Zen v15

Distributed Tuning Objects Guide

Developing Applications Using the Distributed Tuning Objects

Distributed Tuning Objects Guide
2021 年 10 月

目次

このドキュメントについて

このドキュメントでは、Distributed Tuning Objects を使用したアプリケーション開発について説明します。

このドキュメントの読者

このドキュメントは、Zen に精通し、Distributed Tuning Objects を使用して管理レベルのアプリケーションを開発したいユーザー向けにデザインされています。

表記上の規則

特段の記述がない限り、コマンド構文、コード、およびコード例では、以下の表記が使用されます。

大文字と小文字の区別	通常、コマンドと予約語は、大文字で表記されます。本書で別途記述がない限り、これらの項目は大文字、小文字、あるいはその両方を使って入力できます。たとえば、MYPROG、myprog、または MYprog と入力することができます。
太字	太字で表示される単語には次のようなものがあります。メニュー名、ダイアログ ボックス名、コマンド、オプション、ボタン、ステートメントなど。
固定幅フォント	固定幅フォントは、コマンド構文など、ユーザーが入力するテキストに使われます。
[ ]	省略可能な情報には、［*log_name*］のように、角かっこが使用されます。角かっこで囲まれていない情報は必ず指定する必要があります。
|	縦棒は、[*file name* | *@file name*] のように、入力する情報の選択肢を表します。
< >	＜＞ は、/D=<5|6|7> のように、必須項目に対する選択肢を表します。
変数	*file name* のように斜体で表されている語は、適切な値に置き換える必要のある変数です。
…	［*parameter...*］のように、情報の後に省略記号が続く場合は、その情報を繰り返し使用できます。
::=	記号 ::= は、ある項目が別の項目用語で定義されていることを意味します。たとえば、a::=b は、項目 *a* が *b* で定義されていることを意味します。

Distributed Tuning Objects の概要

1

Zen 管理 API への COM インターフェイス

以下のトピックでは、Zen Distributed Tuning Objects を構成する機能について説明します。

- 「DTO とは」
- 「DTO オブジェクト モデルとオブジェクトの関係」
- 「DTO の使用を始める」
- 「DTO オブジェクトの概要」
- 「DTO コレクションを使った作業」
- 「DTO サンプルの参照場所」

以下のトピックへ直接移動して、Zen で DTO を使用する方法の詳細を参照することもできます。

- 「DTO セッションの確立」
- 「DTO を使用した Zen サーバーの設定」
- 「DTO を使用した Zen サーバーの監視」
- 「DTO を使用したカタログと辞書の作成および管理」
- 「Distributed Tuning Objects 列挙」

DTO とは

Distributed Tuning Objects（以降、このドキュメントでは DTO と表記します）は、Zen Distributed Tuning Interface（以降、このドキュメントでは DTI と表記します）の COM ラッパーです。DTO は DTI をカプセル化するオブジェクトのコレクションです。また、DTO にはデータベース エンジンの起動および停止など DTI よりも優れた機能がいくつかあります。

DTO を使用すれば、開発者はカスタマイズされたサーバー管理ツールやインターフェイスを多岐にわたり迅速かつ簡単に開発することができます。DTO の強力な機能や柔軟性は、データベースの作成やパフォーマンスのチューニングおよびメタデータの管理など、データベース管理やデータベース定義のタスク全般に幅広く利用することができます。

DTO はデュアル インターフェイスとして処理中のサーバーに実装されます。開発者は、OLE オートメーション コントローラーを使用することも、あるいは以下に挙げる多くのプログラム言語などを使用して COM クライアントを作成することもできます。

- Microsoft Visual Basic
- Microsoft Active Server Pages（ASP）
- Microsoft Visual C++
- Embarcadero Delphi
- Embarcadero C++ Builder

DTO オブジェクト モデルとオブジェクトの関係

このドキュメントでは、DTO クラスを以下の機能のカテゴリ別に分類しています。

接続

データベース エンジンの動作を構成および監視できるようにするには、ユーザーは最初にそのエンジンに接続する必要があります。このカテゴリでは、データベース エンジンへの接続および接続の切断に必要な機能を提供します。

DtoSession オブジェクトがデータベース エンジンへの接続を管理します。

監視と診断

このカテゴリでは、Zen サーバーとクライアントを監視する機能および診断情報を提供します。

DtoMonitor オブジェクトとその従属オブジェクトによって、サーバーの監視機能および診断情報を提供します。また、DtoEngineInformation オブジェクトを使って DtoSession からエンジン情報を直接取得することもできます。

構成

このカテゴリでは、ユーザーが Zen エンジンやクライアントを構成することができます。DtoCategories コレクションとその従属オブジェクトによって、この機能を提供します。

また、DtoLicenseMgr オブジェクトを使用すれば製品ライセンスの追加と削除も行えます。

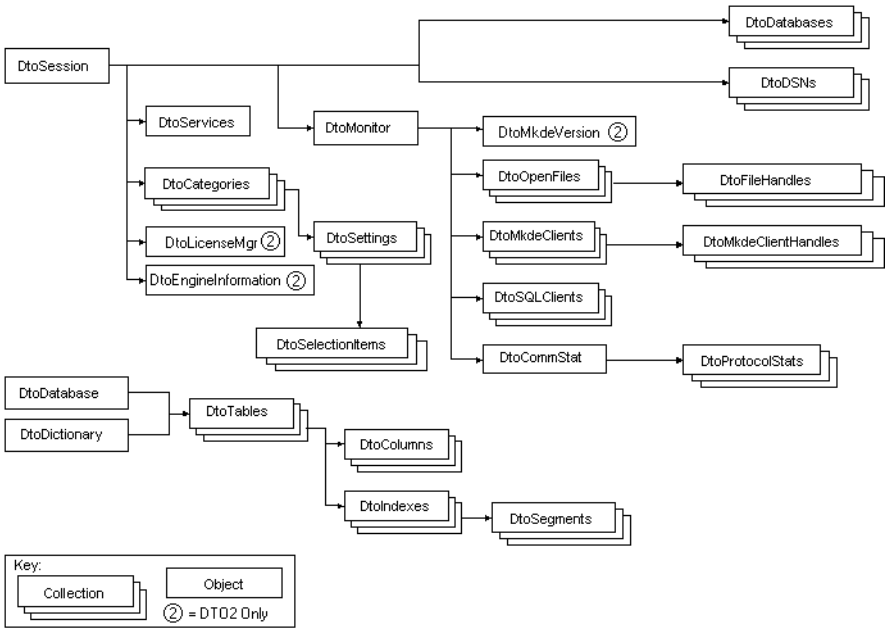
カタログと辞書

このカテゴリでは、グループ化された機能によって、ユーザーはデータベース、データ辞書を新規作成することができます。また、テーブル、列およびインデックスの定義や削除も行えます。

DtoDictionary クラスとその従属オブジェクトによって、カタログ機能を提供します。

DTO オブジェクト ツリー

多くの DTO オブジェクトがその他の DTO オブジェクトのプロパティとして公開されています。この関係によって、開発者はオートメーション コントローラーを使用するプログラミングを簡略化する論理的なツリー形式の構造を使用することができます。多くのオブジェクトは、プロパティやメソッドにアクセスする場合に使うドット表記を使用して参照することができます。



DTO オブジェクト ツリーには 3 つの主要なブランチがあり、設定、監視、およびカタログ用のオブジェクトが論理的にグループ化されています。

DTO バージョン

次の表は、2 つの異なるバージョンの DTO の使用に関する情報を示しています。

Item	DTO2		DTO1
	x86	x64	x86
DLL 名	DTO2.DLL	w64DTO2.DLL	DTO.DLL
ライブラリ	DTOLib2		DTOLib
DtoSession のプログラム ID	DTO.DtoSession.2		DTO.DtoSession.1
DtoDatabase のプログラム ID	DTO.DtoDatabase.2		DTO.DtoDatabase.1
DtoDictionary のプログラム ID	DTO.DtoDictionary.2		DTO.DtoDictionary.1

DTO2

Zen V8 SDK では新しい DLL がリリースされました。この DLL には新しいオブジェクトや既存のオブジェクトへの新しいプロパティが追加されています。下位バージョンとの互換性を保つために、以前の DTO.DLL に互換性を追加するのではなく、新しい DLL が作成されました。両方の DLL がインストールされ Zen SDK で登録されるので、どちらの DLL を使用しても DTO アプリケーションを開発できます。新しい DLL を使用するためにアプリケーションを再コンパイルできない場合は、以前の DTO.DLL を使用する必要があります。ただし、以前の

DTO.DLL を使用していると、新しいオブジェクトや既存のオブジェクトに追加された新しいプロパティを使用することができません。DTO2.DLL は、.NET Framework を含む 32 ビット開発環境をサポートします。

W64DTO2

Zen v11 SP1 SDK では、.NET Framework を含む 64 ビット開発のための 64 ビット サポートが追加されました。

64 ビット アプリケーションで DTO を利用するためには、作成した 64 ビット アプリケーションと一緒に 64 ビット サーバーまたはクライアントをインストールする必要があります。前の表で示されているとおり、64 ビット サーバーまたはクライアントをインストールすると、64 ビット バージョン（W64DTO2.DLL）がインストールされ、32 ビット サーバーまたはクライアントをインストールすると、32 ビット バージョン（DTO2.DLL または DTO.DLL）がインストールされます。

アプリケーションと DLL の相互作用に関する理解

アプリケーションと DLL がどのように相互作用するかをより良く理解するために、次のようなシナリオを検討します。

次の 3 つの DLL を持っていると仮定します。

- DTO.DLL
- DTO2.DLL
- W64DTO2.DLL

そして、次のようなアプリケーションの実行可能ファイルがあるとします。

- ANYCPU.EXE
- X86.EXE
- X64.EXE

以下の表は、アプリケーションの実行可能ファイルと DLL を一緒に、32 ビット マシンおよび 64 ビット マシン上で実行してみた場合の結果を示しています。

表 1　32 ビット マシンのプロセス

アプリケーション実行可能ファイル	実行形態	DTO.DLL	DTO2.DLL	W64DTO2.DLL
ANYCPU.EXE	32 ビット プロセス	読み込み	読み込み	BadImageFormatException
X86.EXE	32 ビット プロセス	読み込み	読み込み	BadImageFormatException
X64.EXE	BadImageFormatException			

表 2　64 ビット マシンのプロセス

アプリケーション実行可能ファイル	実行形態	DTO.DLL	DTO2.DLL	W64DTO2.DLL
ANYCPU.EXE	64 ビット プロセス	BadImageFormatException	BadImageFormatException	読み込み
X86.EXE	32 ビット プロセス	読み込み	読み込み	BadImageFormatException
X64.EXE	64 ビット プロセス	BadImageFormatException	BadImageFormatException	読み込み

DTO の使用を始める

このセクションでは、Visual Basic、Active Server Pages（ASP）および Delphi DTO を使用するためのセットアップ方法を説明します。

- 「Visual Basic」
- 「Active Server Pages」
- 「Delphi」

Visual Basic

DTO は デュアル インターフェイス COM オブジェクトのライブラリなので、Visual Basic ではこれらのオブジェクトを使って作業する 2 つの方法があります。Active Server Pages を使用している場合、2 番目の方法を使う必要があります。第一にお勧めする方法は、プロジェクトにタイプ ライブラリを追加する方法です。この方法を使用すれば、VB で型のチェックを行うことができ、オブジクトの作成や関数のパラメーターに対して便利なドロップダウン オプション（Intellisense 機能）を開発者に提供することができます。

もう 1 つの方法は、CreateObject 関数を使用する方法です。これは実行時にオブジェクトを作成するため、型のチェックと Intellisense 機能がありません。

プロジェクトへの DTO の追加

Visual Basic プロジェクトに Distributed Tuning Library を追加するには、以下の手順に従います。

1 ［**プロジェクト**］メニューの［参照設定］をクリックします。

2 参照可能なエントリをスクロールして、［Pervasive Distributed Tuning Library 1.0］または［Pervasive Distributed Tuning Library 2.0］チェック ボックスを選択します。これらの違いについては、「DTO2」を参照してください。

3 ［OK］をクリックします。

これで VB では DTO に含まれるすべてのオブジェクトを認識することができます。すべてのオブジェクトが参照可能になります。使用可能なオブジェクトを表示するには、以下の手順に従います。

1 ［**表示**］メニューの［**オブジェクト ブラウザー**］を選択します。この代替手順として、F2 キーを押すこともできます。

2 使用可能なライブラリのリストから、DTO バージョン 1 の場合は［DTOLib］を、DTO バージョン 2 の場合は［DTOLib2］を選択します。これらの違いについては、「DTO2」を参照してください。

CreateObject 関数の使用

ASP でオブジェクトのインスタンスを作成する場合は、この方法を使用する必要があります。CreateObject の構文は次のようになります。

```
Dim my_session as Object
' DTO バージョン 2 の場合
Set my_session = CreateObject("DTO.DtoSession.2")
' 旧バージョンの DTO アプリケーションとの互換性を保つ場合は、
' DTO バージョン 1 を使用する
Set my_session = CreateObject("DTO.DtoSession.1")
```

ほとんどの DTO オブジェクトは、後でセッション オブジェクトから取得することができます。

Active Server Pages

ASP で DTO を使用するために必要な初期化は特にありません。ただし、次の点に注意してください。

デフォルトで、ASP は呼び出し間の状態の情報を保存しません。Microsoft IIS の組み込みオブジェクトである Session オブジェクトを使用して、オブジェクト参照と変数の状態を保存する必要があります。

たとえば、DTO バージョン 2 の DtoSession オブジェクトを初期化するには以下のように記述します。

```
Set Session("my_session") = Server.CreateObject("DTO.DtoSession.2")
```

Delphi

Delphi で COM オブジェクトを使用するには、2 つの方法があります。Visual Basic と同様、第一にお勧めするセットアップ方法は Delphi プロジェクトにタイプ ライブラリをインポートする方法です。

もう 1 つの方法では、CreateOleObject 関数を使って直接 COM インターフェイスを呼び出すことができます。この関数は Automation オブジェクトの単一のインスタンスをインスタンス化します。

Delphi プロジェクトへの DTO タイプ ライブラリのインポート

タイプ ライブラリをインポートし、必要な Pascal 宣言を生成するには、以下の手順に従います。

1 ［**プロジェクト**］メニューの［**タイプ ライブラリの取り込み**］を選択します。

2 ［**タイプ ライブラリの取り込み**］ダイアログ ボックスでは、システムに登録されているタイプ ライブラリを表示します。［Pervasive Distributed Tuning Library 1.0］または［Pervasive Distributed Tuning Library 2.0］を選択します。

3 ［ユニットディレクトリ名］フィールドに Pascal ユニットの場所を入力し、［**ユニットの作成**］をクリックします。DTOLib_TLB.pas ファイル（DTO バージョン 1 を使用時）または DTOLib2_TLB.pas ファイル（DTO バージョン 2 を使用時）が作成され、プロジェクトに取り込まれます。

4 次に、以下のコードを組み込んで、生成した Pascal ユニットをメイン ファイルに取り込みます。

```
uses DTOLib2_TLB; // Dto Version 2
uses DTOLib_TLB;  // Dto Version 1
```

Pascal 宣言を使用する例

```
var
    Result:DTOResult;
    Session:DTOSession;
    MySettings:DTOSettings;
    MyCategories:DTOCategories;
    MyCategory:DTOCategory;
    i:integer;
begin
    Session:=CoDTOSession.Create;
    Result:=Session.Connect
            ('ServerName','UserName','Password');
    MyCategories:=Session.Categories;
    for i:=1 to MyCategories.Count do
    MyCategory:=MyCategories.Item[i];
end;
```

COM 直接呼び出しを使用する例

```
Example:

var
  Session, Categories, Category: Variant;
  I: Integer;
begin
  Session := CreateOleObject('DTO.DtoSession');
```

```
  Session.Connect('ServerName','UserName','Password');
  Categories := Session.Categories;
  for I := 1 to Categories.Count do
  Category := Categories.Item[I];
end;
```

DTO オブジェクトの概要

Distributed Tuning Objects のリファレンスは、機能別に 4 つの章に分かれています。このセクションでは、各章のオブジェクトの一覧を表示します。

接続関連オブジェクト

DtoSession

DtoSession オブジェクトは DTO の中核的なオブジェクトです。アプリケーションから Zen サーバーへ接続する場合は、DtoSession オブジェクトを介して行います。特定のデータベース サーバーでセッションが要求された場合、DTO アプリケーションでは DtoSession オブジェクトを作成し、Connect メソッドを使用します。

設定関連オブジェクト

DtoCategory

DtoCategory オブジェクトと DtoCategories コレクションはデータベース エンジンの設定をグループ化しており、ユーザーはより詳細な設定を対象にして DtoSetting オブジェクトにアクセスすることができます。

DtoSetting

DtoSetting オブジェクトと DtoSettings コレクションは、データベース エンジン、通信マネージャーおよびローカル リクエスター コンポーネントの特定の設定を公開しており、ユーザーはこれらの設定を変更することができます。通常、各カテゴリは設定のコレクションを公開します。

DtoSelectionItem

DtoSelectionItem オブジェクトと DtoSelectionItems コレクションには選択が可能な設定に対する全項目が含まれます。DtoSetting.AllPossibleSelections では、指定した設定に対する可能なすべての値のコレクションを返します。

DtoServices

DtoServices オブジェクトを使用すれば、ユーザーは Zen データベース サービスを開始、停止したりサービスの現在の状態を照会したりすることができます。

DtoLicenseMgr

DtoLicenseMgr（DTO2）オブジェクトを使用すれば、ライセンスを追加、削除できるほかに、製品情報を見ることができます。

監視関連オブジェクト

DtoMonitor

DtoMonitor オブジェクトを使用すれば、ユーザーはデータベース エンジンやその他関連するサービスに関するステータス情報をリアルタイムで取得することができます。

また、DtoMonitor オブジェクトでは、ファイル ハンドル、開いているファイルおよびライセンスの現在値、ピーク値、最大値などのリソース使用状況に関する情報も公開しています。ピーク値とは、そのエンジンを最後に再起動してから現在に至るまでの最大値です。

DtoOpenFile

DtoOpenFile オブジェクトと DtoOpenFiles コレクションにはアクティブ ファイルに関する情報が含まれます。これを使用すれば、ユーザーは、開いているファイル数、それらのファイルを開いているユーザーおよびその他の関連情報を調べてファイル アクセスを監視することができます。

DtoFileHandle

DtoFileHandle オブジェクト DtoFileHandles コレクションは、ユーザー名またはエージェント ID、接続番号、タスク番号、サイト、ネットワーク アドレス、オープン モード、レコード ロック タイプ、待ち状態、トランザクション状態を公開します。

DtoMkdeClient

DtoMkdeClient オブジェクトと DtoMkdeClients コレクションは、アクティブ クライアントに関する情報を公開します。特定のクライアントの場合、そのクライアントのアクティブ セッションがあるかどうかを調べ、アクティブ セッションがあった場合はそのセッションに関するデータを取得することができます。また、オプションでそのクライアントを終了することもできます。

DtoMkdeClientHandle

DtoMkdeClientHandle オブジェクトと DtoMkdeClientHandles コレクションは、各ファイルの名前や関連情報が含まれるハンドル情報を公開します。

DtoMkdeVersion

DtoMkdeVersion（DTO2）では、Zen エンジンのメジャー バージョンとマイナー バージョン、ビルド番号およびターゲット オペレーティング システムを公開します。

DtoEngineInformation

DtoEngineInformation（DTO2）では、メジャー バージョンとマイナー バージョン、DTI API バージョンおよびサーバーとクライアントのその他の情報を公開します。

DtoSqlClient

DtoSqlClient オブジェクトと DtoSqlClients コレクションは、アクティブ SQL ユーザーの数や一覧および各クライアントに関する詳細情報など、SQL アクティブ ユーザーに関する情報を公開します。

DtoCommStat

DtoCommStat オブジェクトは、通信統計情報を公開します。適切な場合、現在値、ピーク値および最大値を照会することができます。

DtoProtocolStat

DtoProtocolStat オブジェクトと DtoProtocolStats コレクションは、サーバーで実行している各ネットワーク プロトコルに関する情報を公開します。

データベースおよび辞書関連オブジェクト

DtoDatabase

DtoDatabase オブジェクトと DtoDatabases コレクションは、データベース名、データベース フラグ、セキュリティ、テーブル定義などのデータベース カタログ情報の管理を担当します。

DtoDSN

DtoDSN オブジェクトと DtoDSNs コレクションは、サーバーにある Zen DSN を表します。これらを使用して、新しい DSN を作成したり、既存の Zen ODBC DSN を管理したりすることができます。

DtoDictionary（非推奨）

DtoDictionary オブジェクトは、辞書ファイルに関連するすべての操作のルート オブジェクトです。このオブジェクトを使用して、辞書のオープン、辞書の作成、テーブル情報の取得、テーブルの追加または削除を行うことができます。

Tables コレクションにアクセスするには、DtoDatabase オブジェクトを使用する方法をお勧めします。

DtoTable

DtoTable オブジェクトと DtoTables コレクションは、テーブル名、列およびインデックスなどのテーブル情報の管理を担当します。

DtoColumn

DtoColumn オブジェクトと DtoColumns コレクションは、列に関する情報の管理を担当します。

DtoIndex

DtoIndex オブジェクトと DtoIndexes コレクションは、テーブルのインデックス定義を公開します。

DtoSegment

DtoSegment オブジェクトと DtoSegments コレクションには、テーブル内で指定したインデックスのセグメントに関する情報があります。

DTO コレクションを使った作業

コレクションとは、ほかの複数オブジェクトを含むオブジェクトです。

コレクションのインスタンス化

Visual Basic

Set キーワードを使用して変数をコレクション オブジェクトに設定します。

```
Dim result as DtoResult
Dim my_session as New DtoSession
Dim my_databases as DtoDatabases
result = my_session.Connect("myserver", "username", "pw")

if not(result=0) Then
'Connect メソッドのエラー処理
end if

' オブジェクトまたはコレクションが値を返すタイプの場合は
'Set を使用する

Set my_databases = my_session.Databases
```

ASP

ASP を使用する場合、通常は DtoSession と DtoDictionary という 2 つのオブジェクトのインスタンスのみを直接作成する必要があります。Active Server Pages でオブジェクトのインスタンスを作成する場合、IIS の組み込みオブジェクトである Server オブジェクトの CreateObject 構文を使用します。

```
Set my_session = Server.CreateObject("DTO.DtoSession.2")
```

メモ 複数の HTTP 呼び出しの間でオブジェクトを存在させたい場合、以下のコード例のように ISS の組み込みオブジェクトである Session オブジェクトを使用してオブジェクトの状態を保持する必要があります。

```
Set Session("my_session") = Server.CreateObject("DTO.DtoSession.2")
```

ただし、データベース、DSN、テーブル、列、インデックスまたはセグメントなどのオブジェクトを新規作成する場合は、この同じ構文を使用します。各オブジェクトのプログラム ID は、前述の例で示す規則に従います。

その他のコレクションについては、前述の Visual Basic の Set 構文を使用することができます。

コレクション内のループ

Visual Basic

Visual Basic でコレクション内をループするには、For ループの使用とカウンターの使用という 2 つの方法があります。

For/Next ステートメントを使ってコレクション内をループする Visual Basic の構文を以下に示します。

```
'Categories コレクションを取得する
Dim my_categories as DtoCategories
Dim category as DtoCategory
Set my_categories = my_session.Categories

' コレクション内をループする
```

```
For Each category In my_categories
   settings = category.Settings
Next
```

カウンターを使ってコレクション内をループする Visual Basic の構文を以下に示します。

```
Dim column as DtoColumn
Dim table as DtoTable
Set table = dictionary.Tables("Billing")
Dim i as long
for i=1 to table.Columns.count
  set column=table.Columns(i)
  ' ここで操作を実行
next i
```

ASP

Zen の設定のカテゴリ一覧を表示する ASP のサンプル コードを以下に示します。

```
<%
Set Session("my_session") = Server.CreateObject("DTO.DtoSession.2")
result = Session("my_session").Connect("myserver", "username", "pw")
'Connect メソッドのエラー処理は表示しない

Set my_categories = Session("my_session").Categories
' 次に、HTML の箇条書きリスト (<UL></UL>) 内のカテゴリをループして出力する
%>

<ul>
<% For each category in my_categories %>

<% ' 等号 (=) は HTML ストリーム内の変数を表示します。 %>
<% ' これは、VBScript 組み込みの Response.Write() %>
<% ' メソッドのショートカットです。%>

<li><%=cat.CategoryId> - <%=cat.Name%></li>

<% Next %>
</ul>
```

メンバー数の取得

Visual Basic

Count プロパティを使用してコレクション内のオブジェクトの数を調べます。

```
Dim num_items as Integer
Dim my_session as New DtoSession
Dim result as DtoResult
result = my_session.Connect("myserver", "username", "pw")
'Connect メソッドのエラー処理は表示しない
Set my_databases = my_session.Databases
num_items = my_databases.Count
```

ASP

Count プロパティを使用してコレクション内のオブジェクトの数を調べます。

```
<%
Set Session("my_session") = Server.CreateObject("DTO.DtoSession.2")
```

```
result = Session("my_session").Connect("myserver", "username", "pw")
'Connect メソッドのエラー処理は表示しない
Set my_databases = Session("my_session").Databases
num_items = my_databases.Count
' 出力を HTML ストリームに書き込む
Response.Write("<p>Number of databases = " & num_items)
%>
```

特定のメンバーの取得

Visual Basic と ASP

Item プロパティを使用して、コレクション内の特定のメンバー オブジェクトを取得します。

コレクション内のメンバー（要素）には序数を使ってアクセスすることができます。コレクションによっては、要素の一意のプロパティ（名前など）を使用して各要素にアクセスすることができます。すべての序数は 1 から始まります。

Item プロパティはコレクション オブジェクトのデフォルト プロパティなので、次の 2 つのステートメントは同一です。

```
Collection.Item(index)
Collection(index)
```

DTO サンプルの参照場所

Zen SDK には Visual Basic による完全な DTO サンプルがあります。このサンプルは、SDK のサンプルとヘッダーファイルをインストールした次の場所にあります。

SDK のコンポーネント、コード スニペット、およびサンプルは、Actian Web サイトから入手できます。

Zen ファイルのデフォルトの保存場所については、『*Getting Started with Zen*』の「ファイルはどこにインストールされますか？」を参照してください。

DTO アクセス方法に関するその他の開発者情報およびリソースについて、Actian Web サイトもご覧ください。

DTO セッションの確立

2

Zen 管理機能を行う最初の手順

このトピックでは、Distributed Tuning Objects を使用したセッションの確立について説明します。

- 「DtoSession オブジェクト」

DtoSession オブジェクト

DtoSession オブジェクトは、ほとんどの DTO 操作のルート オブジェクトです。Zen データベース エンジンへの接続を管理します。

プロパティ

Connected	Session オブジェクトが Zen エンジンに接続するかどうかを示すブール値を返します。 True = 接続されています False = 接続されていません
Error	最後のメソッド呼び出しのエラーを返します。メソッド呼び出しの結果を渡し、エラーを説明する dtoResult 文字列を返します。 エラー コードの一覧については 「DtoResult」を参照してください。
ServerName	DtoSession オブジェクトのサーバ名を取得または設定します。
UserName	オブジェクトのユーザー名を設定します。
Password	セッションのパスワードを設定します。

コレクション

「DtoCategories コレクション」

「DtoDatabases コレクション」

「DtoDSNs コレクション」

オブジェクト

「DtoMonitor オブジェクト」

「DtoServices オブジェクト」

「DtoLicenseMgr オブジェクト」

「DtoEngineInformation オブジェクト」

メソッド

「Connect メソッド」

「Disconnect メソッド」

「GetSetting メソッド」

備考

DtoSession オブジェクトは、辞書を除くすべての操作の起点です。DtoSession を使用して、サーバーへの接続、カテゴリや設定などの設定情報の取得、データベースや DSN の調査、Zen の使用情報の監視を行います。

DtoSession を使用するには、まずオブジェクトのインスタンスを作成し、Connect メソッドを使ってセッション オブジェクトのサーバーを指定します。

セッションの接続に使用するユーザー名とパスワードはそのマシン用のみです。これは、Zen データベースに対して認証されるわけではありません。

メモ ASP を使って、または Visual Basic の CreateObject メソッドを使ってこのオブジェクトのインスタンスを作成する場合、DtoSession のプログラム ID は "DTO.DtoSession.2"（DTO バージョン 2）または "DTO.DtoSession.1"（DTO バージョン 1）になります。これら 2 つのバージョンの違いについては、「DTO2」を参照してください。

例

```
' セッション オブジェクトのインスタンスを作成する
Dim my_session as New DtoSession

' サーバーに接続する
result = my_session.Connect("myserver", "username", "password")

'Error プロパティを使って接続が正常かどうか確認する
if Not (result = Dto_Success)
Then Msgbox"Could not connect to the server.Error was "+ my_session.Error(result)

' セッション オブジェクトを使って、Category および Database
' コレクションを取得する

Dim my_categories as DtoCategories
Dim my_databases as DtoDatabases
Set my_categories = my_session.Categories
Set my_databases = my_session.Databases
```

関連項目

「DtoCategories コレクション」

「DtoServices オブジェクト」

「DtoMonitor オブジェクト」

「DtoDatabases コレクション」

「DtoDSNs コレクション」

メソッドの詳細

Connect メソッド

サーバーへの接続を開きます。

構文

```
Dim result as DtoResult
result = Object.Connect([server], [username], [password])
```

引数

Object	DtoSession オブジェクト
server	（省略可能）接続するサーバーの名前。省略した場合、ローカル サーバーへの接続を試行します。また、先に ServerName プロパティを設定しておき、このパラメーターを指定しないで Connect メソッドを呼び出すこともできます。

username	（省略可能）サーバーのユーザー名。先に UserName プロパティを設定しておき、このパラメーターを指定しないで Connect メソッドを呼び出すこともできます。
password	（省略可能）ユーザーのパスワード。先に Password プロパティを設定しておき、このパラメーターを指定しないで Connect メソッドを呼び出すこともできます。

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。Error プロパティを使って結果の説明を取得します。

備考

サーバーへの接続はさまざまな理由で失敗することがあるので、このメソッドの戻り値をチェックしてプログラムから適切な処置を行ってください。

ユーザー名とパスワードを指定しない場合、guest としてログインを試みます。guest としてのログインが成功した場合、いくつかの機能が使用できません。

セッションが現在接続されているかどうかを確認するために isConnected プロパティをチェックしてください。

例

```
Dim result as DtoResult
Dim my_session as New DtoSession
result = my_session.Connect("myserver", "smook", "1234")

Dim result as DtoResult
Dim my_session as New DtoSession
my_session.UserName="smook"
my_session.Password="1234"
my_session.ServerName="myserver"
result = my_session.Connect
```

Disconnect メソッド

サーバへの接続を終了します。

構文

result = Object.**Disconnect**

引数

Object	DtoSession オブジェクト

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。Error プロパティを使って結果の説明を取得します。

備考

セッション オブジェクトを使用して別のサーバー、あるいは既存のアプリケーションへ接続する前に、接続しているすべてのサーバーに対してこのメソッドを呼び出す必要があります。

例

```
Dim result as DtoResult
Dim my_session as New DtoSession
result = my_session.Connect("myserver", "username", "pw")
'
' ここで操作を実行
'
result = my_session.Disconnect
```

GetSetting メソッド

設定 ID を使用して DtoSetting オブジェクトを取得します。

構文

```
Set my_setting = Object.GetSetting(setting_id)
```

引数

Object	DtoSession オブジェクト
setting_id	有効な設定 ID。DtoSetting オブジェクトにはそれぞれ一意に識別する SettingID プロパティがあります。特定のカテゴリのすべての設定を取得する場合は、「DtoCategory オブジェクト」の Settings プロパティを使用します。このメソッドは、取得する特定の設定が既にわかっていて、あらかじめそれらの設定 ID（setting_id）を保存している場合に利用できます。

戻り値

my_setting	DtoSetting オブジェクト。指定した設定が見つからなかった場合は、NULL が返されます。

備考

このメソッドは、最初にカテゴリを取得して DtoSettings コレクションから検索することなく、指定した設定を取得する場合に利用できます。

例

```
Dim oSession As New DtoSession
Dim Result As dtoResult
Result = oSession.Connect("localhost", "", "")

Dim oSetting As DtoSetting
Dim settingFileversion As Integer
settingFileversion = 97
Set oSetting = oSession.GetSetting(settingFileversion)

If oSetting Is Nothing Then
  MsgBox "Invalid setting"
```

```
Else
  Dim new_selections As New DtoSelectionItems
  '0 = 9.5 '1 = 9.0 '2 = 8.x '3 = 7.x '4 = 6.x '5 = 13.0
  ' これらはファイル形式の値です

  new_selections.Add oSetting.AllPossibleSelections.GetByID(0)
  oSetting.Value = new_selections
End If
```

DTO を使用した Zen サーバーの設定

3

Zen Control Center 機能を実行するための COM インターフェイス

この章では、Distributed Tuning Objects の設定グループを構成するオブジェクトについて説明します。

DtoCategories コレクション

このオブジェクトは、特定の DtoSession オブジェクトに使用できるすべての設定カテゴリを表す DtoCategory オブジェクトのコレクションです。

プロパティ

Count	コレクション内のメンバー数を返します。
Item	コレクションの特定のメンバーを返します。

メソッド

なし

備考

このコレクションでは、1 から始まる序数を渡して個別の項目を取得することができます。

Count プロパティを使用してコレクション内のメンバー数を調べます。

```
num_categories = my_categories.Count
```

Item プロパティを使用して、コレクションの 1 から始まるインデックスを取得します。

```
Set first_item = my_categories(1)
```

例

```
' セッション オブジェクトのインスタンスを作成し、サーバーに接続する
Dim my_session as new DtoSession
Dim result as DtoResult
result = my_session.Connect("myserver", "username", "password")

'Categories コレクションを取得する
Dim my_categories as DtoCategories
Set my_categories = my_session.Categories
```

関連項目

「DtoCategory オブジェクト」

「DtoSession オブジェクト」

DtoCategory オブジェクト

このオブジェクトを使用すれば、DtoCategories コレクションの特定のカテゴリで操作を実行することができます。

プロパティ

CategoryID	DtoCategory の一意のカテゴリ ID を返します。
Name	カテゴリの名前を返します。
Session	カテゴリのセッションを返します。

コレクション

「DtoSettings コレクション」

メソッド

なし

備考

カテゴリの設定リストを取得するには、Settings プロパティを使って DtoSettings コレクションを返します。その後、その中に含まれる DtoSetting オブジェクトを使って特定の設定に関する情報を取得することができます。

例

```
' セッション オブジェクトのインスタンスを作成し、サーバーに接続する
Dim my_session as new DtoSession
Dim result as DtoResult
Dim category as DtoCategory
Dim my_categories as DtoCategories
Dim settings as DtoSettings
result = my_session.Connect("myserver", "username", "password")

'Categories コレクションを取得する
Set my_categories = my_session.Categories

' コレクション内をループする
For Each category In my_categories
  Set settings = category.Settings
Next
```

関連項目

「DtoCategories コレクション」

「DtoSession オブジェクト」

DtoLicenseMgr オブジェクト

「DTO2」のみ：このオブジェクトを使用すれば、製品ライセンスの認証と認証解除、ライセンス検証操作の開始、および製品情報の XML 文字列の取得が行えます。

プロパティ

なし

コレクション

なし

メソッド

「AddLicense メソッド」

「DeleteLicense メソッド」

「GetProductInfo メソッド」

「ValidateLicenses メソッド」

備考

Session オブジェクトから取得します。

例

```
' セッション オブジェクトのインスタンスを作成し、サーバーに接続する
Dim my_session as new DtoSession
Dim result as DtoResult
Dim my_licmgr as DtoLicenseMgr

result = my_session.Connect("myserver", "username", "password")

'License Manager オブジェクトを取得する
Set my_licmgr = my_session.LicenseMgr

' ライセンスを追加する

result = my_licmgr.AddLicense("ERXVD3U4ZS9KR94QPDHV5BN2")
```

関連項目

「DtoSession オブジェクト」

メソッドの詳細

AddLicense メソッド

ライセンスを認証します。

構文

result = *LicenseManager*.**AddLicense**(*License*)

引数

LicenseManager	DtoLicenseMgr オブジェクト
License	DtoSession オブジェクトを使って現在接続しているエンジンに対して認証する、有効なライセンス キー。

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

例

```
' セッション オブジェクトのインスタンスを作成し、サーバーに接続する
Dim my_session as new DtoSession
Dim result as DtoResult
Dim my_licmgr as DtoLicenseMgr

result = my_session.Connect("myserver", "username", "password")

'License Manager オブジェクトを取得する
Set my_licmgr = my_session.LicenseMgr

' ライセンスを追加する

result = my_licmgr.AddLicense("ERXVD3U4ZS9KR94QPDHV5BN2")
```

DeleteLicense メソッド

ライセンスを認証解除します。

構文

result = *LicenseManager*.**DeleteLicense**(*License*)

引数

LicenseManager	DtoLicenseMgr オブジェクト
License	DtoSession オブジェクトを使って現在接続しているエンジンから認証解除する、有効なライセンス キー。

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

備考

なし

例

```
' セッション オブジェクトのインスタンスを作成し、サーバーに接続する
Dim my_session as new DtoSession
Dim result as DtoResult
Dim my_licmgr as DtoLicenseMgr

result = my_session.Connect("myserver", "username", "password")

'License Manager オブジェクトを取得する
Set my_licmgr = my_session.LicenseMgr

' ライセンスを削除する

result = my_licmgr.DeleteLicense("ERXVD3U4ZS9KR94QPDHV5BN2")
```

GetProductInfo メソッド

License Manager で検出されたすべての Zen 製品の一覧を取得します。

構文

result = *LicenseManager*.**GetProductInfo**

引数

LicenseManager	DtoLicenseMgr オブジェクト

戻り値

result	製品の一覧を XML 形式の文字列で返します。

備考

XML 形式の文字列の詳細については、『*Distributed Tuning Interface Guide*』で PvGetProductsInfo() の「備考」を参照してください。

例

```
' セッション オブジェクトのインスタンスを作成し、サーバーに接続する
Dim session As New DtoSession
Set session = New DtoSession
Dim result As dtoResult
result = session.Connect("server", "user", "password")

If result <> Dto_Success Then
  MsgBox "Error on connect."& CStr(result)
  Exit Sub
End If
Dim xmlstring As String
xmlstring = session.LicenseMgr.GetProductInfo
RichTextBox1.TextRTF = xmlstring
```

ValidateLicenses メソッド

セッション接続によって示されるコンピューターのすべてのキーに関する有効性のチェックを開始します。

構文

result = LicenseManager.ValidateLicenses

引数

LicenseManager	DtoLicenseMgr オブジェクト

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

備考

ValidateLicenses は、検証チェックの**要求**から生じた結果のみを返します。キーの状態に関する情報は何も**返しません**。キーの状態に関する情報も含め、製品情報の XML 文字列を取得するには、別に 「GetProductInfo メソッド」を呼び出す必要があります。

例

```
' セッション オブジェクトのインスタンスを作成し、サーバーに接続する
Dim my_session as new DtoSession
Dim result as DtoResult
Dim my_licmgr as DtoLicenseMgr

result = my_session.Connect("myserver", "username", "password")

'License Manager オブジェクトを取得する
Set my_licmgr = my_session.LicenseMgr

' すべてのキーの検証チェックを開始する
result = my_licmgr.ValidateLicenses
```

DtoSettings コレクション

このコレクションには、特定の DtoCategory オブジェクトのすべての設定を表す DtoSetting オブジェクトが含まれます。

プロパティ

Count	コレクション内のメンバー数を返します。
Item	コレクションの特定のメンバーを返します。

メソッド

なし

備考

Count プロパティを使用してコレクション内のメンバー数を調べます。

このコレクションでは、1 から始まる序数または設定名を含むバリアントを渡して個別の項目を取得することができます。

メモ また、「DtoSession オブジェクト」の GetSetting を使って個別の設定を取得することもできます。これにより、カテゴリと設定内のループを節約します。

例

```
Dim my_categories as DtoCategories
Set my_categories = my_session.Categories

Dim my_settings as DtoSettings
Dim first_setting as DtoSetting
Set my_settings = my_categories.Settings
Set first_setting = my_settings(1)
```

関連項目

「DtoCategories コレクション」

「DtoCategory オブジェクト」

DtoSetting オブジェクト

このオブジェクトは、環境設定を表します。

プロパティ

AllPossibleSelections	設定タイプが単一選択または複数選択であるとき、選択可能なすべての項目を表す「DtoSelectionItems コレクション」または「DtoSelectionItem オブジェクト」を返します。 このプロパティは、Type プロパティの設定が "dtoSingleSelection" または "dtoMultiSelection" の場合にのみ有効です。これらはそれぞれ、TypeNames プロパティの "Single Selection" および "Multiple Selection" に相当します。
Category	この設定に関連する DtoCategory オブジェクトを返します。
DefaultValue	設定のデフォルト値を返します。 戻り値は Type プロパティの設定に基づくバリアントです。 Type プロパティが "Single selection" の場合は DtoSelectionItem オブジェクトを返します。 Type プロパティが "Multiple selection" の場合は DtoSelectionItems コレクションを返します。
Factor	設定のファクター値を返します。 たとえば、多くの設定が Zen ではバイト単位で保存されていますが、ユーザーは設定を変更するときに値をキロバイトで入力する場合があります。 ある設定の Value プロパティで 16384 が返され、Factor プロパティが 1024 を返した場合、プログラムでは 16384 を 1024 で除算してユーザーに 16 を返します。その後、FactorString プロパティを照会し、新しい単位を取得します。この場合はキロバイトになります。 Value プロパティを設定する前に、Factor プロパティでユーザーが指定した値を乗算する必要があります。
FactorString	Factor プロパティに調整する Value プロパティの単位を返します。たとえば、UnitString プロパティが "byte(s)" を返した場合、FactorString プロパティは "kbyte(s)" を返し、Factor プロパティは "1024" を返します。
FalseString	ブール型の設定に対して False 値を返します。 このプロパティは、ブール型の設定の場合にのみ有効です。Type プロパティを使用して、設定がブール型かどうかを調べます。
Help	ある設定に関連するヘルプ テキストを返します。
IsClient	その設定が Zen クライアントに対して有効、あるいは Enterprise Server に対して有効かどうかを示すブール値を返します。 True = クライアント False = サーバー
Max	Long 型の設定の最大値を返します。 このプロパティは、Long 型の設定の場合にのみ有効です。Type プロパティを使用して、設定が Long 型かどうかを調べます。 このプロパティで負数が返される場合、以下のように解釈されます。 /* 最大有効メモリまたはディスク サイズ */P_MAX_MEM_DISK_SIZE -129 /* 使用可能なディスク スペースによって制限される最大サイズ */P_MAX_LIMITED_BY_DISK -2 /* 使用可能なメモリによって制限される最大サイズ */P_MAX_LIMITED_BY_MEMORY -1

Min	Long 型の設定の最小値を返します。 このプロパティは、Long 型の設定の場合にのみ有効です。Type プロパティを使用して、設定が Long 型かどうかを調べます。 このプロパティは、有効でない場合 -1 が返されます。
Name	設定の名前を返します。
Rank	設定のランクを返します。ランクのグループ設定は、上級ユーザーにのみ適用するかどうかを条件とします。 0 = 標準 1 = 上級
Session	このオブジェクトに関連するセッションを返します。
SettingID	設定の固有の識別子を返します。
TrueString	ブール型の設定に対して True 値を返します。 このプロパティは、ブール型の設定の場合にのみ有効です。Type プロパティを使用して、設定がブール型かどうかを調べます。
Type	設定タイプ（「設定タイプ」列挙）を返します。 0 = Boolean（ブール型） 1 = Long（Long 型） 2 = String（文字列） 3 = Single selection（単一選択） 4 = Multiple selection（複数選択）
TypeName	設定のタイプを以下の文字列で返します。 Boolean Long String Single selection Multiple selection
UnitString	Long 型の設定の単位を返します。 たとえば、seconds、bytes などです。 ユーザーがより使いやすい値の範囲を利用できるよう Value プロパティを調整する場合は、Factor プロパティと FactorString プロパティを使用します。
Value	設定の値を取得または設定します。 戻り値は Type プロパティの設定に基づくバリアントです。 Type プロパティが "Single selection" の場合は DtoSelectionItem オブジェクトを返します。 Type プロパティが "Multiple selection" の場合は DtoSelectionItems コレクションを返します。 Long 型の設定に対してこのプロパティを設定する場合は、Min および Max プロパティで照会して、設定する値がその設定の制限範囲内であるかどうかをチェックしてください。

コレクション

「DtoSelectionItems コレクション」

メソッド

なし

備考

Type プロパティを使用して設定のタイプを見つけます。次のタイプによって異なることに注意してください。

- `TrueString` および `FalseString` プロパティはブール型の設定（0）にのみ適用されます。
- `Factor`、`FactorString`、`Max`、`Min` および `UnitString` プロパティは Long 型の設定（1）にのみ適用されます。

例

```
Set my_settings = my_category.Settings
Set first_setting = my_settings(1)
```

関連項目

「DtoCategories コレクション」

「DtoCategory オブジェクト」

DtoSelectionItems コレクション

選択タイプの設定で、選択可能な値を表す DtoSelectionItem オブジェクトのコレクション。

プロパティ

Count	コレクション内のメンバー数を返します。
Item	コレクションの特定のメンバーを返します。

メソッド

「Add メソッド」

「GetById メソッド」

「Remove メソッド」

備考

「DtoSetting オブジェクト」の AllPossibleSelections プロパティは、DtoSelectionItems コレクションを返します。

Count プロパティを使用してコレクション内のメンバー数を調べます。

例

```
Set first_setting = my_settings(1)

type = first_setting.Type

' 選択タイプの設定の場合のみこれを呼び出す
' 「設定タイプ」列挙を参照

if (type = dtoSingleSel) OR (type = dtoMultiSel)
  Set all_the_selections = first_setting.AllPossibleSelections
```

関連項目

「DtoCategories コレクション」

「DtoSetting オブジェクト」

メソッドの詳細

Add メソッド

DtoSelectionItems コレクションに項目を追加します。

構文

result = *Collection*.**Add**(*Object*)

引数

Collection	オブジェクトを追加する DtoSelectionItems コレクション。
Object	追加する DtoSelectionItem オブジェクト。

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

備考

このメソッドには DtoSelectionItem タイプのパラメーターが必要です。このため、コレクションにオブジェクトを追加する前に、まずオブジェクトのインスタンスを作成してそのプロパティを設定する必要があります。

例

```
Dim Result As dtoResult
Dim Session As New DtoSession
Result = Session.Connect("nik-ntws", "", "")
Dim my_setting As DtoSetting
Dim SetID As Long
SetID = 26
Set my_setting = Session.GetSetting(SetID)
   If my_setting Is Nothing Then
        MsgBox " Setting is wrong"
        Exit Sub
   End If

' 新しい値の割り当てを開始する
'ItemID 1 で項目を追加する
new_selections.Add my_setting.AllPossibleSelections.Item(1)
'TCP

my_setting.Value = new_selections
```

GetById メソッド

指定した ID で DtoSelectionItems コレクションから DtoSelectionItem オブジェクトを返します。

構文

my_selection_item = Collection.**GetById**(*id*)

引数

Collection	DtoSelectionItems コレクション
id	コレクションから取得する項目の ID。特定の選択項目の ID は DtoSelectionItem オブジェクトの ItemId プロパティを使って取得することができます。

戻り値

my_selection_item	*id* に相当する 「DtoSelectionItem オブジェクト」

例

```
Dim Result As DtoResult
Dim Session As New DtoSession
Result = Session.Connect("nik-ntws", "", "")
Dim my_setting As DtoSetting
Dim SetID As Long
SetID = 26
Set my_setting = Session.GetSetting(SetID)
   If my_setting Is Nothing Then
        MsgBox " Setting is wrong"
        Exit Sub
   End If

Dim new_selections As New DtoSelectionItems
new_selections.Add my_setting.AllPossibleSelections.GetByID(3)  'Microsoft TCP/IP

my_setting.Value = new_selections
```

Remove メソッド

DtoSelectionItems コレクションから項目を削除します。

構文

*result = Collection.***Remove**(*item*)

引数

Collection	DtoSelectionItems コレクション
item	コレクションから削除する項目の（1 から始まる）インデックスまたは選択項目の名前を含むバリアント。

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

備考

このメソッドには DtoSelectionItem タイプのパラメーターが必要です。

例

```
Dim Result As DtoResult
Dim Session As New DtoSession
Result = Session.Connect("nik-ntws", "", "")
```

```
Dim my_setting As DtoSetting
Dim SetID As Long
SetID = 26
Set my_setting = Session.GetSetting(SetID)
   If my_setting Is Nothing Then
        MsgBox " Setting is wrong"
        Exit Sub
   End If

Dim new_selections As New DtoSelectionItems
new_selections.Add my_setting.AllPossibleSelections.GetByID(3)  ''Microsoft TCP/IP
new_selections.Remove(1)

my_setting.Value = new_selections
```

DtoSelectionItem オブジェクト

選択タイプの設定で選択可能な値を表すオブジェクトです。

プロパティ

ItemID	選択項目の一意の ID を返します。
Setting	この選択項目を適用する設定を返します。
String	選択項目の値を返します。

メソッド

なし

備考

「DtoSetting オブジェクト」の AllPossibleSelections プロパティは DtoSelectionItems コレクションを返します。

例

```
Set first_setting = my_settings.Item(1)

Dim type as dtoSettingType
type = first_setting.Type

' 選択タイプの設定の場合のみこれを呼び出す
' 「設定タイプ」列挙を参照

if (type = dtoSingleSel) OR (type = dtoMultiSel) then
Dim all_the_selections as DtoSelectionItems
Dim selection as DtoSelectionitem
Set all_the_selections = first_setting.AllPossibleSelections

Dim String_text as String
For each selection in all_the_selections
  String_text = selection.String
Next
```

関連項目

「DtoSelectionItems コレクション」

「DtoSetting オブジェクト」

DtoServices オブジェクト

このオブジェクトは DtoService オブジェクトのコレクションです。これはサーバーで起動している Zen サービスを表します。

プロパティ

Status	サービスの状態を返します。状態を取得する以下のサービスを渡す必要があります。 dtoServiceTransactional dtoServiceRelational dtoServiceIDS
StatusString	現在の状態を表す文字列を返します。

メソッド

「RestartAllServices メソッド」

「StartRelational メソッド」

「StartTransactional メソッド」

「StopRelational メソッド」

「StopTransactional メソッド」

「StartDXAgent メソッド」

「StartDXReplication メソッド」

「StopDXAgent メソッド」

「StopDXReplication メソッド」

備考

DtoServices のメソッドは、DtoSession オブジェクトを使って接続するコンピューターで起動している Zen エンジンのサービスを制御します。これらのメソッドはすべて 「DtoResult」列挙を返します。

このオブジェクトによって、Windows プラットフォームで起動している Zen エンジンのサービスを開始および停止することができます。また、Status または StatusString プロパティを使って Zen サービスの現在の状態を照会することができます。

DtoServices オブジェクトに関するセキュリティ情報

- このオブジェクトは、DtoSession オブジェクトと同じユーザー名とパスワードを使って Windows サーバーに接続することができます。
- Microsoft の Internet Information Service（IIS）によってホストされる Web アプリケーションからこのオブジェクトのメソッドを使用する場合は、IIS によって DTO が IIS サービスと同じプロセス内で実行できるようにするために、DTO の Web アプリケーションがあるディレクトリ上でプロパティを設定する必要があります。これを行わない場合、IIS サービスの現在の状態は取得できますが、Startxx メソッドや Stopxx メソッドを使用したときに DTO エラー 431（アクセス拒否）が返されます。DtoServices オブジェクトのメソッドに必要な IIS フォルダー プロパティを設定するには、DTO Web アプリケーションが置かれているフォルダー上で以下の手順を実行してください。IIS の設定の詳細については、Microsoft IIS のドキュメントを参照してください。

▶▶ DTO の Web アプリケーションからサービスを開始 / 停止できるように IIS を構成するには

1 ［**スタート**］メニューから［**設定**］を選択し、［**コントロール パネル**］をポイントします。

2 ［**管理ツール**］をダブルクリックします。

3 ［**インターネット サービス マネージャー**］をダブルクリックします。

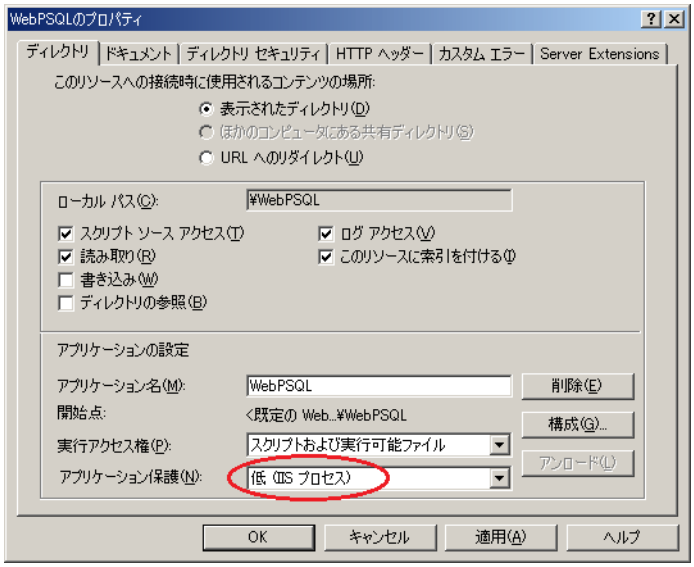
4 DTO の ASP アプリケーションがあるフォルダーを参照します。

5 左ペインのフォルダーを右クリックして［**プロパティ**］を選択します。

6 ［**ディレクトリ**］タブをクリックします。

7 図 1 で示すように、［**アプリケーション保護**］フィールドで "**低（IIS プロセス）**" を指定します。

図 1 DtoServices メソッドで必要な IIS のディレクトリ プロパティ



例

```
' この例では、サーバーに接続し、
'Zen サービスを再開する

Dim my_session as new DtoSession
Dim my_services as DtoServices
Dim result as DtoResult

result = my_session.Connect("myserver", "username", "password")
Set my_services = my_session.Services
result = my_services.RestartAllServices

' この例では、サーバーに接続し、
'DataExchange (DX) エージェント サービスと DX レプリケーション
' サービスを再開する

Dim my_session as new DtoSession
Dim my_services as DtoServices
Dim result1 as DtoResult
Dim result2 as DtoResult

result = my_session.Connect("myserver", "username", "password")
Set my_services = my_session.Services
result1 = my_services.StartDXReplication
result2 = my_services.StartDXAgent
```

関連項目

「DtoSession オブジェクト」

「DtoSetting オブジェクト」

メソッドの詳細

RestartAllServices メソッド

トランザクショナル サービス、リレーショナル サービス、DataExchange（DX）エージェントおよび DX レプリケーション サービスを停止して再開始します。

構文

```
result = Services.RestartAllServices
```

引数

Services	DtoServices オブジェクト

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

StartRelational メソッド

リレーショナル サービスを開始します。Zen v14 現在、これは StartTransactional メソッドと同じです。

構文

```
result = Services.StartRelational
```

引数

Services	DtoServices オブジェクト

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

StartTransactional メソッド

Btrieve トランザクショナル サービスを開始します。Zen v14 現在、これは StartRelational メソッドと同じです。

構文

```
result = Services.StartTransactional
```

引数

Services	DtoServices オブジェクト

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

StopRelational メソッド

リレーショナル エンジン サービスを停止します。Zen v14 現在、これは StopTransactional メソッドと同じです。

構文

```
result = Services.StopRelational
```

引数

Services	DtoServices オブジェクト

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

StopTransactional メソッド

Btrieve トランザクショナル エンジン サービスを停止します。Zen v14 現在、これは StopRelational メソッドと同じです。

構文

```
result = Services.StopTransactional
```

引数

Services	DtoServices オブジェクト

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

StartDXAgent メソッド

DataExchange（DX）エージェント サービスを開始します。DX エージェントは、レプリケーションでの重大な障害を検出し、管理者に電子メールで通知するコンポーネントです。詳細については、DataExchange ドキュメントを参照してください。

DX エージェント サービスは、DX レプリケーション サービスを開始する前でも開始することができますが、その場合はレプリケーション サービスが停止していることを通知するメッセージがエージェントから返されます。レプリケーション サービスがまだ実行していないため、これは正常な動作です。

構文

```
result = Services.StartDXAgent
```

引数

Services	DtoServices オブジェクト

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

StartDXReplication メソッド

DataExchange（DX）レプリケーション サービス（レプリケーション エンジン）を開始します。レプリケーション エンジンでは、DataExchange レプリケーション ネットワーク内で、どれか 1 つの Zen データベースの変更を捕捉し、それをほかのデータベースと共有することができます。詳細については、DataExchange ドキュメントを参照してください。

レプリケーション サービスを開始すると、トランザクショナル サービスやリレーショナル サービスも開始します。

構文

```
result = Services.StartDXReplication
```

引数

Services	DtoServices オブジェクト

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

StopDXAgent メソッド

DataExchange（DX）エージェント サービスを停止します。DX エージェントは、レプリケーションでの重大な障害を検出し、管理者に電子メールで通知するコンポーネントです。詳細については、DataExchange ドキュメントを参照してください。

構文

```
result = Services.StopDXAgent
```

引数

Services	DtoServices オブジェクト

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

StopDXReplication メソッド

DataExchange（DX）レプリケーション エンジンを停止します。レプリケーション エンジンでは、DataExchange レプリケーション ネットワーク内で、どれか 1 つの Zen データベースの変更を捕捉し、それをほかのデータベースと共有することができます。詳細については、DataExchange ドキュメントを参照してください。

構文

```
result = Services.StopDXReplication
```

引数

Services	DtoServices オブジェクト

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

DTO を使用した Zen サーバーの監視

4

Zen の監視機能を実行する COM インターフェイス

この章では、Distributed Tuning Objects の監視グループを構成するオブジェクトについて説明します。

DtoMonitor オブジェクト

このオブジェクトは Zen サーバーに関する使用情報を提供します。このオブジェクトは、その他すべての監視操作のルート オブジェクトです。

プロパティ

CurClients	セッションの現在のクライアント数を返します。
CurFilesInUse	セッションで現在使用中のファイル数を返します。
CurHandlesInUse	セッションで現在使用中のハンドル数を返します。
CurLicensesInUse	セッションで現在使用中のライセンス数を返します。
CurLicDataInUseMB	使用データの現在値をメガバイト（MB）単位で返します。「DTO2」のみ。
CurLocksInUse	セッションで現在使用中のロック数を返します。
CurSessionCountInUse	現在使用中のセッション数（現在のセッション数）を返します。「DTO2」のみ。
CurThreads	セッションのスレッド数を返します。
CurTransInUse	セッションで現在開いているトランザクションの数を返します。
EngineUpTimeSecs	データベース エンジンの実行時間を秒数で返します。「DTO2」のみ。
MaxClients	セッションの最大クライアント数を返します。
MaxFiles	セッションの最大ファイル数を返します。
MaxHandles	セッションの最大ハンドル数を返します。
MaxLicenses	セッションのユーザー ライセンス数を返します。
MaxLicDataMB	使用データの許容最大サイズ（使用データの制限値）をメガバイト（MB）で返します。16 進値 0xFFFFFFFF は無制限を意味します。「DTO2」のみ。
MaxSessionCount	ライセンスによって許可される最大セッション数（セッション数の制限値）を返します。16 進値 0xFFFFFFFF は無制限を意味します。「DTO2」のみ。
MaxThreads	セッションの最大スレッド数を返します。
MaxTrans	セッションの最大トランザクション数を返します。
PeakClients	セッションのクライアント数のピーク値を返します。
PeakFilesInUse	セッションで使用中のファイル数のピーク値を返します。
PeakHandlesInUse	セッションで使用中のハンドル数のピーク値を返します。
PeakLicensesInUse	セッションで使用中のライセンス数のピーク値を返します。
PeakLicDataInUseMB	同時に使用されているデータの最大値をメガバイト（MB）単位で返します。「DTO2」のみ。
PeakLocksInUse	セッションで使用中のロック数のピーク値を返します。
PeakSessionCountInUse	同時に使用されているセッションの最大数を返します。「DTO2」のみ。

PeakThreads	セッションのスレッド数のピーク値を返します。
PeakTransInUse	セッションで使用中のトランザクション数のピーク値を返します。

コレクション

「DtoMkdeClients コレクション」

「DtoOpenFiles コレクション」

「DtoSqlClients コレクション」

オブジェクト

「DtoCommStat オブジェクト」

「DtoSession オブジェクト」

「DtoMkdeVersion オブジェクト」

メソッド

なし

例

```
' セッションのインスタンスを作成し、サーバーに接続する
Dim my_session as new DtoSession
Dim result as DtoResult
result = my_session.Connect("myserver", "username", "password")

'DtoSession から DtoMonitor オブジェクトを取得する
Dim session_monitor as DtoMonitor
Set session_monitor = my_session.Monitor

' 現在使用中のファイルを取得する
Dim current_files as long
current_files = session_monitor.CurFilesInUse
```

関連項目

「DtoOpenFiles コレクション」

「DtoMkdeClients コレクション」

DtoOpenFiles コレクション

現在開いているファイルを表す DtoOpenFile オブジェクトのコレクションです。

プロパティ

Count	コレクション内のメンバー数を返します。
Item	DtoOpenFiles コレクションの特定のメンバーを返します。

メソッド

なし

備考

Count プロパティを使用してコレクション内のメンバー数を調べます。

「DtoMonitor オブジェクト」のプロパティを使って DtoOpenFiles コレクションを取得することができます。

例

```
' セッションのインスタンスを作成して接続する
Dim my_session as new DtoSession
Dim result as DtoResult
result = my_session.Connect("myserver", "username", "password")

' セッションからモニター オブジェクトを取得する
Dim my_monitor as DtoMonitor
Set my_monitor = my_session.Monitor

' モニターから開いているファイルを取得する
Dim my_openfiles as DtoOpenFiles
Set my_openfiles = my_monitor.OpenFiles
```

関連項目

「DtoMonitor オブジェクト」

DtoOpenFile オブジェクト

開いているファイルを表すオブジェクトです。

プロパティ

ActiveCursors	開いているファイルのアクティブ カーソル数を返します。
AFLIndex	開いているファイルの AFL Index を返します。
ContinuousOps	開いているファイルが Continuous オペレーションを使用しているかどうかを返します。
FileName	開いているファイルに関連付けられているファイル名を返します。
IsLocked	開いているファイルがロックされているかどうかを返します。 0 = ロックされていません 1 = ロックされています
IsReadOnly	開いているファイルが読み取り専用かどうかを返します。 0 = 読み取り専用ではありません 1 = 読み取り専用です
IsTrans	開いているファイルがトランザクション状態かどうかを返します。 0 = No 1 = Yes
Monitor	開いているファイルに関連付けられている DtoMonitor オブジェクトを返します。
OpenMode	開いているファイルのオープン モードを返します。
OpenModeName	OpenMode のテキスト バージョンを返します。
PageSize	開いているファイルのページ サイズを返します。
PhysFileSizeKB	ファイルの物理サイズをキロバイト（KB）で返します。「DTO2」のみ。
ReferentialIntegrity	開いているファイルに参照整合性が設定されているかどうかを返します。 0 = No 1 = Yes
TimeStamp	開いているファイルのタイム スタンプを返します。
TTSFlag	今後の使用に備えて予約されています。

メソッド

なし

コレクション

「DtoFileHandles コレクション」

備考

このオブジェクトは現在開いているファイルを表します。開いているすべてのファイルのコレクションの場合は、「DtoOpenFiles コレクション」を使用します。

例

```
Dim my_session as new DtoSession
Dim is_read_only as Boolean
Dim my_monitor as DtoMonitor
Dim my_openfiles as DtoOpenFiles
Dim first_file as DtoOpenFile
Dim result as DtoResult

result = my_session.Connect("myserver", "username", "password")
Set my_monitor = my_session.Monitor
Set my_openfiles = my_monitor.OpenFiles
Set first_file = my_openfiles(1)
is_read_only = first_file.IsReadOnly
```

関連項目

「DtoOpenFiles コレクション」

「DtoMonitor オブジェクト」

DtoFileHandles コレクション

開いているファイルのすべてのファイル ハンドルを表す DtoFileHandle オブジェクトのコレクションです。

プロパティ

Count	コレクション内のメンバー数を返します。
Item	DtoFileHandles コレクションの特定のメンバーを返します。

メソッド

なし

備考

Count プロパティを使用してコレクション内のメンバー数を調べます。

例

```
Dim my_session as new DtoSession
Dim my_monitor as DtoMonitor
Dim my_openfiles as DtoOpenFiles
Dim first_file as DtoOpenFile
Dim my_handles as DtoFileHandles
Dim result as DtoResult

result = my_session.Connect("myserver", "username", "password")
Set my_monitor = my_session.Monitor
Set my_openfiles = my_monitor.OpenFiles
Set first_file = my_openfiles.Item(1)
Set my_handles = first_file.FileHandles
```

関連項目

「DtoFileHandle オブジェクト」

「DtoOpenFile オブジェクト」

「DtoMonitor オブジェクト」

DtoFileHandle オブジェクト

開いているファイルのファイル ハンドルを表すオブジェクトです。

プロパティ

ClientIndex	ファイル ハンドルのインデックスを返します。
IsLocked	ファイル ハンドルがロックされているかどうかを返します。
IsWaiting	ファイル ハンドルのウェイト状態を返します。
OpenMode	ファイル ハンドルのオープン モードを返します。
OpenModeName	OpenMode のテキスト バージョンを返します。
TransState	トランザクション状態を返します。
UserName	ファイル ハンドルに関連付けられているユーザー名を返します。

メソッド

なし

備考

「DtoFileHandles コレクション」を使って、開いているファイルのすべてのファイル ハンドルを取得します。

例

```
Dim my_session as new DtoSession
Dim my_openfiles as DtoOpenFiles
Dim first_file as DtoOpenFile
Dim my_handles as DtoFileHandles
Dim first_handle as DtoFileHandle
Dim locked_state as Boolean
Dim result as DtoResult

result = my_session.Connect("myserver", "username", "password")
Set my_monitor = my_session.Monitor
Set my_openfiles = my_monitor.OpenFiles
Set first_file = my_openfiles.Item(1)
Set my_handles = first_file.FileHandles
Set first_handle = my_handles.Item(1)
locked_state = first_handle.IsLocked
```

関連項目

「DtoFileHandles コレクション」

「DtoOpenFile オブジェクト」

「DtoMonitor オブジェクト」

DtoMkdeClients コレクション

MicroKernel エンジン クライアント オブジェクトのコレクションです。

プロパティ

Count	コレクション内のメンバー数を返します。
Item	DtoMkdeClients コレクションの特定のメンバーを返します。

メソッド

なし

備考

Count プロパティを使用してコレクション内のメンバー数を調べます。

例

```
' セッションのインスタンスを作成して接続する
Dim my_session as new DtoSession
Dim result as DtoResult

result = my_session.Connect("myserver", "username", "password")

' セッションからモニター オブジェクトを取得する
Dim my_monitor as DtoMonitor
Set my_monitor = my_session.Monitor

' モニターから MicroKernel エンジン クライアントを取得する
Dim my_mkdeclients as DtoMkdeClients
Set my_mkdeclients = my_monitor.MkdeClients
```

関連項目

「DtoMkdeClient オブジェクト」

「DtoMonitor オブジェクト」

DtoMkdeClient オブジェクト

アクティブな MicroKernel エンジン クライアントを表すオブジェクトです。

プロパティ

BtrvID	MicroKernel エンジン クライアントの Btrieve ID を返します。
CacheAccesses	MicroKernel エンジン クライアントのキャッシュ アクセス数を返します。
ClientPlatform	MicroKernel エンジン クライアントのクライアント プラットフォーム列挙を返します。可能な値のリストについては、「クライアント プラットフォーム」を参照してください。
ClientPlatformName	ClientPlatform のテキスト バージョン返します。
ClientSite	MicroKernel エンジン クライアントのクライアント サイトを返します。
ConnectionNumber	MicroKernel エンジン クライアントの接続番号を返します。
CurrentLocks	MicroKernel クライアントの現在のロック数を返します。
DiskAccesses	MicroKernel エンジン クライアントのディスク アクセス数を返します。
NetAddress	MicroKernel エンジン クライアントのアドレスを返します。
NumCursors	MicroKernel エンジン クライアントのカーソル数を返します。
RecordsDeleted	MicroKernel エンジン クライアントの削除済みレコード数を返します。
RecordsInserted	MicroKernel エンジン クライアントの挿入レコード数を返します。
RecordsRead	MicroKernel エンジン クライアントの読み込みレコード数を返します。
RecordsUpdated	MicroKernel エンジン クライアントの更新レコード数を返します。
ServiceAgentID	MicroKernel エンジン クライアントのサービス エージェント ID を返します。
TaskNumber	MicroKernel エンジン クライアントのタスク番号を返します。
TransLevel	MicroKernel エンジン クライアントのトランザクション レベルを返します。
TransState	トランザクション状態の列挙を返します。可能な値のリストについては、「トランザクション タイプ」を参照してください。
UserName	MicroKernel エンジン クライアントのユーザー名を返します。

コレクション

「DtoMkdeClientHandles コレクション」

「DtoMonitor オブジェクト」

メソッド

「Disconnect メソッド」

備考

すべての MicroKernel エンジン クライアントを取得するには、「DtoMkdeClients コレクション」を使用します。

例

```
' セッションのインスタンスを作成して接続する
Dim my_session as new DtoSession
Dim result as DtoResult
result = my_session.Connect("myserver", "username", "password")
' セッションからモニター オブジェクトを取得する
Dim my_monitor as DtoMonitor
my_monitor = my_session.Monitor

' モニターから MicroKernel エンジン クライアントを取得する
Dim my_mkdeclients as DtoMkdeClients
Set my_mkdeclients = my_monitor.MkdeClients

' 最初のクライアントを取得し、プロパティを照会する
Dim first_client as DtoMkdeClient
Dim num_locks as long
Set first_client = my_mkdeclients.Item(1)
num_locks = first_client.CurrentLocks
```

関連項目

「DtoMkdeClientHandles コレクション」

「DtoMkdeClients コレクション」

メソッドの詳細

Disconnect メソッド

特定の MicroKernel エンジン クライアントの接続を切断します。

構文

result = Object.**Disconnect**

引数

In	*Object*	DtoMkdeClient オブジェクト

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

備考

接続を切断する MicroKernel エンジン クライアントごとにこのメソッドを呼び出します。

例

```
Dim result as DtoResult
result = my_mkdeclient.Disconnect
```

DtoMkdeClientHandles コレクション

DtoMkdeClientHandle オブジェクトのコレクションです。

プロパティ

Count	コレクション内のメンバー数を返します。
Item	DtoMkdeClientHandles コレクションの特定のメンバーを返します。

メソッド

なし

備考

Count プロパティを使用してコレクション内のメンバー数を調べます。

例

```
' セッションのインスタンスを作成して接続する
Dim my_session as new DtoSession
Dim result as DtoResult
result = my_session.Connect("myserver", "username", "password")

' セッションからモニター オブジェクトを取得する
Dim my_monitor as DtoMonitor
Set my_monitor = my_session.Monitor

' モニターから MicroKernel エンジン クライアントを取得する
Dim my_mkdeclients as DtoMkdeClients
Set my_mkdeclients = my_monitor.MkdeClients

' 最初のクライアントを取得し、そのクライアント ハンドルを取得する
Dim first_client as DtoMkdeClient
Dim my_clienthandles as DtoMkdeClientHandles
Set first_client = my_mkdeclients(1)
' すべてのハンドルを取得する場合は、ClientHandles コレクションを使用する
Set my_clienthandles = first_client.ClientHandles

' コレクション内のメンバー数を調べる
Dim num_clienthandles as Long
num_clienthandles = my_clienthandles.Count
```

関連項目

「DtoMkdeClientHandle オブジェクト」

「DtoMonitor オブジェクト」

DtoMkdeClientHandle オブジェクト

MicroKernel クライアント ハンドルを表すオブジェクトです。

プロパティ

FileName	MicroKernel クライアント ハンドルに関連付けられているファイル名を返します。
LockType	MicroKernel クライアント ハンドルのロック状態の列挙を返します。可能な値のリストについては、「ロック タイプ」を参照してください。
OpenMode	MicroKernel クライアント ハンドルのオープン モード列挙を返します。可能な値のリストについては、「オープン モード」を参照してください。
OpenModeName	OpenMode のテキスト バージョンを返します。
TransState	トランザクション状態を返します。
WaitState	MicroKernel クライアント ハンドルのウェイト状態の列挙を返します。可能な値のリストについては、「ウェイト状態」を参照してください。

メソッド

なし

備考

特定のクライアントのすべての MicroKernel クライアント ハンドルを取得するには、「DtoMkdeClientHandles コレクション」を使用します。

例

```
' セッションのインスタンスを作成して接続する
Dim my_session as new DtoSession
Dim result as DtoResult
result = my_session.Connect("myserver", "username", "password")

' セッションからモニター オブジェクトを取得する
Dim my_monitor as DtoMonitor
Set my_monitor = my_session.Monitor

' モニターから MicroKernel エンジン クライアントを取得する
Dim my_mkdeclients as DtoMkdeClients
Set my_mkdeclients = my_monitor.MkdeClients

' 最初のクライアントを取得し、そのクライアント ハンドルを取得する
Dim first_client as DtoMkdeClient
Dim my_clienthandles as DtoMkdeClientHandles
Set first_client = my_mkdeclients(1)
Set my_clienthandles = first_client.MkdeClientHandles

' コレクション内のメンバー数を調べる
Dim num_clienthandles as long
num_clienthandles = my_clienthandles.Count

' 最初のクライアント ハンドルを取得し、ファイル名を照会する
Dim first_clienthandle as DtoMkdeClientHandle
```

```
Dim fileName as string
Set first_clienthandle = my_clienthandles(1)
fileName = first_clienthandle.FileName
```

関連項目

「DtoMkdeClientHandles コレクション」

「DtoMkdeClient オブジェクト」

「DtoMonitor オブジェクト」

DtoCommStat オブジェクト

サーバーの使用情報を表すオブジェクトです。

プロパティ

CurActiveThreads	セッションの現在のアクティブ スレッド数を返します。
CurQueuedRequests	セッションの待ち行列に入れられるリクエスト数を返します。
CurRemoteSessions	セッションまたはプロトコルの待ち行列に入れられるリクエスト数を返します。
MaxActiveThreads	セッションの最大アクティブ スレッド数を返します。
MaxQueuedRequests	セッションの待ち行列に入れられるリクエスト数の最大値を返します。
MaxRemoteSessions	セッションの最大リモート セッション数を返します。
PeakActiveThreads	セッションのアクティブ スレッド数のピーク値を返します。
PeakQueuedRequests	セッションの待ち行列に入れられるリクエスト数のピーク値を返します。
PeakRemoteSessions	セッションのリモート セッション数のピーク値を返します。
RequestsProcessed	セッションで処理されるリクエストの総数を返します。
TotalTimeOuts	「DTO2」のみ：通信タイム アウトの総数を返します。
TotalRecoveries	「DTO2」のみ：Zen Auto Reconnect（自動再接続の有効化）機能を使用して再接続する総数を返します。詳細については、『*Advanced Operations Guide*』を参照してください。

コレクション

「DtoProtocolStats コレクション」

オブジェクト

「DtoMonitor オブジェクト」

メソッド

なし

備考

このオブジェクトのすべてのプロパティは Long 型整数の値を返します。

例

```
' セッションのインスタンスを作成して接続する
Dim my_session as new DtoSession
Dim result as DtoResult
result = my_session.Connect("myserver", "username", "password")

' セッションからモニター オブジェクトを取得する
Dim my_monitor as DtoMonitor
Set my_monitor = my_session.Monitor
```

```
'CommStat オブジェクトを取得する
Dim my_commstat as DtoCommStat
Set my_commstat = my_monitor.MkdeCommStat

' 処理されたリクエストの総数を取得する
Dim requests as long
requests = my_commstat.RequestsProcessed
```

関連項目

「DtoMonitor オブジェクト」

「DtoProtocolStats コレクション」

「DtoProtocolStat オブジェクト」

DtoProtocolStats コレクション

DtoProtocolStat オブジェクトのコレクションです。

プロパティ

Count	コレクション内のメンバー数を返します。
Item	コレクションの特定のメンバーを返します。

メソッド

なし

備考

Count プロパティを使用してコレクション内のメンバー数を調べます。

例

```
' セッションのインスタンスを作成して接続する
Dim my_session as new DtoSession
Dim result as DtoResult
result = my_session.Connect("myserver", "username", "password")

' セッションからモニター オブジェクトを取得する
Dim my_monitor as DtoMonitor
Set my_monitor = my_session.Monitor

' モニター オブジェクトから CommStat オブジェクトを取得する
Dim my_commstat as DtoCommStat
Set my_commstat = my_monitor.MkdeCommStat

'CommStat から ProtocolStats コレクションを取得する
Dim my_protocols as DtoProtocolStats
Set my_protocols = my_commstat.ProtocolStats
```

関連項目

「DtoProtocolStat オブジェクト」

「DtoMonitor オブジェクト」

DtoProtocolStat オブジェクト

通信プロトコルに関する情報を提供します。

プロパティ

CurRemoteSessions	セッションまたはプロトコルの待ち行列に入れられるリクエスト数を返します。
PeakRemoteSessions	セッションまたはプロトコルのリモート セッション数のピーク値を返します。
ProtocolID	プロトコルの ID を返します。現在は次のリターン コードのみがサポートされています。 • 4 – WINSOCK TPC/IP
RequestsProcessed	セッションまたはプロトコルで処理されるリクエストの総数を返します。

メソッド

なし

備考

このオブジェクトを使って特定のプロトコルにアクセスするには、まず 「DtoMonitor オブジェクト」と「DtoCommStat オブジェクト」を使用して 「DtoProtocolStats コレクション」を取得します。

このオブジェクトのすべてのプロパティは Long 型整数の値を返します。

例

このプロパティを使用して処理されたリクエスト数を取得するには、次のように記述します。

```
num_requests = Object.RequestsProcessed
```

現在のリモート セッション数を取得するには、次のように記述します。

```
RemoteSess_count = Object.CurRemoteSessions
```

関連項目

「DtoProtocolStats コレクション」

「DtoCommStat オブジェクト」

「DtoMonitor オブジェクト」

DtoSqlClients コレクション

サーバーのすべての SQL クライアントを表す DtoSqlClient オブジェクトのコレクションです。

プロパティ

Count	コレクション内のメンバー数を返します。
Item	コレクションの特定のメンバーを返します。

メソッド

なし

備考

Count プロパティを使用してコレクション内のメンバー数を調べます。

例

```
' セッションのインスタンスを作成して接続する
Dim my_session as new DtoSession
Dim result as DtoResult
result = my_session.Connect("myserver", "username", "password")

' セッションからモニター オブジェクトを取得する
Dim my_monitor as DtoMonitor
Set my_monitor = my_session.Monitor

' モニター オブジェクトから SQL クライアントを取得する
Dim my_sqlclients as DtoSqlClients
Set my_sqlclients = my_monitor.SqlClients
```

関連項目

「DtoSqlClient オブジェクト」

「DtoMonitor オブジェクト」

DtoSqlClient オブジェクト

SQL クライアントに関する情報の照会および SQL クライアントの接続を切断することができます。

プロパティ

AppDesc	SQL クライアントを作成したアプリケーションの説明を返します。
ConnectTime	SQL クライアントの接続時間を返します。
CurStatusTime	最後のステータス以降の経過時間を返します。
DSN	SQL クライアントに関連付けられる DSN を返します。
HostName	SQL クライアントのホスト名を返します。
IP	SQL クライアントの IP を返します。
ProcessID	SQL クライアントのプロセス ID を返します。
Status	SQL クライアントの状態を返します。
ThreadId	SQL クライアントのスレッド ID を返します。
UserName	SQL クライアントに関連付けられているユーザー名を返します。

メソッド

「Disconnect メソッド」

備考

現在の SQL クライアントをすべて取得するには、「DtoSqlClients コレクション」を使用します。

例

```
' セッションのインスタンスを作成して接続する
Dim my_session as new DtoSession
Dim result as DtoResult
result = my_session.Connect("myserver", "username", "password")

' セッションからモニター オブジェクトを取得する
Dim my_monitor as DtoMonitor
Set my_monitor = my_session.Monitor

' モニター オブジェクトから SQL クライアントを取得する
Dim my_sqlclients as DtoSqlClients
Set my_sqlclients = my_monitor.SqlClients

' コレクションから最初のクライアントを取得し、
' それに関連付けられている DSN を見つける
Dim first_sqlclient as DtoSqlClient
Dim DSNname as string
Set first_sqlclient = my_sqlclients(1)
DSNname = first_sqlclient.DSN
```

関連項目

「DtoSqlClients コレクション」

「DtoMonitor オブジェクト」

メソッドの詳細

Disconnect メソッド

特定の SQL クライアントの接続を切断します。

構文

result = *Object*.**Disconnect**

引数

Object	DtoSqlClient オブジェクト

戻り値

result	メソッド呼び出しの結果を示す DtoResult (Long 型の値)。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

備考

接続を切断する SQL クライアントごとにこのメソッドを呼び出します。

例

```
result = my_sqlclient.Disconnect
```

DtoMkdeVersion オブジェクト

「DTO2」のみ：MicroKernel エンジンのバージョンを表すオブジェクトです。

プロパティ

MajorVersion	エンジンのメジャー バージョンを返します。
MinorVersion	エンジンのマイナー バージョンを返します。
BuildNumber	MicroKernel エンジン リリースのビルド番号
OSTarget	対象オペレーティング システムが Windows の場合は NTSV、Unix システムの場合は UXSV を返します。

メソッド

なし

備考

「DtoMonitor オブジェクト」のプロパティを使って DtoMkdeVersion オブジェクトを取得することができます。

例

```
' セッションのインスタンスを作成して接続する
Dim my_session as new DtoSession
Dim result as DtoResult
result = my_session.Connect("myserver", "username", "password")

' セッションからモニター オブジェクトを取得する
Dim my_monitor as DtoMonitor
Set my_monitor = my_session.Monitor

' モニターから MkdeVersion オブジェクトを取得する
Dim my_mkdeversion as DtoMkdeVersion
MajorVer = my_mkdeversion.MajorVersion
```

関連項目

「DtoMonitor オブジェクト」

DtoEngineInformation オブジェクト

「DTO2」のみ：データベース エンジンに関する情報を表すオブジェクトです。

プロパティ

MajorVersion	エンジンのメジャー バージョンを返します。
MinorVersion	エンジンのマイナー バージョンを返します。
dbuApiVersion	DTI/DTO インターフェイスのバージョン
IsServerEngine	ターゲットが（ワークグループ エンジンではなく）サーバー エンジンの場合は True を返します。
ServerClientType	次のいずれか 1 つが返されます。 0 = UNKNOWN_ENGINE_CLIENT 1 = NT_SERVER 3 = WIN32_CLIENT 4 = UNIX_SERVER 5 = CACHE_ENGINE_CLIENT 6 = VXWIN_SERVER 7 = VXLINUX_SERVER 9 = REPORT_ENGINE

メソッド

なし

備考

「DtoSession オブジェクト」のプロパティを使って DtoEngineInformation オブジェクトを取得することができます。

例

```
' セッションのインスタンスを作成して接続する
Dim my_session as new DtoSession
Dim result as DtoResult
result = my_session.Connect("myserver", "username", "password")

' セッションからエンジン情報を取得する
Dim my_engineInfo as DtoEngineInformation
Set my_engineInfo = my_session.EngineInformation

' エンジン情報のオブジェクトからクライアントのタイプを取得する
clientType = my_engineInfo.ServerClientType
```

関連項目

「DtoSession オブジェクト」

DTO を使用したカタログと辞書の作成および管理

5

Zen データベースおよび辞書機能を実行する COM インターフェイス

この章では、Distributed Tuning Objects のカタログ グループを構成するオブジェクトについて説明します。

DtoDatabases コレクション

プロパティ

Count	コレクション内のメンバー数を返します。
Item	DtoDatabases コレクションの特定のメンバーを返します。

メソッド

「Add メソッド」

「Remove メソッド」

備考

Count プロパティを使用してコレクション内のメンバー数を調べます。

例

```
' セッション オブジェクトのインスタンスを作成し、サーバーに接続する
Dim my_session as New DtoSession
Dim result as DtoResult
result = my_session.Connect("myserver", "username", "password")

' セッション オブジェクトを使用して Databases コレクションを取得する
Dim my_databases as DtoDatabases
Set my_databases = my_session.Databases
```

関連項目

「DtoDatabase オブジェクト」

「DtoSession オブジェクト」

メソッドの詳細

Add メソッド

DtoDatabases コレクションに項目を追加します。

構文

result = *Collection*.**Add**(*Object*[, *dsnName*])

引数

Collection	オブジェクトを追加する DtoDatabases コレクション。
Object	新しい DtoDatabase オブジェクト。
dsnName	省略可能。新しいデータベースの標準サーバー DSN を作成します。

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

備考

このメソッドはオブジェクト タイプのパラメーターを使用します。このため、コレクションにオブジェクトを追加する前に、まずオブジェクトのインスタンスを作成してそのプロパティを設定する必要があります。

このメソッドは、指定したデータベースをコレクションとサーバー上の基となる DBNAMES.CFG ファイルに追加します。

例

```
Dim result As dtoResult
Dim database As DtoDatabase

Set database = New DtoDatabase
' 新しいデータベースにプロパティを設定する
database.Name = "MyDemodata"
database.DdfPath = "C:¥test"
database.DataPath = "C:¥test"
database.Flags = dtoDbFlagCreateDDF + dtoDbFlagRI

result = my_session.Databases.Add(database)
If NOT result = Dto_Success Then
   MsgBox "Error"+ Session.Error(result)
End If
```

Remove メソッド

DtoDatabases コレクションから項目を削除します。

構文

result = *Collection*.**Remove**(*database*[, *deleteDDF*])

引数

Collection	オブジェクトを削除するコレクション。
database	コレクションから削除する項目の（1 から始まる）インデックスまたは項目のデータベース名を含むバリアントを指定できます。
deleteDDF	辞書ファイルを削除するには True を設定します。 辞書ファイルを完全に残す場合は False を設定します。

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

備考

このメソッドは、データベース コレクションおよび基となる DBNAMES.CFG ファイルから項目を削除します。

例

```
Dim result As dtoResult
result = my_session.Databases.Remove("MyDemodata",1)
If NOT result = Dto_Success Then
   MsgBox "Error"+ my_session.Error(result)
End If
```

DtoDatabase オブジェクト

プロパティ

DataPath	データベースのデータの場所を取得または設定します。
DbCodePage	データベースのコード ページを取得または設定します。このプロパティは列挙型です。値の一覧については、「データベース コード ページ」を参照してください。ゼロの値はサーバーのエンコード（データベース エンジンを起動しているサーバーのコード ページ）を指定します。
DbFlags	データベースのデータベース フラグを取得または設定します。このプロパティは列挙型です。値の一覧については、「データベース フラグ」を参照してください。
DdfPath	データベースの辞書のパスを取得または設定します。
Name	データベースの名前を設定または取得します。
Secured	データベースにセキュリティが設定されているかどうかを返します（0 = セキュリティ未設定、1 = セキュリティ設定済み）
Session	この DtoDatabase オブジェクトに関連付けられている Session オブジェクトを取得または設定します。

コレクション

「DtoTables コレクション」

メソッド

「AddUserToGroup メソッド」

「AlterUserName メソッド」

「AlterUserPassword メソッド」

「Close メソッド」

「Copy メソッド」

「CreateGroup メソッド」

「CreateUser メソッド」

「DropGroup メソッド」

「DropUser メソッド」

「Open メソッド」

「RemoveUserFromGroup メソッド」

「Secure メソッド」

「UnSecure メソッド」

備考

Secure メソッドおよび UnSecure メソッドは、データベースが閉じている場合にのみ使用可能です。

例

以下の例では、セッション オブジェクトのインスタンスを作成し、サーバーに接続する方法を示します。

```
' セッション  オブジェクトのインスタンスを作成し、サーバーに接続する
Dim my_session as New DtoSession
```

```
Dim result as DtoResult
result = my_session.Connect("myserver", "username", "password")

' セッション オブジェクトを使用して Databases コレクションを取得する
Dim my_databases as DtoDatabases
Set my_databases = my_session.Databases

' 最初のデータベースを取得し、その辞書のパスを照会する
Dim first_database as DtoDatabase
Dim dictionarypath as string
Set first_database = my_databases(1)
dictionarypath = first_database.DdfPath
```

以下の例では、"Demodata" サンプル データベースで DBCodePage プロパティを使用したコード ページの取得および設定方法を示します。

```
Dim m_dtoSession1 As New DtoSession
Dim result As dtoResult
result = m_dtoSession1.Connect("localhost", "", "")
Dim sCodePage As String
sCodePage = m_dtoSession1.Databases("DEMODATA").DBCodePage
MsgBox "Code Page for database (before change): " & CStr(sCodePage)
If result = Dto_Success Then
Rem Set the code page for the database by passing in
Rem the code page number (for example, 0, 932, 1252,
Rem and so forth).
m_dtoSession1.Databases("DEMODATA").DBCodePage = 0
End If
MsgBox "Code Page for database: " &
    CStr(m_dtoSession1.Databases("DEMODATA").DBCodePage)
m_dtoSession1.Disconnect
```

関連項目

「DtoDatabases コレクション」

メソッドの詳細

AddUserToGroup メソッド

既存ユーザーをデータベースの既存グループに追加します。

構文

result = *Object*.**AddUserToGroup**(*username*, *groupname*)

引数

Object	Dtodatabase オブジェクト。
username	グループに追加するユーザー名。
groupname	ユーザーを追加するグループ名。

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。

備考

この関数は、指定したグループまたはユーザーがデータベースにあらかじめ存在していない場合や、ユーザーが別のグループのメンバーである場合は失敗します。

以下の前提条件を満たす必要があります。

- まずセッションを作成し、次に「Open メソッド」を使って、"Master" ユーザーとしてデータベースを正常に開いておく。
- 関連するデータベースはデータベース レベルのセキュリティが有効である。
- ユーザーおよびグループは指定したデータベースに既に存在している。
- ユーザーは別のグループのメンバーではない。

次の事後条件を満たす必要があります。

- 「Close メソッド」を使ってデータベースを閉じ、リソースを解放する。

例

```
Function AddUserToGroup(sUserName As String, sGroupName As String) As Boolean
Dim res As dtoResult
Dim m_dbn As New DtoDatabase
Dim m_dbn.Session = m_dto
Dim m_dbn.Name = "demodata"
res = m_dbn.Open("Master", "1234")
If res = Dto_Success Then
    ' 正常に開いたら、グループにユーザーを追加しましょう
    res = m_dbn.AddUserToGroup(sUserName, sGroupName)
    If res <> Dto_Success Then
        LogResult(" グループへのユーザーの追加でエラーが発生しました：" & CStr(res))
    Else
        LogResult(" ユーザー " & sUserName & " がグループ " & sGroupName & " に追加されました。")
    End If
End If
m_dbn.Close
End Function
```

AlterUserName メソッド

指定されたデータベースの既存のユーザーの名前を変更します。

構文

result = *Object*.**AlterUserName**(*username*, *new_username*)

引数

Object	Dtodatabase オブジェクト。

username	既存のデータベース ユーザーの名前。
new_username	データベース ユーザーの新しい名前。ヌルを設定すると関数は失敗します。

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。

備考

以下の前提条件を満たす必要があります。

- まずセッションを作成し、次に「Open メソッド」を使って、"Master" ユーザーとしてデータベースを正常に開いておく。
- 関連するデータベースはデータベース レベルのセキュリティが有効である。
- ユーザー名は指定したデータベースに既に存在している。
- 新しいユーザー名が指定したデータベースに存在していない。

次の事後条件を満たす必要があります。

- 「Close メソッド」を使ってデータベースを閉じ、リソースを解放する。

例

```
Function AlterUserName(sUserName As String, sNewUserName As String) As Boolean
Dim res As dtoResult
Dim m_dbn As New DtoDatabase
Dim m_dbn.Session = m_dto
Dim m_dbn.Name = "demodata"
res = m_dbn.Open("Master", "1234")
If res = Dto_Success Then
    ' 正常に開いたら、ユーザー名を変更しましょう
    res = m_dbn.AlterUserName(sUserName, sNewUserName)
    If res <> Dto_Success Then
        LogResult(" ユーザー名の変更でエラーが発生しました：" & CStr(res))
    Else
        LogResult(" ユーザー名は正常に変更されました。新しいユーザー名：" & sNewUserName)
    End If
End If
m_dbn.Close
End Function
```

AlterUserPassword メソッド

既存のユーザーのパスワードを変更します。

構文

result = *Object*.**AlterUserPassword**(*username*, *new_password*)

引数

Object	Dtodatabase オブジェクト。
username	パスワードを変更するデータベース ユーザーの名前。
new_password	ユーザーの新しいパスワード。ヌルを設定するとパスワードがクリアされます。

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。

備考

以下の前提条件を満たす必要があります。

- まずセッションを作成し、次に「Open メソッド」を使って、"Master" ユーザーとしてデータベースを正常に開いておく。
- 関連するデータベースはデータベース レベルのセキュリティが有効である。
- ユーザー名は指定したデータベースに既に存在している。

次の事後条件を満たす必要があります。

- 「Close メソッド」を使ってデータベースを閉じ、リソースを解放する。

例

```
Function AlterUserPassword(sUser As String, sNewPassword As String) As Boolean
Dim res As dtoResult
Dim m_dbn As New DtoDatabase
Dim m_dbn.Session = m_dto
Dim m_dbn.Name = "demodata"
res = m_dbn.Open("Master", "1234")
If res = Dto_Success Then
    ' 正常に開いたら、ユーザーのパスワードを変更しましょう
    res = m_dbn.AlterUserPassword(sUser, sNewPassword)
    If res <> Dto_Success Then
        LogResult(" ユーザーのパスワードの変更でエラーが発生しました：" & CStr(res))
    Else
        LogResult(" ユーザーのパスワードは正常に変更されました。")
    End If
End If
m_dbn.Close
End Function
```

Close メソッド

Open メソッドを使用して開いたデータ辞書ファイルのセットを閉じます。

構文

result = *Object*.**Close**

引数

Object	DtoDatabase オブジェクト

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

備考

Open メソッドを使ってデータベースを開い後にこのメソッドを呼び出します。エラー情報は *Error* プロパティを使って取得することができます。

例

```
Dim m_database as new DtoDatabase
Dim result as DtoResult

result = m_database.Open("dbuser","pwd")

' ここで操作を実行
'
result = m_database.Close
```

Copy メソッド

現在のデータベースを基にして新しいデータベースを作成します。

構文

result = *Object*.**Copy**(*username*, *password*, *newDBname*, *newDictionaryPath*, *newDataPath*)

引数

Object	DtoDatabase オブジェクト
username	データベース用のデータベース ユーザー名です。データベースにセキュリティが設定されていない場合、空文字列を設定します。
password	データベース ユーザーのパスワードです。データベースにセキュリティが設定されていない場合、空文字列を設定します。
newDBname	コピーしたデータベース用のデータベース名です。
newDictionaryPath	辞書ファイルを作成するディレクトリへの絶対パス。これは既存のディレクトリでなければなりません。
newDataPath	データベースのデータ パスです。デフォルトのデータ パス（つまり、辞書パスと同じパス）を使用するには、空文字列を渡します。

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

備考

コピーされたデータベースにおいて参照整合性は保持されます。

このメソッドで返されるエラーの詳細については、*Error* プロパティを使って取得することができます。

例

```
Dim Database As New DtoDatabase
Dim result as DtoResult
Database.Session = my_session ' セッションが存在すると仮定
Database.Name = "DEMODATA"
' セキュリティが設定されていないデータベースではユーザー名とパスワードは不要
result = Database.Copy("", "", "DEMODATA2", "D:¥DEMODATA2", "D:¥DEMODATA2")

If NOT result = Dto_Success Then
  MsgBox "Error"+ Session.Error(result)
End If
```

CreateGroup メソッド

既存のデータベースに新しいユーザー グループを作成します。

構文

result = *Object*.**CreateGroup**(*groupname*)

引数

Object	Dtodatabase オブジェクト。
groupname	データベースに追加するグループの名前。

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。

備考

以下の前提条件を満たす必要があります。

- まずセッションを作成し、次に「Open メソッド」を使って、"Master" ユーザーとしてデータベースを正常に開いておく。
- 関連するデータベースはデータベース レベルのセキュリティが有効である。
- 同じ名前のグループが指定したデータベースに存在していない。

次の事後条件を満たす必要があります。

- 「Close メソッド」を使ってデータベースを閉じ、リソースを解放する。

例

```
Function CreateGroup(sGroupName As String) As Boolean
Dim res As dtoResult
Dim m_dbn As New DtoDatabase
Dim m_dbn.Session = m_dto
Dim m_dbn.Name = "demodata"
res = m_dbn.Open("Master", "1234")
If res = Dto_Success Then
    ' 正常に開いたら、グループを作成しましょう
    res = m_dbn.CreateGroup(sGroupName)
    If res <> Dto_Success Then
        LogResult(" グループの作成でエラーが発生しました：" & CStr(res))
    Else
        LogResult(" グループ " & sGroupName & " が作成されました。")
    End If
End If
m_dbn.Close
End Function
```

CreateUser メソッド

既存のデータベースに新しいユーザーを作成します。任意で、パスワードを設定することと、新しいユーザーを既存のグループに割り当てることができます。

構文

result = *Object*.**CreateUser**(*username*[, *password*][, *groupname*])

引数

Object	Dtodatabase オブジェクト。
username	データベースに追加するユーザーの名前。
password	ユーザー パスワード。ヌルを設定するとパスワードは設定されません。
groupname	ユーザーを割り当てるデータベース グループの名前。ヌルを設定するとユーザーはグループに割り当てられません。

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。

備考

以下の前提条件を満たす必要があります。

- まずセッションを作成し、次に「Open メソッド」を使って、"Master" ユーザーとしてデータベースを正常に開いておく。
- 関連するデータベースはデータベース レベルのセキュリティが有効である。

- 同じ名前のユーザーが指定したデータベースに存在していない。

次の事後条件を満たす必要があります。

- 「Close メソッド」を使ってデータベースを閉じ、リソースを解放する。

例

```
Function CreateUser(sUserName As String, sPassword As String, sGroupName As String)
    As Boolean
Dim res As dtoResult
Dim m_dbn As New DtoDatabase
Dim m_dbn.Session = m_dto
Dim m_dbn.Name = "demodata"
res = m_dbn.Open("Master", "1234")
If res = Dto_Success Then
    ' 正常に開いたら、グループを作成しましょう
    res = m_dbn.CreateUser(sUserName, sPassword, sGroupName)
    If res <> Dto_Success Then
        LogResult(" ユーザーの作成でエラーが発生しました：" & CStr(res))
    Else
        LogResult(" ユーザー " & sUserName & " がグループ " & sGroupName & " に作成されま
    した。")
    End If
End If
m_dbn.Close
End Function
```

DropGroup メソッド

データベースから既存のグループを削除します。

構文

result = *Object*.**DropGroup**(*groupname*)

引数

Object	Dtodatabase オブジェクト。
groupname	データベースから削除するグループの名前。

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。

備考

以下の前提条件を満たす必要があります。

- まずセッションを作成し、次に「Open メソッド」を使って、"Master" ユーザーとしてデータベースを正常に開いておく。
- 関連するデータベースはデータベース レベルのセキュリティが有効である。
- 同じ名前のグループが指定したデータベースに存在していない。

- グループにはメンバーが含まれていない。

次の事後条件を満たす必要があります。

- 「Close メソッド」を使ってデータベースを閉じ、リソースを解放する。

例

```
Function DropGroup(sGroupName As String) As Boolean
Dim res As dtoResult
Dim m_dbn As New DtoDatabase
Dim m_dbn.Session = m_dto
Dim m_dbn.Name = "demodata"
res = m_dbn.Open("Master", "1234")
If res = Dto_Success Then
    ' 正常に開いたら、グループを削除しましょう
    res = m_dbn.DropGroup(sGroupName)
    If res <> Dto_Success Then
        LogResult(" グループの削除でエラーが発生しました：" & CStr(res))
    Else
        LogResult(" グループ " & sGroupName & " が削除されました。")
    End If
End If
m_dbn.Close
End Function
```

DropUser メソッド

データベースから既存のユーザーを削除します。

構文

result = *Object*.**DropUser**(*username*)

引数

Object	Dtodatabase オブジェクト。
username	データベースから削除するユーザーの名前。

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。

備考

以下の前提条件を満たす必要があります。

- まずセッションを作成し、次に「Open メソッド」を使って、"Master" ユーザーとしてデータベースを正常に開いておく。
- 関連するデータベースはデータベース レベルのセキュリティが有効である。
- 同じ名前のユーザーが指定したデータベースに存在している。

次の事後条件を満たす必要があります。

- 「Close メソッド」を使ってデータベースを閉じ、リソースを解放する。

例

```
Function DropUser(sUserName As String) As Boolean
Dim res As dtoResult
Dim m_dbn As New DtoDatabase
Dim m_dbn.Session = m_dto
Dim m_dbn.Name = "demodata"
res = m_dbn.Open("Master", "1234")
If res = Dto_Success Then
    ' 正常に開いたら、ユーザーを削除しましょう
    res = m_dbn.DropUser(sUserName)
    If res <> Dto_Success Then
        LogResult(" ユーザーの削除でエラーが発生しました：" & CStr(res))
     Else
        LogResult(" ユーザー " & sUserName & " の削除は完了しました。")
End If
End If
m_dbn.Close
End Function
```

Open メソッド

指定したユーザー名とパスワードでデータベースへの接続を開きます。

構文

result = *Object*.**Open**(*username*, *password*)

引数

Object	DtoDatabase オブジェクト
username	データベース用のユーザー名です。データベースにセキュリティが設定されていない場合、空の文字列を設定します。
password	データベース用のパスワードです。データベースにセキュリティが設定されていない場合、空の文字列を設定します。

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

備考

この操作は、辞書ファイルのセットを開く手段として使用されます。このセットには、FILE.DDF、INDEX.DDF および FIELD.DDF が含まれます。また、多くのオプション DDF ファイルも含まれています。メモリを解放するために *Close* メソッドを呼び出すことを忘れないでください。データベースを一度開くと、Close メソッドが呼び出されるまでほかの誰もその辞書セットに変更を行うことができません。

データベースが開いている間は、Secure または UnSecure メソッドを実行できません。

このメソッドで返されるエラーの詳細については、「DtoSession オブジェクト」の *Error* プロパティを使って取得することができます。

例

```
Dim m_session as new DtoSession
Dim m_database as new DtoDatabase
Dim result as DtoResult
result = m_session.Connect("myserver","user","pwd")
m_database.Session = m_session
m_database.Name = "DEMODATA"
result = m_database.Open("dbuser","pwd")
```

RemoveUserFromGroup メソッド

既存のグループから既存のユーザーを削除します。

構文

result = *Object*.**RemoveUserFromGroup**(*groupnamem*, *username*)

引数

Object	Dtodatabase オブジェクト。
groupname	データベース グループ名
username	データベース ユーザー名

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。

備考

以下の前提条件を満たす必要があります。

- まずセッションを作成し、次に「Open メソッド」を使って、"Master" ユーザーとしてデータベースを正常に開いておく。
- 関連するデータベースはデータベース レベルのセキュリティが有効である。
- ユーザーおよびグループは指定したデータベースに既に存在している。
- ユーザーは別のグループのメンバーではない。

次の事後条件を満たす必要があります。

- 「Close メソッド」を使ってデータベースを閉じ、リソースを解放する。

例

```
Function RemoveUserFromGroup(sUserName As String, sGroupName As String) As Boolean
Dim res As dtoResult
Dim m_dbn As New DtoDatabase
Dim m_dbn.Session = m_dto
Dim m_dbn.Name = "demodata"
res = m_dbn.Open("Master", "1234")
If res = Dto_Success Then
    ' 正常に開いたら、グループからユーザーを削除しましょう
    res = m_dbn.RemoveUserFromGroup(sGroupName, sUserName)
```

```
    If res <> Dto_Success Then
        LogResult("グループからのユーザーの削除でエラーが発生しました：" & CStr(res))
     Else
        LogResult("ユーザー " & sUserName & " をグループ " & sGroupName & " から削除しました。")
    End If
End If
m_dbn.Close
End Function
```

Secure メソッド

データベースのセキュリティを有効にします。

構文

result = *Object*.**Secure**(*user*, *password*)

引数

Object	DtoDatabase オブジェクト
user	データベースのセキュリティを有効にできる「Master」として設定するユーザー。
password	Master ユーザーのパスワード。

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

備考

データベースのセキュリティを有効にする際、データベース ユーザー名として Master を指定し、パスワードを選択する。データベースのセキュリティは、そのデータベースに定義されているアクセス権に基づいて施行されます。このセキュリティは SQL または ODBC アクセス方法で見られる動作と一致します。

セキュリティを設定する場合は、データベースが閉じていることを確認してください。

このメソッドで返されるエラーの詳細については、「DtoSession オブジェクト」の *Error* プロパティを使って取得することができます。

例

```
Dim m_database as new DtoDatabase
Dim result as DtoResult
m_database.Name = "DEMODATA"
m_database.Session = my_session ' セッションが存在すると仮定
result = m_database.Secure("Master", "password")
```

UnSecure メソッド

データベースのセキュリティを無効にします。

構文

result = *Object*.**UnSecure**(*user*, *password*)

引数

Object	DtoDatabase オブジェクト
user	データベースのセキュリティを無効にできる「Master」として設定するユーザー。
password	Master ユーザーのパスワード。

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

備考

データベースのセキュリティを無効にする際、データベース ユーザーとして Master を指定し、Master ユーザー パスワードを提供する必要があります。

セキュリティを無効にする場合は、データベースが閉じていることを確認してください。

このメソッドで返されるエラーの詳細については、「DtoSession オブジェクト」の *Error* プロパティを使って取得することができます。

例

```
Dim m_database as new DtoDatabase
Dim result as DtoResult
m_database.Name = "DEMODATA"
m_database.Session = my_session ' セッションが存在すると仮定
result = m_database.UnSecure("Master", "password")
```

DtoDSNs コレクション

DtoDSN オブジェクトのコレクションです。

プロパティ

Count	コレクション内のメンバー数を返します。
Item	DtoDSNs コレクションの特定のメンバーを返します。

メソッド

「Add メソッド」

「Remove メソッド」

備考

Count プロパティを使用してコレクション内のメンバー数を調べます。

例

```
' セッション オブジェクトのインスタンスを作成する
Dim my_session as New DtoSession
Dim result as DtoResult

' サーバーに接続する
result = my_session.Connect("myserver", "username", "password")

'DSN コレクションを取得する
Dim my_dsns as DtoDSNs
Set my_dsns = my_session.DSNs
```

関連項目

「DtoDSN オブジェクト」

「DtoSession オブジェクト」

メソッドの詳細

Add メソッド

DtoDSNs コレクションに項目を追加し、サーバーに DSN を作成します。

構文

result = *Collection*.**Add**(*Object*)

引数

Collection	オブジェクトを追加する DtoDSNs コレクション。
Object	新しい DtoDSN オブジェクト。

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

備考

このメソッドはオブジェクト タイプのパラメーターを使用します。このため、コレクションにオブジェクトを追加する前に、まずオブジェクトのインスタンスを作成してそのプロパティを設定する必要があります。

例

```
Dim result As dtoResult
Dim DSNs As DtoDSNs
Dim dsn As DtoDSN

Set dsn = New DtoDSN

' 新しい DSN にプロパティを設定する
dsn.Name = "MyDemodata_DSN"
dsn.Description = "a sample DSN"
dsn.Dbname = "MyDemodata"
dsn.Openmode = dtoNormalDSNOpenMode

result = my_session.DSNs.Add(dsn)
If NOT result = Dto_Success Then
  MsgBox "Error"+ my_session.Error(result)
End If
```

Remove メソッド

DtoDSNs コレクションから DSN 項目を削除し、サーバーからも削除します。

構文

result = *Collection*.**Remove**(*dsn*)

引数

Collection	オブジェクトを削除するコレクション。
dsn	コレクションから削除する項目の（1 から始まる）インデックスまたは項目の名前を含むバリアントを指定できます。

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

備考

このメソッドでは、関連するデータベースまたはデータベース名を削除しません。

例

```
Dim result As dtoResult
Dim DSNs As DtoDSNs
result = my_session.DSNs.Remove("MYDSN")
If NOT result = Dto_Success Then
  MsgBox "Error"+ my_session.Error(result)
End If
```

DtoDSN オブジェクト

Zen DSN を表すオブジェクトです。

プロパティ

DbName	DSN に関連付けられているデータベース名を取得または設定します。
Description	DSN の説明を設定または取得します。
Name	DSN の名前を設定または取得します。
OpenMode	DSN のオープン モード（列挙型）を設定または取得します。 可能な値のリストについては、「DSN オープン モード」を参照してください。
Session	この DtoDSN オブジェクトに関連付けられている DtoSession オブジェクトを取得または設定します。
Translate	エンコード変換を取得または設定します。これはデータベース エンジンとクライアント アプリケーション間で文字データをどのように変換するかを指定します。このプロパティは列挙型です。値の一覧については、「DSN 変換オプション」を参照してください。

メソッド

なし

備考

特定のデータベースに関する情報を取得するには、「DtoDatabase オブジェクト」を使用します。

例

DSN の関連するデータベース名を照会するには

```
' セッション オブジェクトのインスタンスを作成する
Dim my_session as New DtoSession
' サーバーに接続する
my_session.Connect("myserver", "username", "password")
' セッション オブジェクトを使用して Databases コレクションを取得する
my_dsns = my_session.DSNs
first_dsn = my_dsns.Item(1)
dsn_dbname = first_dsn.DbName
```

新しい DSN を追加するには

```
' セッション オブジェクトのインスタンスを作成する
Dim my_session as New DtoSession
Dim result as dtoResult

' サーバーに接続する
result = my_session.Connect("myserver", "username", "password")
' セッション オブジェクトを使用して DSNs コレクションを取得する
Dim my_dsns as DtoDSNs
Set my_dsns = my_session.DSNs

' 新しい DtoDSN オブジェクトを作成する
```

```
Dim NewDSN as New DtoDSN
NewDSN.DbName = "DEMODATA"
NewDSN.Description = "A DSN for the DEMODATA db"
NewDSN.Name = "Demodata_DSN"

' 新しい DSN をコレクションに追加する
result = my_dsns.Add(NewDSN)
```

エンコード変換を取得または設定するには

```
Dim m_dtoSession1 As New DtoSession
Dim result As dtoResult
result = m_dtoSession1.Connect("localhost", "", "")
Dim sTranslate As String
Dim iTranslate As Integer
iTranslate = m_dtoSession1.DSNs("DEMODATA").Translate
If iTranslate = 0 Then sTranslate = "None"
If iTranslate = 1 Then sTranslate = "OEM/ANSI Conversion"
If iTranslate = 2 Then sTranslate = "Automatic"

MsgBox "DSN Translate Setting (before change): " & sTranslate
If result = Dto_Success Then
Rem set the encoding translation.
m_dtoSession1.DSNs("DEMODATA").Translate = 1
End If
iTranslate = m_dtoSession1.DSNs("DEMODATA").Translate
If iTranslate = 0 Then sTranslate = "None"
If iTranslate = 1 Then sTranslate = "OEM/ANSI Conversion"
If iTranslate = 2 Then sTranslate = "Automatic"

MsgBox "DSN Translate Setting (after change): " & sTranslate
m_dtoSession1.Disconnect
```

関連項目

「DtoDSNs コレクション」

「DtoSession オブジェクト」

DtoDictionary オブジェクト

Zen 辞書を表すオブジェクトです。このオブジェクトは「DtoDatabase オブジェクト」に置き換わるため、使用することは推奨されません。DtoDictionary は、Open メソッドで辞書へのパスを指定する場合にのみ今後も使用できます。

プロパティ

Path	辞書オブジェクトのパスを返します。

コレクション

「DtoTables コレクション」

メソッド

「Open メソッド」

「Create メソッド」

「Close メソッド」

「AddTable メソッド」

「DropTable メソッド」

「Reload メソッド」

「Delete メソッド」

備考

辞書ファイルに影響するすべての操作は、このオブジェクトから行う必要があります。このオブジェクトを使用してユーザーは、辞書のオープン、辞書の作成、テーブル情報の取得、テーブルの追加または削除を行うことができます。

メモ ASP を使って、または Visual Basic の CreateObject メソッドを使ってこのオブジェクトのインスタンスを作成する場合、DtoDictionary のプログラム ID は、DTO2 の場合は "DTO.DtoDictionary.2"、DTO バージョン 1 の場合は "DTO.DtoDictionary.1" になります。これら 2 つのバージョンの違いについては、「DTO2」を参照してください。

例

```
Dim result as DtoResult
Dim dictionary as New DtoDictionary
result = dictionary.Open("d:¥MyDemodata")
```

関連項目

「DtoTables コレクション」

「DtoTable オブジェクト」

メソッドの詳細

Open メソッド

データベース名または辞書のパスを使ってデータ辞書ファイル セットを開きます。

構文

result = *Object*.**Open**(*path*[, *user*][, *password*])

引数

Object	DtoDictionary オブジェクト。
path	ローカルの場合、辞書ファイルまたは名前付きデータベースの名前があるディレクトリへの絶対パスです。 リモート サーバーに接続している場合、この引数に対し名前付きデータベースを使用することはできません。
user	DDF セットの任意のユーザー名。
password	DDF セットの任意のパスワード。

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

備考

メモ 引数 *path* には、DDF ファイルがあるディレクトリへのパス、あるいは ローカルの DBNAMES.CFG に含まれるデータベース名を使用することができます。 データベース名の作成および管理については、「DtoDatabases コレクション」を参照してください。

この操作は、辞書ファイルのセットを開く手段として使用されます。 このセットには、FILE.DDF、INDEX.DDF および FIELD.DDF が含まれます。また、多くのオプション DDF ファイルも含まれています。 メモリを解放するために *Close* メソッドを呼び出すことを忘れないでください。辞書セットを一度開くと、Close メソッドが呼び出されるまでほかの誰もその辞書セットに変更を行うことができません。

このメソッドで返されるエラーの詳細については、「DtoSession オブジェクト」の *Error* プロパティを使って取得することができます。

例

```
Dim dictionary as new DtoDictionary
Dim result as DtoResult

result = dictionary.Open("d:¥MyDemodata")
```

Create メソッド

空のデータ辞書ファイルのセットを作成します。

構文

result = *Object*.**Create**(*path*[, *username*][, *password*])

引数

Object	DtoDictionary オブジェクト。
path	辞書ファイルを作成するディレクトリへの絶対パス。
username	DDF セットの任意のユーザー名。
password	DDF セットの任意のパスワード。

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

備考

path 引数に指定したディレクトリが存在しなかった場合、そのディレクトリの作成を試行します。操作が成功した場合、file.ddf、field.ddf、index.ddf のセットが作成されます。

メモリを解放するために *Close* メソッドを呼び出すことを忘れないでください。辞書セットが一度作成されると、その他のクライアントは Close メソッドが呼び出されるまでその辞書セットを開くことも変更を行うこともできません。

このメソッドで返されるエラーの詳細については、*Error* プロパティを使って取得することができます。Open メソッドとは異なり、path パラメーターに指定できるのは絶対パスのみです。

例

```
Dim Dictionary As New DtoDictionary
Dim result as DtoResult

result = Dictionary.Create("C:¥TEST", "login", "password")
If NOT result = Dto_Success Then
  MsgBox "Error"+ Session.Error(result)
End If
```

Close メソッド

データ辞書ファイルのセットを閉じます。*Open* メソッドを使って開いている、または *Create* メソッドを使って作成されていることが前提です。

構文

result = *Object*.**Close**

引数

Object	DtoDictionary オブジェクト。

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

備考

辞書ファイルのセットを *Open* メソッドを使って開いている、または *Create* メソッドを使って作成した後にこのメソッドを呼び出します。エラー情報は *Error* プロパティを使って取得することができます。

例

```
Dim dictionary as new DtoDictionary
Dim result as DtoResult

result = dictionary.Open("d:¥MyDemodata")
'
' ここで操作を実行
'
result = dictionary.Close
```

AddTable メソッド

データ辞書ファイルにテーブル情報を追加し、定義を一致させるためにデータ ファイルを作成します。

メモ 同じディレクトリに、ファイル名が同一で拡張子のみが異なるようなファイルを置かないでください。たとえば、同じディレクトリ内に Invoice.btr と Invoice.mkd という名前のデータ ファイルを作成しないでください。このような制限が設けられているのは、データベース エンジンがさまざまな機能でファイル名のみを使用し、ファイルの拡張子を無視するためです。ファイルの識別にはファイル名のみが使用されるため、ファイルの拡張子だけが異なるファイルは、データベース エンジンでは同一のものであると認識されます。

構文

result = *Object*.**AddTable**(*table*)

引数

Object	DtoDictionary オブジェクト。
table	DtoTable オブジェクト。

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

備考

このメソッドは、DDF ファイルにテーブル定義を追加し、テーブルの Location プロパティで指定したデータ ファイルの作成を試みます。Location プロパティが空のままの場合、このメソッドでは *tableName.mkd* という名前のデータ ファイルの作成を試みます。この名前のテーブルが既に存在していた場合は、その名前に 1 つの番号を追加して再度作成を試みます。

この操作が正常終了するためには、少なくとも 1 列が定義されていなければなりません。

例

以下の例では、辞書オブジェクトを作成し、次にその辞書ファイルにテーブルを追加する方法を示します。

```
Dim Dictionary As New DtoDictionary
Dim Table As DtoTable
Dim Tables As DtoTables
Dim result As dtoResult
Dim Columns As DtoColumns
Dim Indexes As DtoIndexes
Dim Column As DtoColumn
Dim Index As DtoIndex
Dim Segments As DtoSegments
Dim Segment As DtoSegment

    result = Dictionary.Create("C:¥TEST", "login", "password")
    If NOT result = Dto_Success Then
        MsgBox "Error"+ Session.Error(result)
    End If

' ******* テーブルの追加の開始 ******************

    Set Table = New DtoTable

    Set Column = New DtoColumn
    With Column
    .Decimal = 0
    .Flags = dtoColumnNullable
    .ISR = ""
    .Name = "F_Int"
    .Number = 0
    .Size = 4
    .Type = dtoTypeInteger
    End With
    Table.Columns.Add Column

    Set Column = New DtoColumn
    With Column
    .Decimal = 4
    .Flags = dtoColumnNullable + dtoColumnCaseInsensitive
    .ISR = ""
    .Name = "F_Str"
    .Number = 1
```

```
    .Size = 55
    .Type = dtoTypeLString
    End With
    Table.Columns.Add Column

Set Column = New DtoColumn
    With Column
    .Decimal = 4
    .Flags = dtoColumnCaseInsensitive
    .ISR = ""
    .Name = "F_Str_Second"
    .Number = 2
    .Size = 100
    .Type = dtoTypeLString
    End With
    Table.Columns.Add Column

    Set Column = New DtoColumn
    With Column
    .Decimal = 10
    .Flags = dtoColumnDefault
    .ISR = ""
    .Name = "F_Float"
    .Number = 3
    .Size = 8
    .Type = dtoTypeBFloat
    End With
    Table.Columns.Add Column

' インデックスを追加
    Set Index = New DtoIndex
    result = Index.AddSegment("F_Int", 0)
    Set Segment = New DtoSegment
    Segment.Number = 0
    Segment.ColumnName = "F_Int"
    Segment.Flags = dtoSegmentAscending
    Index.Segments.Add Segment

    Index.Name = "FintInd"
    Index.Number = 0
    Index.Flags = dtoIndexModifiable
    Table.Indexes.Add Index
'2 番目のインデックスを追加
    Set Index = New DtoIndex
    Set Segment = New DtoSegment
    Segment.Number = 0
    Segment.ColumnName = "F_Str"
    Segment.Flags = dtoSegmentAscending
    Index.Segments.Add Segment

    Set Segment = New DtoSegment
    Segment.Number = 1
    Segment.ColumnName = "F_Str_Second"
    Segment.Flags = dtoSegmentAscending
    Index.Segments.Add Segment

    Index.Name = "FStrTagInd"
    Index.Number = 1
```

```
Index.Flags = dtoIndexModifiable
Table.Indexes.Add Index

Table.Overwrite = true
Table.Flags = dtoTableTrueNullable
Table.Name = "Table3"

result = Dictionary.AddTable(Table)

If NOT result = Dto_Success Then
    MsgBox "Error"+ Session.Error(result)
End If
```

DropTable メソッド

現在の辞書からテーブルを削除します。

構文

result = *Object*.**DropTable**(*tableName*[, *deleteFile*])

引数

Object	DtoDictionary オブジェクト。
tableName	削除するテーブルの名前。
deleteFile	基となるデータ ファイルを削除するかどうかを示すブール値。

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

備考

この操作を成功させるには、辞書が正常に開かれている必要があります。

例

```
result = Dictionary.DropTable("Table3", true)
If NOT result = Dto_Success Then
    MsgBox "Error"+ Session.Error(result)
End If
```

Reload メソッド

辞書オブジェクトをリフレッシュします。

構文

result = *Object*.**Reload**

引数

Object	DtoDictionary オブジェクト。

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

Delete メソッド

辞書オブジェクトおよび対応する DDF ファイルを削除します。

構文

```
result = Object.Delete
```

引数

Object	DtoDictionary オブジェクト。

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

DtoTables コレクション

DtoTable オブジェクトのコレクションを返します。

プロパティ

Count	コレクション内のメンバー数を返します。
Item	コレクションの特定のメンバーを返します。序数の値またはテーブル名を渡すことができます。

メソッド

なし

備考

このコレクションにはユーザー定義テーブルのみが含まれ、システム テーブルは含まれません。辞書は正常に開かれている必要があります。コレクションにテーブルを追加、あるいはコレクションからテーブルを削除するには、AddTable と DropTable を使用します。

Count プロパティを使用して DtoTables コレクション内のメンバー数を見つけます。

例

DtoDatabase の使用

Tables コレクションを取得する前に、まずデータベース オブジェクトに対して Open メソッドを実行する必要があります。これはそのデータベースにセキュリティが設定されていない場合でも必要です。

```
Dim m_session as new DtoSession
Dim m_database as new DtoDatabase
Dim table as new DtoTable
Dim result as DtoResult

result = m_session.Connect("server","user","password")
m_database.Name = "demodata"
m_database.Session = m_session
' データベースを開く。データベースのセキュリティは設定されていないことを前提とする。
result = m_database.Open("","")

For each table in m_database.Tables
   if table.Name = "Billing" then
      'billing テーブルの検索
   End If
next
m_database.Close ' データベースを開いていた場合は閉じる
```

DtoDictionary の使用

```
Dim dictionary as new DtoDictionary
Dim table as new DtoTable
Dim result as DtoResult
Dim location as string

'Mytable テーブルの場所を検索する
result = dictionary.Open("d:¥MyDemodata")
```

```
For Each table In dictionary.Tables
   If table.Name = "Mytable" Then
      location = table.Location
      exit For
   End If
next
```

関連項目

「DtoDatabase オブジェクト」

「AddTable メソッド」

「DropTable メソッド」

「DtoTable オブジェクト」

DtoTable オブジェクト

データベース内のテーブルを表すオブジェクトです。

プロパティ

Flags	このテーブルに関連付けられているフラグを取得または設定します。可能な値のリストについては、「テーブル フラグ」を参照してください。
Location	テーブルのファイル名を設定します。このファイルのパスを調べるには、「DtoDatabase オブジェクト」のプロパティを使用します。
Name	テーブルの名前を設定または取得します。
Overwrite	True の場合、このテーブルを 「AddTable メソッド」呼び出しの中での同じ名前のテーブルで上書きすることができます。 True = テーブルを上書きします False = テーブルを上書きしないで、エラーを返します

コレクション

「DtoColumns コレクション」

「DtoIndexes コレクション」

メソッド

なし

備考

DtoTable オブジェクトには Columns と Indexes という 2 つのコレクション オブジェクトがあります。 列やインデックスに関連するすべての操作は、これらのオブジェクトを使って行います。

辞書に新しいテーブルを追加するには、「AddTable メソッド」を使用します。

辞書からテーブルを削除するには、「DropTable メソッド」を使用します。

例

DtoTable オブジェクトを新規作成する例については、「AddTable メソッド」を参照してください。

```
Dim dictionary as new DtoDictionary
Dim table as new DtoTable
Dim result as DtoResult
Dim location as string

'Mytable テーブルのファイル名を調べる
result = dictionary.Open("d:¥MyDemodata")

For Each table In dictionary.Tables
   If table.Name = "Mytable" Then
      location = table.Location
   End If
next
```

関連項目

「DtoTables コレクション」

「DtoColumn オブジェクト」

「DtoIndex オブジェクト」

DtoColumns コレクション

テーブル内のすべての列を表す DtoColumn オブジェクトのコレクションです。

プロパティ

Count	コレクション内のメンバー数を返します。
Item	DtoColumns コレクションの特定のメンバーを返します。

メソッド

「Add メソッド」

「Remove メソッド」

「Clear メソッド」

備考

「DtoTable オブジェクト」のプロパティからこのコレクションを取得することができます。

Count プロパティを使用してコレクション内のメンバー数を調べます。

例

```
Dim dictionary as new DtoDictionary
dictionary.Open("d:¥MyDemodata")
students_table = dictionary.GetTable("STUDENT")
students_cols = students_table.Columns
```

関連項目

「DtoIndexes コレクション」

「DtoColumn オブジェクト」

「DtoTable オブジェクト」

メソッドの詳細

Add メソッド

DtoColumns コレクションに項目を追加します。

構文

result = Collection.**Add**(*Object*)

引数

Collection	オブジェクトを追加する DtoColumns コレクション。
Object	新しい DtoColumn オブジェクト。

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

備考

このメソッドは DtoColumn タイプのパラメーターを使用します。このため、コレクションにオブジェクトを追加する前に、まずオブジェクトのインスタンスを作成してそのプロパティを設定する必要があります。

メモ 既存の Zen テーブルに列を追加する場合に、このメソッドを使用することはできません。このメソッドでは、データ ファイルおよび DDF ファイルを変更しません。テーブルを作成する前に、メモリ内で列を追加する場合にのみ使用できます。参考として、「AddTable メソッド」のコード例をご覧ください。

Remove メソッド

DtoColumns コレクションから項目を削除します。

構文

```
result = Collection.Remove(column)
```

引数

Collection	オブジェクトを削除する DtoColumns コレクション。
column	DtoColumns コレクションから削除する項目の（1 から始まる）インデックスまたは項目の名前を含むバリアントを指定できます。

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

備考

DtoColumns コレクション内の項目の列名または 1 から始まる序数を渡すことができます。

メモ 既存の Zen テーブルから列を削除する場合に、このメソッドを使用することはできません。このメソッドでは、データ ファイルおよび DDF ファイルを変更しません。テーブルを作成する前に、メモリ内で列を削除する場合にのみ使用できます。参考として、「AddTable メソッド」のコード例をご覧ください。

Clear メソッド

DtoColumns コレクションからすべての項目を削除します。

構文

result = *Collection*.**Clear**

引数

In	*Collection*	DtoTable オブジェクトから取得する DtoColumns または DtoIndexes コレクション。

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

備考

このメソッドはメモリ内にあるテーブルからすべての列を削除します。

メモ 既存の Zen テーブルから列を削除する場合に、このメソッドを使用することはできません。このメソッドでは、データ ファイルおよび DDF ファイルを変更しません。テーブルを作成する前に、メモリ内ですべての列を削除する場合にのみ使用できます。参考として、「AddTable メソッド」のコード例をご覧ください。

DtoColumn オブジェクト

このオブジェクトはテーブルの列を表します。

プロパティ

Decimal	列の小数以下の桁数を取得または設定します。
Flags	列のフラグを取得または設定します。
ISR	列の ISR を取得または設定します。
Name	列の名前を取得または設定します。
Number	列の序数を取得または設定します。
Size	列のサイズを設定または取得します。
Type	列のタイプを取得または設定します。これは列挙型の値です。可能な値のリストについては、「Btrieve 型」を参照してください。
TypeName	タイプの名前を含む文字列値を返します。

メソッド

なし

備考

このオブジェクトを使用すれば、特定のテーブル列のプロパティを表示することができます。

例

```
' 辞書をインスタンス化して開く
Dim dictionary as new DtoDictionary
Dim result as DtoResult
result = dictionary.Open("d:¥MyDemodata")

'MyDemodata データベースから STUDENT テーブルを取得する
Dim students_table as DtoTable
Set students_table = dictionary.Tables("STUDENTS")

'STUDENT テーブルから Columns コレクションを取得する
Dim students_cols as DtoColumns
Set students_cols = students_table.Columns

' 最初の列を取得し、その名前を取得する
Dim first_col as DtoColumn
Set first_col = students_cols(1)
name = first_col.Name
```

関連項目

「DtoColumns コレクション」

「DtoTable オブジェクト」

DtoIndexes コレクション

テーブルのインデックスを表す DtoIndex オブジェクトのコレクションです。

プロパティ

Count	コレクション内のメンバー数を返します。
Item	DtoIndexes コレクションの特定のメンバーを返します。インデックスの 1 から始まる序数または名前を渡すことができます。

メソッド

「Add メソッド」

「Remove メソッド」

「Clear メソッド」

備考

Count プロパティを使用してコレクション内のメンバー数を調べます。

例

```
' 辞書をインスタンス化して開く
Dim dictionary as new DtoDictionary
Dim result as DtoResult
result = dictionary.Open("d:¥mydemodata")

'MYDEMODATA データベースから STUDENT テーブルを取得する
Dim students_table as DtoTable
Set students_table = dictionary.Tables("STUDENT")

'DEMODATA の Indexes コレクションを取得する
Dim students_idx as DtoIndexes
Set students_idx = students_table.Indexes
```

関連項目

「DtoIndex オブジェクト」

「DtoTable オブジェクト」

メソッドの詳細

Add メソッド

コレクションに項目を追加します。

構文

result = *Collection*.**Add**(*Object*)

引数

Collection	オブジェクトを追加するコレクション。
Object	DtoIndexes コレクションに追加する新しい DtoIndex オブジェクト。

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

備考

このメソッドは DtoIndex タイプのパラメーターを使用します。このため、コレクションにオブジェクトを追加する前に、まずオブジェクトのインスタンスを作成してそのプロパティを設定する必要があります。

メモ 既存の Zen テーブルにインデックスを追加する場合に、このメソッドを使用することはできません。このメソッドでは、データ ファイルおよび DDF ファイルを変更しません。テーブルを作成する前に、メモリ内でインデックスを追加する場合にのみ使用できます。参考として、「AddTable メソッド」のコード例をご覧ください。

Remove メソッド

コレクションから項目を削除します。

構文

```
result = Collection.Remove(index)
```

引数

Collection	オブジェクトを削除するコレクション。
index	DtoIndexes コレクションから削除する項目の（1 から始まる）インデックスまたは項目の名前を含むバリアントを指定できます。

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

備考

Remove メソッドには、項目の名前または 1 から始まる序数を渡すことができます。

メモ 既存の Zen テーブルからインデックスを削除する場合に、このメソッドを使用することはできません。このメソッドでは、データ ファイルおよび DDF ファイルを変更しません。テーブルを作成する前に、メモリ内でインデックスを削除する場合にのみ使用できます。参考として、「AddTable メソッド」のコード例をご覧ください。

Clear メソッド

DtoColumns または DtoIndexes コレクションからすべての項目を削除します。

構文

result = *Collection*.**Clear**

引数

Collection	DtoTable オブジェクトから取得する DtoIndexes コレクション。

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

備考

このメソッドはメモリ内にあるテーブルからすべてのインデックスを削除します。

メモ 既存の Zen テーブルからインデックスを削除する場合に、このメソッドを使用することはできません。このメソッドでは、データ ファイルおよび DDF ファイルを変更しません。テーブルを作成する前に、メモリ内ですべてのインデックスを削除する場合にのみ使用できます。参考として、「AddTable メソッド」のコード例をご覧ください。

DtoIndex オブジェクト

このオブジェクトはテーブルのインデックスを表します。

プロパティ

Flags	インデックス セグメントの列名を取得または設定します。これは列挙型の値です。可能な値のリストについては、「インデックス フラグ」を参照してください。
Name	インデックスの名前を設定または取得します。
Number	インデックスの 0 から始まる番号を取得または設定します。
Tag	インデックス タグを取得します。インデックスを構成するすべての列の名前が含まれます。

コレクション

「DtoSegments コレクション」

メソッド

なし

備考

1 つのインデックスに対し、119 個のセグメントが許可されます。1 つのインデックスのセグメントに含まれるすべての列を結合したサイズは、255 バイトより大きくすることはできないことに注意してください。

インデックス セグメントの中で最後の列のみが部分インデックス フラグを持つことができます。インデックスの最後のセグメントでないのに部分インデックス フラグを使用しているインデックス セグメントでは、部分フラグが無視されます。

例

```
' 辞書をインスタンス化して開く
Dim dictionary as new DtoDictionary
Dim result as DtoResult
result = dictionary.Open("d:¥mydemodata")

'MYDEMODATA データベースから STUDENT テーブルを取得する
Dim students_table as DtoTable
Set students_table = dictionary.Tables("STUDENT")

'DEMODATA の Indexes コレクションを取得する
Dim students_idx as DtoIndexes
Set students_idx = students_table.Indexes

' 最初のインデックスを取得し、その名前を調べる
Dim first_idx as DtoIndex
Set first_idx = students_idx(1)
Dim index_name as String
index_name = first_idx.Name
```

関連項目

「DtoIndexes コレクション」

「DtoSegments コレクション」

DtoSegments コレクション

インデックスのセグメントを表す DtoSegment オブジェクトのコレクションです。

プロパティ

Count	コレクション内のメンバー数を返します。
Item	コレクションの特定のメンバーを返します。

メソッド

「Add メソッド」

「Remove メソッド」

「Clear メソッド」

備考

Count プロパティを使用してコレクション内のメンバー数を調べます。

例

```
' 辞書を開く
Dim dictionary as new DtoDictionary
Dim result as DtoResult
result = dictionary.Open("d:¥mydemodata")

'Students テーブルを取得する
Dim students_table as DtoTable
Set students_table = dictionary.GetTable("Student")

'Students テーブルから Indexes コレクションを取得する
Dim students_idx as DtoIndexes
Set students_idx = students_table.Indexes

' すべてのインデックスを削除する
Dim first_idx as DtoIndex
Set first_idx = students_idx(1)

' 最初のインデックスから DtoSegments コレクションを取得する
Dim my_segments as DtoSegments
Set my_segments as first_idx.Segments
```

関連項目

「DtoSegment オブジェクト」

「DtoTable オブジェクト」

メソッドの詳細

Add メソッド

コレクションに項目を追加します。

構文

```
result = Collection.Add(Object)
```

引数

Collection	オブジェクトを追加する DtoSegments コレクション。
Object	新しい DtoSegment オブジェクト。

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

備考

このメソッドは DtoSegment タイプのパラメーターを使用します。このため、コレクションにオブジェクトを追加する前に、まずオブジェクトのインスタンスを作成してそのプロパティを設定する必要があります。

メモ 既存の Zen テーブルにセグメントを追加する場合に、このメソッドを使用することはできません。このメソッドでは、データ ファイルおよび DDF ファイルを変更しません。テーブルを作成する前に、メモリ内でセグメントを追加する場合にのみ使用できます。参考として、「AddTable メソッド」のコード例をご覧ください。

Remove メソッド

コレクションから項目を削除します。

構文

```
result = Collection.Remove(segment)
```

引数

Collection	オブジェクトを削除する DtoSegments コレクション。
segment	コレクションから削除する項目の（1 から始まる）インデックスまたは項目の名前を含むバリアントを指定できます。

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

備考

セグメントの 1 から始まる序数または名前を渡すことができます。

メモ 既存の Zen テーブルからセグメントを削除する場合に、このメソッドを使用することはできません。このメソッドでは、データ ファイルおよび DDF ファイルを変更しません。テーブルを作成する前に、メモリ内でセグメントを削除する場合にのみ使用できます。参考として、「AddTable メソッド」のコード例をご覧ください。

Clear メソッド

DtoSegments コレクションからすべての項目を削除します。

構文

result = Collection.**Clear**

引数

Collection	DtoIndex オブジェクトから取得する DtoSegments コレクション。

戻り値

result	メソッド呼び出しの結果を示す DtoResult（Long 型の値）。「DtoSession オブジェクト」の Error プロパティを使って結果の説明を取得します。

備考

このメソッドはメモリ内にあるインデックスからすべてのセグメントを削除します。

メモ 既存の Zen テーブルからセグメントを削除する場合に、このメソッドを使用することはできません。このメソッドでは、データ ファイルおよび DDF ファイルを変更しません。テーブルを作成する前に、メモリ内でセグメントを削除する場合にのみ使用できます。参考として、「AddTable メソッド」のコード例をご覧ください。

DtoSegment オブジェクト

このオブジェクトはインデックス内のセグメントを表します。

プロパティ

ColumnName	このセグメントに関連付けられている列名を取得または設定します。
Flags	セグメント フラグを取得または設定します。これは列挙型の値です。可能な値のリストについては、「セグメント フラグ」を参照してください。
Number	0 から始まるセグメント番号を取得または設定します。

メソッド

なし

備考

1 つまたは複数のセグメントでインデックスが構成されます。1 つのインデックスに対し、119 個のセグメントが許可されます。1 つのインデックスのセグメントに含まれるすべての列を結合したサイズは、255 バイトより大きくすることはできないことに注意してください。

例

```
' 辞書を開く
Dim dictionary as new DtoDictionary
Dim result as DtoResult
result = dictionary.Open("d:¥mydemodata")

'Students テーブルを取得する
Dim students_table as DtoTable
Set students_table = dictionary.GetTable("Student")

'Students テーブルから Indexes コレクションを取得する
Dim students_idx as DtoIndexes
Set students_idx = students_table.Indexes

' すべてのインデックスを削除する
Dim first_idx as DtoIndex
Set first_idx = students_idx(1)

' 最初のインデックスから DtoSegments コレクションを取得する
Dim my_segments as DtoSegments
Set my_segments as first_idx.Segments

' 最初のセグメントを取得し、列名を照会する
Dim first_seg as DtoSegment
Set first_seg = my_segments(1)
Dim colname as String
colname = first_seg.ColumnName
```

関連項目

「DtoSegments コレクション」

「DtoTable オブジェクト」

「DtoIndexes コレクション」

Distributed Tuning Objects 列挙

6

Zen Distributed Tuning Objects の列挙

この章では、Distributed Tuning Objects で使用する列挙について説明します。

DTO の列挙型

DTO は以下の列挙型をサポートしています。

- 「Btrieve 型」
- 「列フラグ」
- 「インデックス フラグ」
- 「セグメント フラグ」
- 「テーブル フラグ」
- 「DtoResult」
- 「設定ランク」
- 「設定タイプ」
- 「クライアント サイト」
- 「クライアント プラットフォーム」
- 「トランザクション タイプ」
- 「オープン モード」
- 「DSN オープン モード」
- 「DSN 変換オプション」
- 「ロック タイプ」
- 「ウェイト状態」
- 「データベース コード ページ」
- 「データベース フラグ」
- 「SQL 接続状態」
- 「サービス ID」
- 「サービス状態」

Btrieve 型

列挙	値
0	dtoTypeString
1	dtoTypeInteger
2	dtoTypeFloat
3	dtoTypeDate
4	dtoTypeTime
5	dtoTypeDecimal
6	dtoTypeMoney
7	dtoTypeLogical
8	dtoTypeNumeric
9	dtoTypeBfloat

列挙	値
10	dtoTypeLString
11	dtoTypeZString
12	dtoTypeNote
13	dtoTypeLvar
14	dtoTypeBinary
15	dtoTypeIdentity
16	dtoTypeBit
17	dtoTypeNumericSTS
18	dtoTypeNumericSA
19	dtoTypeCurrency
20	dtoTypeTimestamp
21	dtoTypeBlob
22	dtoTypeGDecimal
25	dtoTypeWString
26	dtoTypeWZString
27	dtoTypeGUID
30	dtoTypeDateTime

列フラグ

列挙	値
0	dtoColumnDefault
1	dtoColumnCaseSensitive
4	dtoColumnNullable
256	dtoColumnBinary
2048	dtoTypeColumnNText
4096	dtoTypeColumnBinary

インデックス フラグ

列挙	値
0	dtoIndexDefault
1	dtoIndexDuplicatesAllowed
2	dtoIndexModifiable
64	dtoIndexDescending
512	dtoIndexPartial

セグメント フラグ

列挙	値
0	dtoSegmentAscending
64	dtoSegmentDescending

テーブル フラグ

列挙	値
0	dtoTableLegacy
64	dtoTableTrueNullable

DtoResult

列挙	値
0	Dto_Success
1	Dto_errFailed
2	Dto_errMemoryAllocation
3	Dto_errDictionaryNotFound
4	Dto_errDictionaryAlreadyOpen
5	Dto_errDictionaryNotOpen
6	Dto_errInvalidDictionaryHandle
7	Dto_errTableNotFound
8	Dto_errInvalidTableName

列挙	値
9	Dto_errInvalidColumnName
10	Dto_errInvalidColumnDataType
11	Dto_errDuplicateColumnName
12	Dto_errInvalidDataSize
13	Dto_errInvalidColumnOrder
14	Dto_errInvalidIndexName
15	Dto_errColumnNotFound
16	Dto_errTooManySegments
17	Dto_errStringTooShort
18	Dto_errDictionaryAlreadyExists
19	Dto_errDirectoryError
20	Dto_errSessionSecurityError
21	Dto_errDuplicateTable
22	Dto_errDuplicateIndex
27	Dto_errInvalidNameLength
28	Dto_errInternalProtocolError
29	Dto_errInvalidAccountName
30	Dto_errUserAlreadyExists
31	Dto_errGroupNotEmpty
32	Dto_errGroupAlreadyExists
33	Dto_errUserAlreadyPartOfGroup
34	Dto_errUserNotPartOfGroup
35	Dto_errNotAllowedToDropAdministrator
36	Dto_errDatabaseHasNoSecurity
37	Dto_errInvalidPassword
38	Dto_SuccessWithInfo
87	Dto_errServiceInvalidParameter
123	Dto_errInvalidServiceName
161	Dto_errMaxUserCountReached
423	Dto_errInvalidSession
424	Dto_errInvalidArgument

列挙	値
425	Dto_errNotConnected
426	Dto_errInvalidComputerName
427	Dto_errUnknownError
428	Dto_errTableCouldNotBeDeleted
429	Dto_errItemNotFound
430	Dto_errAPINotImplemented
431	Dto_errAccessDenied
1051	Dto_errServiceDependentServiceRunning
1052	Dto_errServiceInvalidServiceControl
1053	Dto_errServiceRequestTimeout
1055	Dto_errServiceDatabaseLocked
1056	Dto_errServiceAlreadyRunning
1057	Dto_errInvalidServiceAccount
1058	Dto_errServiceDisabled
1059	Dto_errServiceCircularDependency
1060	Dto_errServiceDoesNotExist
1062	Dto_errServiceNotActive
1065	Dto_errServiceDatabaseDoesNotExist
1068	Dto_errServiceDependencyFail
1069	Dto_errServiceLogonFailed
1072	Dto_errServiceMarkedForDelete
1075	Dto_errServiceDependencyDeleted
7001	Dto_errInvalidHandle
7002	Dto_errNullPointer
7003	Dto_errBufferTooSmall
7004	Dto_errDtiFailed
7005	Dto_errInvalidDataType
7006	Dto_errOutOfRange
7007	Dto_errInvalidSelection
7008	Dto_errInvalidSequence
7009	Dto_errDataUnavailable

列挙	値
7010	Dto_errInvalidClient
7011	Dto_errAccessRights
7012	Dto_errDuplicateName
7013	Dto_errDatabaseDoesNotExist
7015	Dto_errFileNotOpen
7016	Dto_errDDFAlreadyExist
7017	Dto_errSharedDDFExist
7018	Dto_errInvalidName
7019	Dto_errDSNAlreadyExist
7020	Dto_errDSNDoesNotExist
7021	Dto_errInvalidOpenMode
以下の列挙は、「DTO2」でのみ存在します。	
7063	「161」を参照してください。
7064	Dto_errNoLicenseObtained
7065	Dto_errNoProductObtained
7101	Dto_errInvalidLicKeyCharacter
7102	Dto_errIllegalLicType
7108	Dto_errLicKeyTooLong
7109	Dto_errLicNotFound
7110	Dto_errLicExpired
7111	Dto_errLicIsTemporary
7112	Dto_errLicAlreadyInstalled
7113	Dto_errLicInvalid
7115	Dto_errInvalidProductId
7118	Dto_errServerNotRunning
7119	Dto_errLocalServerNotRunning
7120	Dto_errLicNotRemovable
7122	Dto_errNoActiveLicense

設定ランク

列挙	値
0	dtoNormal
1	dtoAdvanced

設定タイプ

列挙	値
0	dtoBooleanType
1	dtoLongType
2	dtoStringType
3	dtoSingleSel
4	dtoMultiSel

クライアント サイト

列挙	値
0	dtoClientSiteLocal
1	dtoClientSiteRemote

クライアント プラットフォーム

列挙	値
0	dtoPlatformNotAvailable
1	dtoPlatformWin
2	dtoPlatformWin95
3	dtoPlatformWinWg
4	dtoPlatformNTW
5	dtoPlatformNTS
6	dtoPlatformNW
7	dtoPlatformOS2W

列挙	値
8	dtoPlatformOS2S
9	dtoPlatformDOS

トランザクション タイプ

列挙	値
0	dtoNone
19	dtoExclusive
1019	dtoConcurrent

オープン モード

列挙	値
0	dtoNormalOpenMode
255	dtoAcceleratedOpenMode
254	dtoReadOnlyOpenMode
253	dtoVerifyOpenMode
252	dtoExclusiveOpenMode
248	dtoNormalNonTransOpenMode
247	dtoAcceleratedNonTransOpenMode
246	dtoReadOnlyNonTransOpenMode
245	dtoVerifyNonTransOpenMode
244	dtoExclusiveNonTransOpenMode
240	dtoNormalSharedLockingOpenMode
239	dtoAcceleratedSharedLockingOpenMode
238	dtoReadOnlySharedLockingOpenMode
237	dtoVerifySharedLockingOpenMode
236	dtoExclusiveSharedLockingOpenMode

DSN オープン モード

列挙	値
0	dtoNormalDSNOpenMode
1	dtoAcceleratedDSNOpenMode
2	dtoReadOnlyDSNOpenMode
3	dtoExclusiveDSNOpenMode

DSN 変換オプション

列挙	値
0	dtoDSNFlagDefault
1	dtoDSNFlagEomAnsi
2	dtoDSNFlagAuto

ロック タイプ

列挙	値
0	dtoNotLocked
1	dtoSingleLock
2	dtoMultipleLock

ウェイト状態

列挙	値
0	dtoNotWaiting
1	dtoWaitingForRecordLock
2	dtoWaitingForFileLock

データベース コード ページ

列挙	値
0	dtoDbZeroCodePage
65001	dtoDBCodePageUTF8

データベース フラグ

列挙	値
0	dtoDbFlagNotApplicable
1	dtoDbFlagBound
2	dtoDbFlagRI
4	dtoDbFlagCreateDDF
32	dtoDbFlagLONGMETADATA

SQL 接続状態

列挙	値
0	dtoSQLConnectionIdle
1	dtoSQLConnectionActive
2	dtoSQLConnectionDying

サービス ID

列挙	値
0	dtoServiceTransactional
1	dtoServiceRelational
2	dtoServiceIDS

サービス状態

列挙	値
0	dtoServiceStopped
1	dtoServiceStartPending
2	dtoServiceStopPending
3	dtoServiceRunning
4	dtoServiceContinuePending
5	dtoServicePausePending
6	dtoServicePaused
7	dtoServiceNotFound