]
: Commits the database transaction.

Dispose(bool) Method [page 337]

Rollback Method [page 338]
: Rolls back a transaction from a pending state.

Save(string) Method [page 339]
: Creates a savepoint in the transaction that can be used to roll back a portion of the transaction, and specifies the savepoint name.

Connection Property [page 340]
: The SAConnection object associated with the transaction, or a null reference (Nothing in Visual Basic) if the transaction is no longer valid.

DbConnection Property [page 340]
: Specifies the System.Data.Common.DbConnection object associated with the transaction.

IsolationLevel Property [page 341]
: Specifies the isolation level for this transaction.

SAIsolationLevel Property [page 341]
: Specifies the extended isolation level for this transaction.

Related Information

BeginTransaction() Method [page 125]
BeginTransaction(SAIsolationLevel) Method [page 127]
Transaction Property [page 90]

1.30.1 Commit() Method

Commits the database transaction.

Syntax

Visual Basic

```
Public Overrides Sub Commit ()
```

C#

```
public override void Commit ()
```

1.30.2 Dispose(bool) Method

Syntax

Visual Basic

```
Protected Overrides Sub Dispose (ByVal disposing As Boolean)
```

C#

```
protected override void Dispose (bool disposing)
```

1.30.3 Rollback Method

Rolls back a transaction from a pending state.

Overload list

Modifier and Type	Overload name	Description
public override void	Rollback() [page 338]	Rolls back a transaction from a pending state.
public unsafe void	Rollback(string) [page 339]	Rolls back a transaction from a pending state.

In this section:

Rollback() Method [page 338]
Rolls back a transaction from a pending state.

Rollback(string) Method [page 339]
Rolls back a transaction from a pending state.

1.30.3.1 Rollback() Method

Rolls back a transaction from a pending state.

Syntax

Visual Basic

```
Public Overrides Sub Rollback ()
```

C#

```
public override void Rollback ()
```

Remarks

The transaction can only be rolled back from a pending state (after BeginTransaction has been called, but before Commit is called).

1.30.3.2 Rollback(string) Method

Rolls back a transaction from a pending state.

Syntax

Visual Basic

```
Public Sub Rollback (ByVal savePoint As String)
```

C#

```
public unsafe void Rollback (string savePoint)
```

Parameters

savePoint The name of the savepoint to roll back to.

Remarks

The transaction can only be rolled back from a pending state (after BeginTransaction has been called, but before Commit is called).

1.30.4 Save(string) Method

Creates a savepoint in the transaction that can be used to roll back a portion of the transaction, and specifies the savepoint name.

Syntax

Visual Basic

```
Public Sub Save (ByVal savePoint As String)
```

C#

```
public unsafe void Save (string savePoint)
```

Parameters

savePoint The name of the savepoint to which to roll back.

1.30.5 Connection Property

The SAConnection object associated with the transaction, or a null reference (Nothing in Visual Basic) if the transaction is no longer valid.

Syntax

Visual Basic

```
Public ReadOnly Shadows Property Connection As SAConnection
```

C#

```
public new SAConnection Connection {get;}
```

Remarks

A single application can have multiple database connections, each with zero or more transactions. This property enables you to determine the connection object associated with a particular transaction created by BeginTransaction.

1.30.6 DbConnection Property

Specifies the System.Data.Common.DbConnection object associated with the transaction.

Syntax

Visual Basic

```
Protected ReadOnly Overrides Property DbConnection As DbConnection
```

C#

```
protected override DbConnection DbConnection {get;}
```

Returns

The System.Data.Common.DbConnection object associated with the transaction.

1.30.7 IsolationLevel Property

Specifies the isolation level for this transaction.

Syntax

Visual Basic

```
Public ReadOnly Overrides Property IsolationLevel As
System.Data.IsolationLevel
```

C#

```
public override System.Data.IsolationLevel IsolationLevel {get;}
```

Remarks

The IsolationLevel for this transaction. This can be one of:

- Unspecified
- Chaos
- ReadUncommitted
- ReadCommitted
- RepeatableRead
- Serializable
- Snapshot

The default is ReadCommitted.

1.30.8 SAIsolationLevel Property

Specifies the extended isolation level for this transaction.

Syntax

Visual Basic

```
Public ReadOnly Property SAIsolationLevel As SAIsolationLevel
```

C#

```
public SAIsolationLevel SAIsolationLevel {get;}
```

Remarks

The SAIsolationLevel for this transaction. This can be one of:

- Unspecified
- Chaos
- ReadUncommitted
- ReadCommitted
- RepeatableRead
- Serializable
- Snapshot
- StatementSnapshot
- ReadOnlySnapshot

The default is ReadCommitted.

Parallel transactions are not supported. Therefore, the SAIsolationLevel applies to the entire transaction.

1.31 SAInfoMessageEventHandler(object, SAInfoMessageEventArgs) Delegate

Represents the method that handles the SAConnection.InfoMessage event of an SAConnection object.

Syntax

Visual Basic

```
Public Delegate Sub SAInfoMessageEventHandler (
    ByVal obj As Object,
    ByVal args As SAInfoMessageEventArgs
) As delegate void
```

C#

```
public delegate void SAInfoMessageEventHandler (
    object obj,
    SAInfoMessageEventArgs args
);
```

Related Information

SAConnection Class [page 118]
InfoMessage Event [page 146]

1.32 SARowsCopiedEventHandler(object, SARowsCopiedEventArgs) Delegate

Represents the method that handles the SABulkCopy.SARowsCopied event of an SABulkCopy.

Syntax

Visual Basic

```
Public Delegate Sub SARowsCopiedEventHandler (
    ByVal sender As Object,
    ByVal rowsCopiedEventArgs As SARowsCopiedEventArgs
) As delegate void
```

C#

```
public delegate void SARowsCopiedEventHandler (
    object sender,
    SARowsCopiedEventArgs rowsCopiedEventArgs
);
```

Related Information

SABulkCopy Class [page 14]

1.33 SARowUpdatedEventHandler(object, SARowUpdatedEventArgs) Delegate

Represents the method that handles the RowUpdated event of an SADataAdapter.

Syntax

Visual Basic

```
Public Delegate Sub SARowUpdatedEventHandler (
    ByVal sender As Object,
    ByVal e As SARowUpdatedEventArgs
) As delegate void
```

C#

```
public delegate void SARowUpdatedEventHandler (
    object sender,
    SARowUpdatedEventArgs e
);
```

1.34 SARowUpdatingEventHandler(object, SARowUpdatingEventArgs) Delegate

Represents the method that handles the RowUpdating event of an SADataAdapter.

Syntax

Visual Basic

```
Public Delegate Sub SARowUpdatingEventHandler (
    ByVal sender As Object,
    ByVal e As SARowUpdatingEventArgs
) As delegate void
```

C#

```
public delegate void SARowUpdatingEventHandler (
    object sender,
    SARowUpdatingEventArgs e
);
```

1.35 SABulkCopyOptions Enumeration

A bitwise flag that specifies one or more options to use with an instance of SABulkCopy.

Syntax

Visual Basic

```
Public Enum SABulkCopyOptions
```

C#

```
enum SABulkCopyOptions
```

Members

Member name	Description	Value
Default	Specifying only this value causes the default behavior to be used. By default, triggers are enabled.	0x0

Member name	Description	Value
DoNotFireTriggers	When specified, triggers are not fired. Disabling triggers requires DBA permission. Triggers are disabled for the connection at the start of WriteToServer and the value is restored at the end of the method.	0x1
KeepIdentity	When specified, the source values to be copied into an identity column are preserved. By default, new identity values are generated in the destination table.	0x2
TableLock	When specified the table is locked using the command LOCK TABLE table_name WITH HOLD IN SHARE MODE. This lock is in place until the connection is closed.	0x4
UseInternalTransaction	When specified, each batch of the bulkcopy operation is executed within a transaction. When not specified, transaction aren't used. If you indicate this option and also provide an SATransaction object to the constructor, then a System.ArgumentException occurs.	0x8

Remarks

The SABulkCopyOptions enumeration is used when you construct an SABulkCopy object to specify how the WriteToServer methods behave.

The CheckConstraints and KeepNulls options are not supported.

Related Information

SABulkCopy Class [page 14]

1.36 SADbType Enumeration

Enumerates the database server .NET database data types.

Syntax

Visual Basic

```
Public Enum SADbType
```

C#

```
enum SADbType
```

Members

Member name	Description
BigInt	Signed 64-bit integer.
Binary	Binary data, with a specified maximum length. The enumeration values Binary and VarBinary are aliases of each other.
Bit	1-bit flag.
Char	Character data, with a specified length. This type always supports Unicode characters. The types Char and VarChar are fully compatible.
Date	Date information.
DateTime	Timestamp information (date, time). The enumeration values DateTime and TimeStamp are aliases of each other.
DateTimeOffset	Timestamp information (date, time) offset.
Decimal	Exact numerical data, with a specified precision and scale. The enumeration values Decimal and Numeric are aliases of each other.
Double	Double-precision floating-point number (8 bytes).
Float	Single-precision floating-point number (4 bytes). The enumeration values Float and Real are aliases of each other.
Image	Stores binary data of arbitrary length.

Member name	Description
Integer	Unsigned 32-bit integer.
LongBinary	Binary data, with variable length.
LongNVarchar	Character data in the NCHAR character set, with variable length. This type always supports Unicode characters.
LongVarbit	Bit arrays, with variable length.
LongVarchar	Character data, with variable length. This type always supports Unicode characters.
Money	Monetary data.
NChar	Stores Unicode character data, up to 32767 characters.
NText	Stores Unicode character data of arbitrary length.
Numeric	Exact numerical data, with a specified precision and scale. The enumeration values Decimal and Numeric are aliases of each other.
NVarChar	Stores Unicode character data, up to 32767 characters.
Real	Single-precision floating-point number (4 bytes). The enumeration values Float and Real are aliases of each other.
SmallDateTime	A domain, implemented as TIMESTAMP.
SmallInt	Signed 16-bit integer.
SmallMoney	Stores monetary data that is less than one million currency units.
SysName	Stores character data of arbitrary length.
Text	Stores character data of arbitrary length.
Time	Time information.
TimeStamp	Timestamp information (date, time). The enumeration values DateTime and TimeStamp are aliases of each other.
TimeStampWithTimeZone	Timestamp information (date, time, time zone). The enumeration values DateTime and TimeStamp are aliases of each other.
TinyInt	Unsigned 8-bit integer.
UniqueIdentifier	Universally Unique Identifier (UUID/GUID).
UniqueIdentifierStr	A domain, implemented as CHAR(36). UniqueIdentifierStr is used for remote data access when mapping Microsoft SQL Server uniqueidentifier columns.

Member name	Description
UnsignedBigInt	Unsigned 64-bit integer.
UnsignedInt	Unsigned 32-bit integer.
UnsignedSmallInt	Unsigned 16-bit integer.
VarBinary	Binary data, with a specified maximum length. The enumeration values *Binary* and *VarBinary* are aliases of each other.
VarBit	Bit arrays that are from 1 to 32767 bits in length.
VarChar	Character data, with a specified maximum length. This type always supports Unicode characters. The types Char and VarChar are fully compatible.
Xml	XML data. This type stores character data of arbitrary length, and is used to store XML documents.

Remarks

The table below lists which .NET types are compatible with each SADbType. In the case of integral types, table columns can always be set using smaller integer types, but can also be set using larger types as long as the actual value is within the range of the type.

SADbType	Compatible .NET type	C# built-in type	Visual Basic built-in type
BigInt	System.Int64	long	Long
Binary, *VarBinary*	System.Byte[], or System.Guid if size is 16	byte[]	Byte()
Bit	System.Boolean	bool	Boolean
Char, *VarChar*	System.String	String	String
Date	System.DateTime	DateTime (no built-in type)	Date
DateTime, *TimeStamp*	System.DateTime	DateTime (no built-in type)	DateTime
DateTimeOffset	System.DateTimeOffset	DateTimeOffset (no built-in type)	DateTimeOffset
Decimal, *Numeric*	System.String	decimal	Decimal
Double	System.Double	double	Double
Float, *Real*	System.Single	float	Single
Image	System.Byte[]	byte[]	Byte()
Integer	System.Int32	int	Integer
LongBinary	System.Byte[]	byte[]	Byte()

SADbType	Compatible .NET type	C# built-in type	Visual Basic built-in type
LongNVarChar	System.String	String	String
LongVarChar	System.String	String	String
Money	System.String	decimal	Decimal
NChar	System.String	String	String
NText	System.String	String	String
Numeric	System.String	decimal	Decimal
NVarChar	System.String	String	String
SmallDateTime	System.DateTime	DateTime (no built-in type)	Date
SmallInt	System.Int16	short	Short
SmallMoney	System.String	decimal	Decimal
SysName	System.String	String	String
Text	System.String	String	String
Time	System.TimeSpan	TimeSpan (no built-in type)	TimeSpan (no built-in type)
TimeStamp	System.DateTime	DateTime (no built-in type)	Date
TimeStampWithTimeZone	System.DateTimeOffset	DateTimeOffset (no built-in type)	DateTimeOffset
TinyInt	System.Byte	byte	Byte
UniqueIdentifier	System.Guid	Guid (no built-in type)	Guid (no built-in type)
UniqueIdentifierStr	System.String	String	String
UnsignedBigInt	System.UInt64	ulong	UInt64 (no built-in type)
UnsignedInt	System.UInt32	uint	UInt64 (no built-in type)
UnsignedSmallInt	System.UInt16	ushort	UInt64 (no built-in type)
Xml	System.Xml	String	String

Binary columns of length 16 are fully compatible with the UniqueIdentifier type.

Related Information

GetFieldType(int) Method [page 220]
GetDataTypeName(int) Method [page 215]

1.37 SAIsolationLevel Enumeration

Specifies database server isolation levels.

Syntax

Visual Basic

```
Public Enum SAIsolationLevel
```

C#

```
enum SAIsolationLevel
```

Members

Member name	Description
Chaos	This isolation level is unsupported. If you call SAConnection.BeginTransaction with this isolation level, then an exception is thrown. For more information, see "BeginTransaction method".
ReadCommitted	Sets the behavior to be equivalent to isolation level 1. This isolation level prevents dirty reads, but allows non-repeatable reads and phantom rows. For more information, see "Isolation levels and consistency".
ReadUncommitted	Sets the behavior to be equivalent to isolation level 0. This isolation level allows dirty reads, non-repeatable reads, and phantom rows. For more information, see "Isolation levels and consistency".
RepeatableRead	Sets the behavior to be equivalent to isolation level 2. This isolation level prevents dirty reads and guarantees repeatable reads. However, it allows phantom rows. For more information, see "Isolation levels and consistency".
Serializable	Sets the behavior to be equivalent to isolation level 3. This isolation level prevents dirty reads, guarantees repeatable reads, and prevents phantom rows. For more information, see "Isolation levels and consistency".

Member name	Description
Snapshot	Uses a snapshot of committed data from the time when the first row is read, inserted, updated, or deleted by the transaction. This isolation level uses a snapshot of committed data from the time when the first row is read or updated by the transaction. For more information, see "Isolation levels and consistency".
Unspecified	This isolation level is unsupported. If you call SAConnection.BeginTransaction with this isolation level, then an exception is thrown. For more information, see "BeginTransaction method".
ReadOnlySnapshot	For read-only statements, use a snapshot of committed data from the time when the first row is read from the database. Non-repeatable reads and phantom rows can occur within a transaction, but not within a single statement. For updatable statements, use the isolation level specified by the updatable_statement_isolation option (can be one of 0 (the default), 1, 2, or 3). For more information, see "Isolation levels and consistency".
StatementSnapshot	Use a snapshot of committed data from the time when the first row is read by the statement. Each statement within the transaction sees a snapshot of data from a different time. For each statement, use a snapshot of committed data from the time when the first row is read from the database. Non-repeatable reads and phantom rows can occur within a transaction, but not within a single statement. For more information, see "Isolation levels and consistency".

Remarks

This class augments the System.Data.IsolationLevel class.

The .NET Data Provider supports all database server isolation levels, including the snapshot isolation levels. To use snapshot isolation, specify one of SAIsolationLevel.Snapshot, SAIsolationLevel.ReadOnlySnapshot, or SAIsolationLevel.StatementSnapshot as the parameter to BeginTransaction. BeginTransaction has been overloaded so it can take either an IsolationLevel or an SAIsolationLevel. The values in the two enumerations are the same, except for ReadOnlySnapshot and StatementSnapshot which exist only in SAIsolationLevel. There is a new property in SATransaction called SAIsolationLevel that gets the SAIsolationLevel.

For more information, see "Snapshot isolation".

1.38 SAMessageType Enumeration

Identifies the type of message.

Syntax

Visual Basic

```
Public Enum SAMessageType
```

C#

```
enum SAMessageType
```

Members

Member name	Description	Value
Action	Message of type ACTION.	2
Info	Message of type INFO.	0
Status	Message of type STATUS.	3
Warning	Message of type WARNING.	1

Remarks

This can be one of: Action, Info, Status, or Warning.

Important Disclaimers and Legal Information

Hyperlinks

Some links are classified by an icon and/or a mouseover text. These links provide additional information.
About the icons:

- Links with the icon  : You are entering a Web site that is not hosted by SAP. By using such links, you agree (unless expressly stated otherwise in your agreements with SAP) to this:
  - The content of the linked-to site is not SAP documentation. You may not infer any product claims against SAP based on this information.
  - SAP does not agree or disagree with the content on the linked-to site, nor does SAP warrant the availability and correctness. SAP shall not be liable for any damages caused by the use of such content unless damages have been caused by SAP's gross negligence or willful misconduct.
- Links with the icon  : You are leaving the documentation for that particular SAP product or service and are entering a SAP-hosted Web site. By using such links, you agree that (unless expressly stated otherwise in your agreements with SAP) you may not infer any product claims against SAP based on this information.

Videos Hosted on External Platforms

Some videos may point to third-party video hosting platforms. SAP cannot guarantee the future availability of videos stored on these platforms. Furthermore, any advertisements or other content hosted on these platforms (for example, suggested videos or by navigating to other videos hosted on the same site), are not within the control or responsibility of SAP.

Beta and Other Experimental Features

Experimental features are not part of the officially delivered scope that SAP guarantees for future releases. This means that experimental features may be changed by SAP at any time for any reason without notice. Experimental features are not for productive use. You may not demonstrate, test, examine, evaluate or otherwise use the experimental features in a live operating environment or with data that has not been sufficiently backed up.
The purpose of experimental features is to get feedback early on, allowing customers and partners to influence the future product accordingly. By providing your feedback (e.g. in the SAP Community), you accept that intellectual property rights of the contributions or derivative works shall remain the exclusive property of SAP.

Example Code

Any software coding and/or code snippets are examples. They are not for productive use. The example code is only intended to better explain and visualize the syntax and phrasing rules. SAP does not warrant the correctness and completeness of the example code. SAP shall not be liable for errors or damages caused by the use of example code unless damages have been caused by SAP's gross negligence or willful misconduct.

Bias-Free Language

SAP supports a culture of diversity and inclusion. Whenever possible, we use unbiased language in our documentation to refer to people of all cultures, ethnicities, genders, and abilities.

www.sap.com/contactsap

THE BEST RUN SAP