SYBASE iAnywhere

Ultra Light .NET プログラミング

2009 年 2 月

バージョン 11.0.1

版権と商標

目次

はじめに vii

SQL Anywhere のマニュアルについて viii

Ultra Light.NET の概要 1

Ultra Light と .NET 2
システムの稼働条件とサポートされるプラットフォーム 3
Ultra Light.NET アーキテクチャ 4

Ultra Light 用 SQL Anywhere Explorer 5

SQL Anywhere Explorer の概要 6
Ultra Light プロジェクトでの SQL Anywhere Explorer の使用 7

Ultra Light.NET 開発の概要 11

Visual Studio .NET での SQL Anywhere ツールの使用 12
データベースへの接続 13
暗号化と難読化 15
SQL を使用したデータへのアクセスと操作 16
テーブル API によるデータ・アクセスと操作 20
トランザクションの管理 26
スキーマ情報へのアクセス 27
エラー処理 28
ユーザの認証 29
Ultra Light アプリケーションの同期 30

チュートリアル：Ultra Light.NET アプリケーションのビルド 33

Ultra Light.NET 開発のチュートリアル 34
レッスン 1：Visual Studio プロジェクトの作成 35
レッスン 2：Ultra Light データベースの作成 39
レッスン 3 ：データベースへの接続 41

レッスン 4：データの挿入、更新、削除 43
レッスン 5：アプリケーションのビルドと配置 48
C# チュートリアルのコード・リスト 50
Visual Basic チュートリアルのコード・リスト 53

Ultra Light .NET 2.0 API リファレンス 55

ULActiveSyncListener インタフェース 57
ULAuthStatusCode 列挙体 61
ULBulkCopy クラス 62
ULBulkCopyColumnMapping クラス 74
ULBulkCopyColumnMappingCollection クラス 81
ULBulkCopyOptions 列挙体 90
ULCommand クラス 91
ULCommandBuilder クラス 129
ULConnection クラス 139
ULConnectionParms クラス 189
ULConnectionParms.UnusedEventHandler デリゲート 202
ULConnectionStringBuilder クラス 203
ULCreateParms クラス 221
ULCursorSchema クラス 233
ULDataAdapter クラス 242
ULDatabaseManager クラス 255
ULDatabaseSchema クラス 265
ULDataReader クラス 276
ULDateOrder 列挙体 316
ULDbType 列挙体 317
ULDBValid 列挙体 321
ULException クラス 322
ULFactory クラス 326
ULFileTransfer クラス 332
ULFileTransferProgressData クラス 348
ULFileTransferProgressListener インタフェース 351
ULIndexSchema クラス 353
ULInfoMessageEventArgs クラス 362

ULInfoMessageEventHandler デリゲート 365
ULMetaDataCollectionNames クラス 366
ULParameter クラス 375
ULParameterCollection クラス 392
ULPublicationSchema クラス 412
ULResultSet クラス 415
ULResultSetSchema クラス 442
ULRowsCopiedEventArgs クラス 445
ULRowsCopiedEventHandler デリゲート 448
ULRowUpdatedEventArgs クラス 449
ULRowUpdatedEventHandler デリゲート 453
ULRowUpdatingEventArgs クラス 454
ULRowUpdatingEventHandler デリゲート 457
ULRuntimeType 列挙体 458
ULServerSyncListener インタフェース 459
ULSQLCode 列挙体 462
ULSqlPassthroughProgressListener インタフェース 477
ULSqlProgressData クラス 479
ULSqlProgressState 列挙体 481
ULStreamErrorCode 列挙体 482
ULStreamType 列挙体 512
ULSyncParms クラス 514
ULSyncProgressData クラス 528
ULSyncProgressListener インタフェース 538
ULSyncProgressState 列挙体 540
ULSyncResult クラス 542
ULTable クラス 548
ULTableSchema クラス 571
ULTransaction クラス 586

索引 591

はじめに

このマニュアルの内容

このマニュアルでは、Ultra Light.NET について説明します。Ultra Light.NET を使用すると、PC、ハンドヘルド・デバイス、モバイル・デバイス、または埋め込みデバイスのデータベース・アプリケーションを開発し、これらのデバイスに配備できます。

対象読者

このマニュアルは、Ultra Light リレーショナル・データベースのパフォーマンス、リソース効率、堅牢性、セキュリティを利用してデータを格納、同期することを目的とする .NET アプリケーション開発者を対象にしています。

SQL Anywhere のマニュアルについて

SQL Anywhere の完全なマニュアルは 4 つの形式で提供されており、いずれも同じ情報が含まれています。

- **HTML ヘルプ**　オンライン・ヘルプには、SQL Anywhere の完全なマニュアルがあり、SQL Anywhere ツールに関する印刷マニュアルとコンテキスト別のヘルプの両方が含まれています。

  Microsoft Windows オペレーティング・システムを使用している場合は、オンライン・ヘルプは HTML ヘルプ (CHM) 形式で提供されます。マニュアルにアクセスするには、**[スタート] - [プログラム] - [SQL Anywhere 11] - [マニュアル] - [オンライン・マニュアル]** を選択します。

  管理ツールのヘルプ機能でも、同じオンライン・マニュアルが使用されます。

- **Eclipse**　UNIX プラットフォームでは、完全なオンライン・ヘルプは Eclipse 形式で提供されます。マニュアルにアクセスするには、SQL Anywhere 11 インストール環境の *bin32* または *bin64* ディレクトリから *sadoc* を実行します。

- **DocCommentXchange**　DocCommentXchange は、SQL Anywhere マニュアルにアクセスし、マニュアルについて議論するためのコミュニティです。

  DocCommentXchange は次の目的に使用できます (現在のところ、日本語はサポートされておりません)。

  ○ マニュアルを表示する

  ○ マニュアルの項目について明確化するために、ユーザによって追加された内容を確認する

  ○ すべてのユーザのために、今後のリリースでマニュアルを改善するための提案や修正を行う

  http://dcx.sybase.com を参照してください。

- **PDF**　SQL Anywhere の完全なマニュアル・セットは、Portable Document Format (PDF) 形式のファイルとして提供されます。内容を表示するには、PDF リーダが必要です。Adobe Reader をダウンロードするには、http://get.adobe.com/reader/ にアクセスしてください。

  Microsoft Windows オペレーティング・システムで PDF マニュアルにアクセスするには、**[スタート] - [プログラム] - [SQL Anywhere 11] - [マニュアル] - [オンライン・マニュアル - PDF]** を選択します。

  UNIX オペレーティング・システムで PDF マニュアルにアクセスするには、Web ブラウザを使用して *install-dir/documentation/ja/pdf/index.html* を開きます。

マニュアル・セットに含まれる各マニュアルについて

SQL Anywhere のマニュアルは次の構成になっています。

- **『SQL Anywhere 11 - 紹介』**　このマニュアルでは、データの管理および交換機能を提供する包括的なパッケージである SQL Anywhere 11 について説明します。SQL Anywhere を使用す

ると、サーバ環境、デスクトップ環境、モバイル環境、リモート・オフィス環境に適したデータベース・ベースのアプリケーションを迅速に開発できるようになります。

- **『SQL Anywhere 11 - 変更点とアップグレード』** このマニュアルでは、SQL Anywhere 11 とそれ以前のバージョンに含まれる新機能について説明します。
- **『SQL Anywhere サーバ - データベース管理』** このマニュアルでは、SQL Anywhere データベースを実行、管理、構成する方法について説明します。データベース接続、データベース・サーバ、データベース・ファイル、バックアップ・プロシージャ、セキュリティ、高可用性、Replication Server を使用したレプリケーション、管理ユーティリティとオプションについて説明します。
- **『SQL Anywhere サーバ - プログラミング』** このマニュアルでは、C、C++、Java、PHP、Perl、Python、および Visual Basic や Visual C# などの .NET プログラミング言語を使用してデータベース・アプリケーションを構築、配備する方法について説明します。ADO.NET や ODBC などのさまざまなプログラミング・インタフェースについても説明します。
- **『SQL Anywhere サーバ - SQL リファレンス』** このマニュアルでは、システム・プロシージャとカタログ (システム・テーブルとビュー) に関する情報について説明します。また、SQL Anywhere での SQL 言語の実装 (探索条件、構文、データ型、関数) についても説明します。
- **『SQL Anywhere サーバ - SQL の使用法』** このマニュアルでは、データベースの設計と作成の方法、データのインポート・エクスポート・変更の方法、データの検索方法、ストアド・プロシージャとトリガの構築方法について説明します。
- **『Mobile Link - クイック・スタート』** このマニュアルでは、セッションベースのリレーショナル・データベース同期システムである Mobile Link について説明します。Mobile Link テクノロジは、双方向レプリケーションを可能にし、モバイル・コンピューティング環境に非常に適しています。
- **『Mobile Link - クライアント管理』** このマニュアルでは、Mobile Link クライアントを設定、構成、同期する方法について説明します。Mobile Link クライアントには、SQL Anywhere または Ultra Light のいずれかのデータベースを使用できます。また、dbmlsync API についても説明します。dbmlsync API を使用すると、同期を C++ または .NET のクライアント・アプリケーションにシームレスに統合できます。
- **『Mobile Link - サーバ管理』** このマニュアルでは、Mobile Link アプリケーションを設定して管理する方法について説明します。
- **『Mobile Link - サーバ起動同期』** このマニュアルでは、Mobile Link サーバ起動同期について説明します。この機能により、Mobile Link サーバは同期を開始したり、リモート・デバイス上でアクションを実行することができます。
- **『QAnywhere』** このマニュアルでは、モバイル・クライアント、ワイヤレス・クライアント、デスクトップ・クライアント、およびラップトップ・クライアント用のメッセージング・プラットフォームである、QAnywhere について説明します。
- **『SQL Remote』** このマニュアルでは、モバイル・コンピューティング用の SQL Remote データ・レプリケーション・システムについて説明します。このシステムによって、SQL Anywhere の統合データベースと複数の SQL Anywhere リモート・データベースの間で、電子メールやファイル転送などの間接的リンクを使用したデータ共有が可能になります。

- **『Ultra Light - データベース管理とリファレンス』** このマニュアルでは、小型デバイス用 Ultra Light データベース・システムの概要を説明します。
- **『Ultra Light - C/C++ プログラミング』** このマニュアルでは、Ultra Light C および Ultra Light C++ のプログラミング・インタフェースについて説明します。Ultra Light を使用すると、ハンドヘルド・デバイス、モバイル・デバイス、埋め込みデバイスのデータベース・アプリケーションを開発し、これらのデバイスに配備できます。
- **『Ultra Light - M-Business Anywhere プログラミング』** このマニュアルは、Ultra Light for M-Business Anywhere について説明します。Ultra Light for M-Business Anywhere を使用すると、Palm OS、Windows Mobile、または Windows を搭載しているハンドヘルド・デバイス、モバイル・デバイス、または埋め込みデバイスの Web ベースのデータベース・アプリケーションを開発し、これらのデバイスに配備できます。
- **『Ultra Light - .NET プログラミング』** このマニュアルでは、Ultra Light.NET について説明します。Ultra Light.NET を使用すると、PC、ハンドヘルド・デバイス、モバイル・デバイス、または埋め込みデバイスのデータベース・アプリケーションを開発し、これらのデバイスに配備できます。
- **『Ultra Light J』** このマニュアルでは、Ultra Light J について説明します。Ultra Light J を使用すると、Java をサポートしている環境用のデータベース・アプリケーションを開発し、配備することができます。Ultra Light J は、BlackBerry スマートフォンと Java SE 環境をサポートしており、iAnywhere Ultra Light データベース製品がベースになっています。
- **『エラー・メッセージ』** このマニュアルでは、SQL Anywhere エラー・メッセージの完全なリストを示し、その診断情報を説明します。

表記の規則

この項では、このマニュアルで使用されている表記規則について説明します。

オペレーティング・システム

SQL Anywhere はさまざまなプラットフォームで稼働します。ほとんどの場合、すべてのプラットフォームで同じように動作しますが、いくつかの相違点や制限事項があります。このような相違点や制限事項は、一般に、基盤となっているオペレーティング・システム (Windows、UNIX など) に由来しており、使用しているプラットフォームの種類 (AIX、Windows Mobile など) またはバージョンに依存していることはほとんどありません。

オペレーティング・システムへの言及を簡素化するために、このマニュアルではサポートされているオペレーティング・システムを次のようにグループ分けして表記します。

- **Windows** Microsoft Windows ファミリを指しています。これには、主にサーバ、デスクトップ・コンピュータ、ラップトップ・コンピュータで使用される Windows Vista や Windows XP、およびモバイル・デバイスで使用される Windows Mobile が含まれます。

  特に記述がないかぎり、マニュアル中に Windows という記述がある場合は、Windows Mobile を含むすべての Windows ベース・プラットフォームを指しています。

- **UNIX**　特に記述がないかぎり、マニュアル中に UNIX という記述がある場合は、Linux および Mac OS X を含むすべての UNIX ベース・プラットフォームを指しています。

ディレクトリとファイル名

ほとんどの場合、ディレクトリ名およびファイル名の参照形式はサポートされているすべてのプラットフォームで似通っており、それぞれの違いはごくわずかです。このような場合は、Windows の表記規則が使用されています。詳細がより複雑な場合は、マニュアルにすべての関連形式が記載されています。

ディレクトリ名とファイル名の表記を簡素化するために使用されている表記規則は次のとおりです。

- **大文字と小文字のディレクトリ名**　Windows と UNIX では、ディレクトリ名およびファイル名には大文字と小文字が含まれている場合があります。ディレクトリやファイルが作成されると、ファイル・システムでは大文字と小文字の区別が維持されます。

  Windows では、ディレクトリおよびファイルを参照するとき、大文字と小文字は**区別されません**。大文字と小文字を混ぜたディレクトリ名およびファイル名は一般的に使用されますが、参照するときはすべて小文字を使用するのが通常です。SQL Anywhere では、*Bin32* や *Documentation* などのディレクトリがインストールされます。

  UNIX では、ディレクトリおよびファイルを参照するとき、大文字と小文字は**区別されます**。大文字と小文字を混ぜたディレクトリ名およびファイル名は一般的に使用されません。ほとんどの場合は、すべて小文字の名前が使用されます。SQL Anywhere では、*bin32* や *documentation* などのディレクトリがインストールされます。

  このマニュアルでは、ディレクトリ名に Windows の形式を使用しています。ほとんどの場合、大文字と小文字が混ざったディレクトリ名をすべて小文字に変換すると、対応する UNIX 用のディレクトリ名になります。

- **各ディレクトリおよびファイル名を区切るスラッシュ**　マニュアルでは、ディレクトリの区切り文字に円記号を使用しています。たとえば、PDF 形式のマニュアルは *install-dir¥Documentation¥ja¥pdf* にあります。これは Windows の形式です。

  UNIX では、円記号をスラッシュに置き換えます。PDF マニュアルは *install-dir/documentation/ja/pdf* にあります。

- **実行ファイル**　マニュアルでは、実行ファイルの名前は、Windows の表記規則が使用され、*.exe* や *.bat* などの拡張子が付きます。UNIX では、実行ファイルの名前に拡張子は付きません。

  たとえば、Windows でのネットワーク・データベース・サーバは *dbsrv11.exe* です。UNIX では *dbsrv11* です。

- ***install-dir***　インストール・プロセス中に、SQL Anywhere をインストールするロケーションを選択します。このロケーションを参照する環境変数 SQLANY11 が作成されます。このマニュアルでは、そのロケーションを *install-dir* と表します。

  たとえば、マニュアルではファイルを install-dir¥readme.txt のように参照します。これは、Windows では、%SQLANY11%¥readme.txt に対応します。UNIX では、$SQLANY11/readme.txt または ${SQLANY11}/readme.txt に対応します。

install-dir のデフォルト・ロケーションの詳細については、「SQLANY11 環境変数」『SQL Anywhere サーバ - データベース管理』を参照してください。

- ***samples-dir*** インストール・プロセス中に、SQL Anywhere に含まれるサンプルをインストールするロケーションを選択します。このロケーションを参照する環境変数 SQLANYSAMP11 が作成されます。このマニュアルではそのロケーションを *samples-dir* と表します。

  Windows エクスプローラ・ウィンドウで *samples-dir* を開くには、**[スタート] - [プログラム] - [SQL Anywhere 11] - [サンプル・アプリケーションとプロジェクト]** を選択します。

  samples-dir のデフォルト・ロケーションの詳細については、「SQLANYSAMP11 環境変数」『SQL Anywhere サーバ - データベース管理』を参照してください。

コマンド・プロンプトとコマンド・シェル構文

ほとんどのオペレーティング・システムには、コマンド・シェルまたはコマンド・プロンプトを使用してコマンドおよびパラメータを入力する方法が、1 つ以上あります。Windows のコマンド・プロンプトには、コマンド・プロンプト (DOS プロンプト) および 4NT があります。UNIX のコマンド・シェルには、Korn シェルおよび bash があります。各シェルには、単純コマンドからの拡張機能が含まれています。拡張機能は、特殊文字を指定することで起動されます。特殊文字および機能は、シェルによって異なります。これらの特殊文字を誤って使用すると、多くの場合、構文エラーや予期しない動作が発生します。

このマニュアルでは、一般的な形式のコマンド・ラインの例を示します。これらの例に、シェルにとって特別な意味を持つ文字が含まれている場合、その特定のシェル用にコマンドを変更することが必要な場合があります。このマニュアルではコマンドの変更について説明しませんが、通常、その文字を含むパラメータを引用符で囲むか、特殊文字の前にエスケープ文字を記述します。

次に、プラットフォームによって異なるコマンド・ライン構文の例を示します。

- **カッコと中カッコ** 一部のコマンド・ライン・オプションは、詳細な値を含むリストを指定できるパラメータを要求します。リストは通常、カッコまたは中カッコで囲まれています。このマニュアルでは、カッコを使用します。次に例を示します。

  ```
  -x tcpip(host=127.0.0.1)
  ```

  カッコによって構文エラーになる場合は、代わりに中カッコを使用します。

  ```
  -x tcpip{host=127.0.0.1}
  ```

  どちらの形式でも構文エラーになる場合は、シェルの要求に従ってパラメータ全体を引用符で囲む必要があります。

  ```
  -x "tcpip(host=127.0.0.1)"
  ```

- **引用符** パラメータの値として引用符を指定する必要がある場合、その引用符はパラメータを囲むために使用される通常の引用符と競合する可能性があります。たとえば、値に二重引用符を含む暗号化キーを指定するには、キーを引用符で囲み、パラメータ内の引用符をエスケープします。

  ```
  -ek "my ¥"secret¥" key"
  ```


多くのシェルでは、キーの値は my "secret" key のようになります。

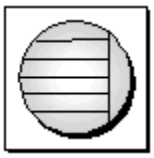
- **環境変数**　マニュアルでは、環境変数設定が引用されます。Windows のシェルでは、環境変数は構文 *%ENVVAR%* を使用して指定されます。UNIX のシェルでは、環境変数は構文 *$ENVVAR* または *${ENVVAR}* を使用して指定されます。

グラフィック・アイコン

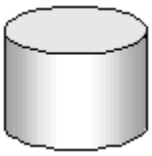
このマニュアルでは、次のアイコンを使用します。

- クライアント・アプリケーション。

- SQL Anywhere などのデータベース・サーバ。



- データベース。ハイレベルの図では、データベースとデータベースを管理するデータ・サーバの両方をこのアイコンで表します。



- レプリケーションまたは同期のミドルウェア。ソフトウェアのこれらの部分は、データベース間のデータ共有を支援します。たとえば、Mobile Link サーバ、SQL Remote Message Agent などが挙げられます。

- プログラミング・インタフェース。

ドキュメンテーション・チームへのお問い合わせ

このヘルプに関するご意見、ご提案、フィードバックをお寄せください。

SQL Anywhere ドキュメンテーション・チームへのご意見やご提案は、弊社までご連絡ください。頂戴したご意見はマニュアルの向上に役立たせていただきます。ぜひとも、ご意見をお寄せください。

DocCommentXchange

DocCommentXchange を使用して、ヘルプ・トピックに関するご意見を直接お寄せいただくこともできます。DocCommentXchange (DCX) は、SQL Anywhere マニュアルにアクセスしたり、マニュアルについて議論するためのコミュニティです。DocCommentXchange は次の目的に使用できます (現在のところ、日本語はサポートされておりません)。

- マニュアルを表示する
- マニュアルの項目について明確化するために、ユーザによって追加された内容を確認する
- すべてのユーザのために、今後のリリースでマニュアルを改善するための提案や修正を行う

http://dcx.sybase.com を参照してください。

詳細情報の検索／テクニカル・サポートの依頼

詳しい情報やリソースについては、iAnywhere デベロッパー・コミュニティ (http://www.iAnywhere.jp/developers/index.html) を参照してください。

ご質問がある場合や支援が必要な場合は、次に示す Sybase iAnywhere ニュースグループのいずれかにメッセージをお寄せください。

ニュースグループにメッセージをお送りいただく際には、ご使用の SQL Anywhere バージョンのビルド番号を明記し、現在発生している問題について詳しくお知らせくださいますようお願いいたします。バージョンおよびビルド番号を調べるには、コマンド **dbeng11 -v** を実行します。

ニュースグループは、ニュース・サーバ *forums.sybase.com* にあります。

以下のニュースグループがあります。

- ianywhere.public.japanese.general

Web 開発に関する問題については、http://groups.google.com/group/sql-anywhere-web-development を参照してください。

ニュースグループに関するお断り
iAnywhere Solutions は、ニュースグループ上に解決策、情報、または意見を提供する義務を負うものではありません。また、システム・オペレータ以外のスタッフにこのサービスを監視させて、操作状況や可用性を保証する義務もありません。

iAnywhere のテクニカル・アドバイザとその他のスタッフは、時間のある場合にかぎりニュースグループでの支援を行います。こうした支援は基本的にボランティアで行われるため、解決策や情報を定期的に提供できるとはかぎりません。支援できるかどうかは、スタッフの仕事量に左右されます。

Ultra Light.NET の概要

目次

Ultra Light と .NET **2**
システムの稼働条件とサポートされるプラットフォーム **3**
Ultra Light.NET アーキテクチャ **4**

Ultra Light と .NET

Ultra Light.NET アプリケーションは、Windows Mobile および Windows に配備できます。Windows Mobile に配備する場合、Ultra Light.NET は .NET Compact Framework を必要とします。Windows に配備する場合は、.NET Framework を必要とします。Ultra Light.NET では、ActiveSync 同期もサポートしています。

.NET Compact Framework は、Windows Mobile 用の Microsoft .NET ランタイム・コンポーネントです。複数のプログラミング言語をサポートします。Visual Basic.NET または C# を使用して、Ultra Light.NET を使用したアプリケーションを構築できます。

Ultra Light.NET には、次のネームスペースが用意されています。

- **iAnywhere.Data.UltraLite** このネームスペースには、Ultra Light への ADO.NET インタフェースが用意されています。業界標準モデルに基づいて構築でき、非常によく似た SQL Anywhere ADO.NET インタフェースへのマイグレーション・パスが提供されるという利点があります。

iAnywhere.Data.UltraLite ネームスペース (.NET 2.0)

この章では、.NET Framework 2.0 と .NET Compact Framework 2.0 用 Ultra Light.NET データ・プロバイダの API について説明します。

SQL Anywhere の ADO.NET 用データ・プロバイダで利用できない Ultra Light.NET 拡張については、この API リファレンスでは **UL 拡張：** と示します。

Ultra Light.NET の Ultra Light エンジンのランタイムを使用するには、「RuntimeType プロパティ」 256 ページを適切な値に設定してから、他の Ultra Light.NET API を使用します。

データベースで操作を実行するには、アプリケーションは接続を開く必要があります。接続を開くには、「ULConnection クラス」 139 ページを使用します。

iAnywhere.Data.UltraLite アセンブリは、**iAnywhere.Data.UltraLite.resources** というサテライト・リソース・アセンブリを使用します。メイン・アセンブリは、このリソース・アセンブリを、CultureInfo.CurrentUICulture、CultureInfo.CurrentCulture、国 **EN** の順序で国別に検索します。

この章の多くのプロパティとメソッドは、.NET Framework Data Provider for OLE DB (System.Data.OleDb) とよく似ています。詳細と例については、Microsoft .NET Framework のマニュアルを参照してください。

システムの稼働条件とサポートされるプラットフォーム

開発プラットフォーム

Ultra Light.NET を使用してアプリケーションを開発するための要件は次のとおりです。

- サポートされているデスクトップ・バージョンの Microsoft Windows
- Microsoft Visual Studio 2005 または Visual Studio 2008
- Windows Mobile デバイスを使用する場合は、.NET Compact Framework バージョン 2 以降

ターゲット・プラットフォーム

Ultra Light.NET でサポートしているターゲット・プラットフォームは次のとおりです。

- Microsoft .NET Compact Framework 2.0 と .NET Framework 2.0
- Windows Mobile デバイスを使用する場合は、Microsoft .NET Compact Framework バージョン 2 以降
- Windows 上で動作する Microsoft .NET Compact Framework バージョン 2 以降

アプリケーション・コードと .NET Compact Framework に加えて、次のファイルを Windows Mobile デバイスまたは Windows がインストールされているコンピュータに配備してください。

- **iAnywhere.Data.UltraLite.dll** iAnywhere.Data.UltraLite ADO.NET ネームスペースを含むアセンブリ。このファイルは、アプリケーションで iAnywhere.Data.UltraLite ネームスペースを使用する場合のみ必要です。
- **iAnywhere.Data.UltraLite.resources.dll** iAnywhere.Data.UltraLite ADO.NET ネームスペースで必要とされるリソース。このファイルは、アプリケーションで iAnywhere.Data.UltraLite ネームスペースを使用する場合のみ必要です。
- **ulnet11.dll** このファイルには、1 つのクライアントが使用する Ultra Light ランタイムが含まれてます。このランタイムの異なるバージョンが、ターゲット・プラットフォームごとに用意されています。
- **ulnetclient11.dll** このファイルには、Ultra Light エンジンへのインタフェースが含まれており、複数のクライアントがエンジンにアクセスするのに使用されます。

Ultra Light がサポートされているプラットフォームの詳細については、http://www.ianywhere.jp/developers/technotes/os_components_1101.html を参照してください。

Ultra Light.NET アーキテクチャ

Ultra Light.NET ネームスペースの名前は、iAnywhere.Data.UltraLite (ADO.NET インタフェース) です。

次のリストは、iAnywhere.Data.UltraLite ADO.NET ネームスペースでよく使用される高レベル・クラスの一部を説明しています。

- **ULConnection** 各 ULConnection オブジェクトは、Ultra Light データベースとの接続を表します。ULConnection オブジェクトは 1 つまたは複数作成できます。
- **ULTable** 各 ULTable オブジェクトを使用して、単一テーブル内のデータにアクセスできます。
- **ULCommand オブジェクト** 各 ULCommand オブジェクトは、データベースに対して実行される SQL 文を格納します。
- **ULDataReader オブジェクト** 各 ULDataReader オブジェクトは、単一のクエリの結果セットを格納します。
- **ULSyncParms** ULSyncParms オブジェクトを使用して、Ultra Light データベースを Mobile Link サーバと同期させます。

ADO.NET インタフェースの詳細については、「Ultra Light .NET 2.0 API リファレンス」 55 ページを参照してください。

Ultra Light 用 SQL Anywhere Explorer

目次

SQL Anywhere Explorer の概要 6
Ultra Light プロジェクトでの SQL Anywhere Explorer の使用 7

SQL Anywhere Explorer の概要

SQL Anywhere Explorer は、Visual Studio から SQL Anywhere と Ultra Light に接続できるコンポーネントです。

SQL Anywhere データベースでの SQL Anywhere Explorer の使用については、「SQL Anywhere Explorer」『SQL Anywhere サーバ - プログラミング』を参照してください。

Ultra Light プロジェクトでの SQL Anywhere Explorer の使用

Visual Studio では、SQL Anywhere Explorer を使用して Ultra Light データベースへの接続を作成できます。データベースに接続すると、次の操作を実行できます。

- テーブルとテーブル・データを参照する。
- Ultra Light データベースへの接続を開くプログラムやデータの取り出しと操作を行うプログラムを設計する。
- データベース・オブジェクトを C# や Visual Basic のコードやフォームにドラッグ・アンド・ドロップすることで、選択されたオブジェクトを参照するコードを IDE で自動的に生成する。

[ツール] メニューから対応するコマンドを選択することで、Sybase Central と Interactive SQL を Visual Studio から開くこともできます。

インストールの注意

Ultra Light ソフトウェアをインストールした Windows コンピュータに Visual Studio がインストール済みの場合、インストール・プロセスは Visual Studio の存在を検出し、必要な統合手順を実行します。Visual Studio を Ultra Light の後にインストールする場合、または Visual Studio の新しいバージョンをインストールする場合、次の手順に従って、コマンド・プロンプトから手動で Ultra Light を Visual Studio に統合する必要があります。

- Visual Studio が実行されていないことを確認します。
- コマンド・プロンプトで、*install-dir¥UltraLite¥UltraLite.NET¥Assembly¥v2¥installULNet.exe* を実行します。

Visual Studio での Ultra Light データベース接続の使用

SQL Anywhere Explorer を使用して、**[データ接続]** ノードで Ultra Light データベース接続を表示します。テーブル内のデータを表示するには、データ接続を作成してください。

SQL Anywhere Explorer でデータベース・テーブルを表示し、個々のテーブルを展開してカラムを表示できます。SQL Anywhere Explorer ウィンドウで選択されたオブジェクトのプロパティは、Visual Studio の **[プロパティ]** ウィンドウ枠に表示されます。

♦ **Visual Studio で Ultra Light データベース接続を追加するには、次の手順に従います。**

1. **[表示]** - **[SQL Anywhere エクスプローラ]** を選択して、SQL Anywhere Explorer を開きます。
2. SQL Anywhere Explorer ウィンドウで、**[データ接続]** を右クリックし、**[接続の追加]** を選択します。
3. **[Ultra Light]** を選択し、**[OK]** をクリックします。

4. 適切な値を入力して、データベースに接続します。
5. **[OK]** をクリックします。

   データベースへの接続が確立され、接続が **[データ接続]** リストに追加されます。

♦ **Visual Studio から Ultra Light データベース接続を削除するには、次の手順に従います。**

1. **[表示]** - **[SQL Anywhere エクスプローラ]** を選択して、SQL Anywhere Explorer を開きます。
2. SQL Anywhere Explorer ウィンドウで、削除する Ultra Light データベース接続を右クリックし、**[削除]** を選択します。

   SQL Anywhere Explorer ウィンドウから接続が削除されます。

SQL Anywhere Explorer の設定

Visual Studio の **[オプション]** ウィンドウには、SQL Anywhere Explorer を設定するのに使用できる設定が含まれています。オプションには、一般的なオプションと Ultra Light でのみ使用する固有のオプションがあります。

♦ **SQL Anywhere Explorer オプションを使用するには、次の手順に従います。**

1. Visual Studio で、**[ツール]** - **[オプション]** を選択します。
2. **[オプション]** ウィンドウの左ウィンドウ枠で、**[SQL Anywhere]** を展開します。
3. **[一般]** をクリックして、必要に応じて SQL Anywhere Explorer の一般オプションを設定します。

   [クエリ結果を制限する] **[出力]** ウィンドウに表示するローの数を指定します。デフォルト値は 500 です。

   [システム・オブジェクトをサーバ・エクスプローラに表示する] システム・オブジェクトを Microsoft サーバ・エクスプローラに表示する場合は、このオプションをオンにします。これは SQL Anywhere Explorer のオプションではなく、サーバ・エクスプローラのオプションです。システム・オブジェクトには、"dbo" ユーザが所有するオブジェクトが含まれます。

   [オブジェクトのソート] Ultra Light 用 Explorer ウィンドウで、オブジェクト名またはオブジェクトの所有者名順にオブジェクトをソートするときに選択します。

   [テーブルまたはビューをデザイナにドロップするときに UI コードを生成する] Windows フォーム・デザイナにドラッグ・アンド・ドロップするテーブルまたはビューのコードを生成します。

   [データ・アダプタ用に INSERT、UPDATE、DELETE コマンドを生成する] C# または Visual Basic ドキュメントにテーブルまたはビューをドラッグ・アンド・ドロップするときに、データ・アダプタ用の INSERT、UPDATE、DELETE コマンドを生成します。

   [データ・アダプタ用にテーブル・マッピングを生成する] C# または Visual Basic ドキュメントにテーブルをドラッグ・アンド・ドロップするときに、データ・アダプタ用のテーブル・マッピングを生成します。

4. [Ultra Light] をクリックして、Ultra Light に固有のオプションを設定します。

   [テーブルをコードにドロップするときに生成する] アプリケーション・コードにドラッグ・アンド・ドロップするテーブルに特定タイプのコードを生成します。次のうちの 1 つを選択してください。

   - ULResultSet は、位置付け更新や削除の実行対象となる編集可能な結果を表します。
   - ULDataReader は、読み込み専用の結果セットを表します。
   - ULTable は、テーブル・データの格納、削除、更新、読み込みに使用するコードを表します。
   - ULDataAdapter は、テーブル・データをオフラインで使用するための方法です。

SQL Anywhere Explorer を使用した Ultra Light データベース・オブジェクトの追加

Visual Studio では、SQL Anywhere Explorer から Visual Studio デザイナに特定のデータベース・オブジェクトをドラッグ・アンド・ドロップすると、選択されたオブジェクトを参照する新しいコンポーネントが IDE によって自動的に作成されます。ドラッグ・アンド・ドロップ操作の設定を指定するには、Visual Studio の **[ツール] - [オプション]** を選択し、SQL Anywhere ノードを開きます。

次の表に、SQL Anywhere Explorer からドラッグできる Ultra Light オブジェクトと、Visual Studio フォーム・デザイナまたはコード・エディタにドロップしたときに作成されるコンポーネントを示します。

項目	結果
データ接続	データ接続を作成します。
テーブル	アダプタを作成します。

♦ SQL Anywhere Explorer を使用して、新しい Ultra Light データ・コンポーネントを作成するには、次の手順に従います。

1. データ・コンポーネントを追加するフォームまたはクラスを開きます。
2. SQL Anywhere Explorer で、使用する Ultra Light オブジェクトを選択します。
3. SQL Anywhere Explorer からフォーム・デザイナまたはコード・エディタにオブジェクトをドラッグします。

SQL Anywhere Explorer を使用した Ultra Light テーブルの操作

SQL Anywhere Explorer では、Visual Studio からデータベースの Ultra Light テーブルのプロパティとデータを表示できます。

♦ **Visual Studio でテーブルを表示するには、次の手順に従います。**

1. SQL Anywhere Explorer を使用して、Ultra Light データベースに接続します。
2. SQL Anywhere Explorer ウィンドウで Ultra Light データベースを展開し、[テーブル] を展開します。
3. テーブルを右クリックして、**[データの取得]** を選択します。

   選択したテーブルのデータが Visual Studio の **[出力]** ウィンドウに表示されます。

Ultra Light.NET 開発の概要

目次

Visual Studio .NET での SQL Anywhere ツールの使用 12
データベースへの接続 13
暗号化と難読化 15
SQL を使用したデータへのアクセスと操作 16
テーブル API によるデータ・アクセスと操作 20
トランザクションの管理 26
スキーマ情報へのアクセス 27
エラー処理 28
ユーザの認証 29
Ultra Light アプリケーションの同期 30

Visual Studio .NET での SQL Anywhere ツールの使用

一部の SQL Anywhere ツールを Visual Studio 2005 や Visual Studio 2008 に統合することで、SQL Anywhere データベースと Ultra Light データベースの両方に使用できる統合データベース開発環境を実現できます。

- SQL Anywhere Explorer を使用すると、テーブルやその中のデータを表示したり、データの操作や取得を行うためにデータベースへの接続を開くプログラムを設計したりできます。SQL Anywhere Explorer は、**[表示]** メニューから開くことができます。
- Sybase Central と Interactive SQL は、Visual Studio .NET を離れずに開くことができます。Sybase Central と Interactive SQL は、**[ツール]** メニューから開くことができます。

注意
Ultra Light 開発では、ビュー用のアダプタを追加したり、ストアド・プロシージャ用のコマンドを追加することはできません。これらの機能は SQL Anywhere データベースに対してのみ使用できます。

参照

- 「Visual Studio での Ultra Light データベース接続の使用」 7 ページ
- 「SQL Anywhere Explorer を使用した Ultra Light データベース・オブジェクトの追加」 9 ページ

データベースへの接続

データベースのデータを操作するには、Ultra Light アプリケーションをデータベースに接続する必要があります。この項では、Ultra Light データベースに接続する方法について説明します。

ULConnection オブジェクトの使用

ULConnection オブジェクトの次のプロパティは、アプリケーションのグローバルな動作を管理します。

- **コミット動作**　デフォルトでは、Ultra Light.NET アプリケーションは AutoCommit モードに設定されています。Insert 文、Update 文、Delete 文はすべて、すぐにデータベースにコミットされます。アプリケーションでのトランザクションの開始を定義するには、ULConnection.BeginTransaction を使用します。「トランザクションの管理」 26 ページを参照してください。
- **ユーザ認証**　接続許可の付与と取り消しを行うメソッドを使用すると、アプリケーションのユーザ ID とパスワードをデフォルト値である DBA と sql から別の値に変更できます。各 Ultra Light データベースには、4 つまでのユーザ ID を定義できます。「ユーザの認証」 29 ページを参照してください。
- **同期**　Connection オブジェクトから、同期を管理するオブジェクトのセットにアクセスできます。「Ultra Light アプリケーションの同期」 30 ページを参照してください。
- **テーブル**　Ultra Light テーブルには、Connection オブジェクトのメソッドを使用してアクセスします。「テーブル API によるデータ・アクセスと操作」 20 ページを参照してください。
- **コマンド**　動的 SQL 文の実行を処理し、結果セットをナビゲーションするために、オブジェクトのセットが提供されています。「SQL を使用したデータへのアクセスと操作」 16 ページを参照してください。

「ULConnection クラス」 139 ページを参照してください。

マルチスレッド・アプリケーション

ULConnection オブジェクトと、このオブジェクトから作成されるすべてのオブジェクトは、それぞれ単一のスレッドで使用してください。アプリケーションが Ultra Light データベースにアクセスするのに複数のスレッドを必要とする場合は、スレッドごとに個別の接続が必要です。たとえば、独立したスレッドから同期を実行するアプリケーションを設計する場合、同期用に別の接続を使用し、その接続を独立したスレッドから開く必要があります。

♦ Ultra Light データベースに接続するには、次の手順に従います。

1. ULConnection オブジェクトを宣言します。

   ほとんどのアプリケーションは、Ultra Light データベースへの単一接続を使用し、接続を開いたままにしておきます。複数の接続が必要になるのは、マルチスレッド・データ・アクセスの場合だけです。そのため、ULConnection オブジェクトは、アプリケーションに対してグローバルに宣言するのが最も効果的です。

   ```
   ULConnection conn;
   ```

2. 既存のデータベースへの接続を開きます。

   Ultra Light アプリケーションには、初期データベース・ファイルが配備されているか、データベース・ファイルを作成するコードが含まれている必要があります。初期データベース・ファイルは、Sybase Central、または Ultra Light に付属のコマンド・ライン・ユーティリティを使用して作成できます。

   接続パラメータは、接続文字列として指定するか、ULConnectionParms オブジェクトを使用して指定します。次の例では、ULConnectionParms オブジェクトを使用して、*mydata.udb* という名前の Ultra Light データベースに接続します。

   ```
   ULConnectionParms parms = new ULConnectionParms();
   parms.DatabaseOnDesktop = "mydata.udb";
   conn = new ULConnection( parms.ToString() );
   conn.Open();
   ```

C# のコード・サンプル
この章のサンプル・コードは Microsoft C# で記述されています。サポートされている他の開発ツールを使用している場合は、このマニュアルの記述を適宜読み替えてください。

暗号化と難読化

デフォルトでは、Ultra Light データベースは暗号化されていません。適切なデータベース作成パラメータを指定することで、データベースを作成するときに強力に暗号化したり簡単な難読化を適用したりできます。難読化とは、非常に弱い形態の、キーを使用しない暗号化のことで、(低レベル・ファイル検証ユーティリティが使用された場合など) データベース内のデータが偶発的に見られることを防ぐことだけを目的としています。

暗号化

強力な暗号化を使用してデータベースを作成するには、Sybase Central を使用してデータベースを作成するときに暗号化キーを指定します。データベースの作成に ULCreateDatabase を呼び出すか ulcreate ユーティリティを使用する場合は、該当するパラメータに暗号化キーを指定します。高い効果を得るために、文字、数字、特殊記号を組み合わせて暗号化キーを作成してください。暗号化キーは、長くするほど推測されにくくなります。

データベースが暗号化されると、暗号化キーは復元できなくなります。正しい暗号化キーを指定しないと、データベースにはまったくアクセスできません。暗号化キーは機密情報として扱い、適切に保管してください。

「Ultra Light DBKEY 接続パラメータ」『Ultra Light データベース管理とリファレンス』を参照してください。

既存の Ultra Light データベースの暗号化キーを変更するには、Connection.ChangeEncryptionKey メソッドを使用して新しい暗号化キーを適用します。

「ULConnection クラス」 139 ページと「ULConnectionParms クラス」 189 ページを参照してください。

データベースが暗号化された後は、データベースへの接続で正しい暗号化キーを指定する必要があります。指定しないと、接続は失敗します。

難読化

データベースを難読化するには、データベース作成パラメータとして **obfuscate=y** を指定します。データベースの暗号化と難読化のパラメータの詳細については、「Ultra Light で使用するデータベース作成パラメータの選択」『Ultra Light データベース管理とリファレンス』を参照してください。

SQL を使用したデータへのアクセスと操作

Ultra Light アプリケーションは、SQL 文またはテーブル API を使用してテーブル・データにアクセスできます。この項では、SQL 文を使用したデータ・アクセスについて説明します。

テーブル API の使用方法の詳細については、「テーブル API によるデータ・アクセスと操作」 20 ページを参照してください。

この項では、SQL を使用して次の操作を行う方法を説明します。

- ローの挿入、削除、更新
- クエリの実行と結果セットのローの取得
- 結果セットのローのスクロール

この項では、SQL 言語そのものについては説明しません。SQL の機能の詳細については、「SQL 文」『SQL Anywhere サーバ - SQL リファレンス』を参照してください。

データ操作：INSERT、UPDATE、DELETE

Ultra Light では、SQL データ操作言語の操作を実行できます。この操作は、ULCommand.ExecuteNonQuery メソッドを使用して実行します。

「ULCommand クラス」 91 ページを参照してください。

SQL 文のパラメータのプレースホルダは、? 文字を使用して指定します。INSERT、UPDATE、DELETE では必ず、コマンドのパラメータでの順序位置に従ってそれぞれの ? が参照されます。たとえば、最初の ? は 0、2 番目の ? は 1 のようになります。

♦ ローを挿入するには、次の手順に従います。

1. ULCommand を宣言します。

   ```
   ULCommand cmd;
   ```

2. SQL 文を ULCommand オブジェクトに割り当てます。

   ```
   cmd = conn.CreateCommand();
   cmd.Command = "INSERT INTO MyTable(MyColumn) values (?)";
   ```

3. 文の入力パラメータ値を割り当てます。

   次のコードは、文字列パラメータを示します。

   ```
   String newValue;
   // assign value
   cmd.Parameters.add("", newValue);
   ```

4. 文を実行します。

   戻り値は、文に影響されたローの数を示します。

   ```
   int rowsInserted = cmd.ExecuteNonQuery();
   ```


5. 明示的なトランザクションを使用している場合は、変更をコミットします。

```
myTransaction.Commit();
```

♦ **ローを更新するには、次の手順に従います。**

1. ULCommand を宣言します。

```
ULCommand cmd;
```

2. ULCommand オブジェクトに文を割り当てます。

```
cmd = conn.CreateCommand();
cmd.Command = "UPDATE MyTable SET MyColumn1 = ? WHERE MyColumn2 = ?";
```

3. 文の入力パラメータ値を割り当てます。

```
String newValue;
String oldValue;
// assign values
cmd.Parameters.add("", newValue);
cmd.Parameters.add("", oldValue);
```

4. 文を実行します。

```
int rowsUpdated = cmd.ExecuteNonQuery();
```

5. 明示的なトランザクションを使用している場合は、変更をコミットします。

```
myTransaction.Commit();
```

♦ **ローを削除するには、次の手順に従います。**

1. ULCommand を宣言します。

```
ULCommand cmd;
```

2. ULCommand オブジェクトに文を割り当てます。

```
cmd = conn.CreateCommand();
cmd.Command = "DELETE FROM MyTable WHERE MyColumn = ?";
```

3. 文の入力パラメータ値を割り当てます。

```
String deleteValue;
// assign value
cmd.Parameters.add("", deleteValue);
```

4. 文を実行します。

```
int rowsDeleted = cmd.ExecuteNonQuery();
```

5. 明示的なトランザクションを使用している場合は、変更をコミットします。

```
myTransaction.Commit();
```


データの取得: SELECT

SELECT 文を使用すると、データベースから情報を取り出すことができます。この項では、SELECT 文の実行方法と返される結果セットの処理方法について説明します。

♦ **SELECT 文を実行するには、次の手順に従います。**

1. クエリを保持する ULCommand オブジェクトを宣言します。

   ```
   ULCommand cmd;
   ```

2. このオブジェクトに文を割り当てます。

   ```
   cmd = conn.CreateCommand();
   cmd.Command = "SELECT MyColumn FROM MyTable";
   ```

3. 文を実行します。

   クエリの結果は、オブジェクトのいくつかの種類の 1 つとして返されます。この例では、ULDataReader オブジェクトが使用されています。次のコードでは、SELECT 文の結果に文字列が含まれています。これはコマンド・プロンプトへ出力されます。

   ```
   ULDataReader customerNames = prepStmt.ExecuteReader();
   int fc = customerNames.GetFieldCount();
   while( customerNames.MoveNext() ) {
     for ( int i = 0; i < fc; i++ ) {
       System.Console.Write(customerNames.GetString( i ) + " " );
     }
     System.Console.WriteLine();
   }
   ```

SQL 結果セットのナビゲーション

ULDataReader オブジェクトに関連したメソッドを使用して、結果セット内をナビゲーションできます。

結果セット・オブジェクトは、結果セットをナビゲーションする次のメソッドを提供します。

- **MoveAfterLast**　最後のローの後に移動します。
- **MoveBeforeFirst**　最初のローの前に移動します。
- **MoveFirst**　最初のローに移動します。
- **MoveLast**　最後のローに移動します。
- **MoveNext**　次のローに移動します。
- **MovePrevious**　前のローに移動します。
- **MoveRelative(offset)**　現在のローを基準にして、オフセットが指定した数だけローを移動します。オフセット値を正の値で指定すると、現在の結果セットのカーソル位置から前方に移動します。負の値で指定すると後方に移動します。オフセット値が 0 の場合、カーソルは移動しませんが、ロー・バッファが再配置されます。

結果セット・スキーマの説明

ULDataReader.GetSchemaTable メソッドと ULDataReader.Schema プロパティを使用すると、カラム名、カラムの総数、カラム・スケール、カラム・サイズ、カラムの SQL 型など、結果セットに関する情報を取得できます。

例

次のサンプル・コードは、ULDataReader.Schema プロパティと ResultSet.Schema プロパティを使用して、スキーマ情報をコマンド・プロンプトに表示する方法を示しています。

```
for ( int i = 0; i < MyResultSet.Schema.GetColumnCount(); i++ ) {
    System.Console.WriteLine( MyResultSet.Schema.GetColumnName(i)
                        + " "
                        + MyResultSet.Schema.GetColumnSQLType(i) );
}
```


テーブル API によるデータ・アクセスと操作

Ultra Light アプリケーションは、SQL 文またはテーブル API を使用してテーブル・データにアクセスできます。この項では、テーブル API を使用したデータ・アクセスについて説明します。

SQL の詳細については、「SQL を使用したデータへのアクセスと操作」 16 ページを参照してください。

この項では、テーブル API を使用して次の操作を行う方法について説明します。

- テーブルのローのスクロール
- 現在のローの値へのアクセス
- find メソッドと lookup メソッドを使用したテーブルのローの検索
- ローの挿入、削除、更新

テーブルのローのナビゲーション

Ultra Light.NET は、幅広いナビゲーション作業を行うために、テーブルをナビゲーションする複数のメソッドを提供します。

テーブル・オブジェクトは、テーブルをナビゲーションする次のメソッドを提供します。

- **MoveAfterLast**　最後のローの後に移動します。
- **MoveBeforeFirst**　最初のローの前に移動します。
- **MoveFirst**　最初のローに移動します。
- **MoveLast**　最後のローに移動します。
- **MoveNext**　次のローに移動します。
- **MovePrevious**　前のローに移動します。
- **MoveRelative(offset)**　現在のローを基準にして、オフセットが指定した数だけローを移動します。オフセット値を正の値で指定すると、現在のテーブルのカーソル位置から前方に移動します。負の値で指定すると後方に移動します。オフセット値が 0 の場合、カーソルは移動しませんが、ロー・バッファが再配置されます。

例

次のサンプル・コードは、MyTable テーブルを開き、各ローの MyColumn カラムの値を表示します。

```
ULTable t = conn.ExecuteTable( "MyTable" );
int colID = t.GetOrdinal( "MyColumn" );
while ( t.MoveNext() ){
   System.Console.WriteLine( t.GetString( colID ) );
}
```


テーブル・オブジェクトを開くと、テーブルのローがアプリケーションに公開されます。デフォルトでは、ローはプライマリ・キー値の順に並んでいますが、テーブルを開くときにインデックスを指定すると特定の順序でローにアクセスできます。

例

次のコードは、ix_col インデックスで順序付けられた MyTable テーブルの最初のローに移動します。

```
ULTable t = conn.ExecuteTable( "MyTable", "ix_col" );
t.MoveFirst();
```

「ULTable クラス」 548 ページと「ULTableSchema クラス」 571 ページを参照してください。

Ultra Light のモードの使用

Ultra Light モードは、バッファ内の値の使用目的を指定します。Ultra Light には、デフォルト・モードに加えて、次の 4 つの操作モードがあります。

- **挿入モード** Insert メソッドを呼び出すと、バッファ内のデータが新しいローとしてテーブルに追加されます。
- **更新モード** Update メソッドを呼び出すと、現在のローがバッファ内のデータに置き換えられます。
- **検索モード** find メソッドの 1 つが呼び出されたときに、値がバッファ内のデータに正確に一致するローの検索に使用されます。
- **ルックアップ・モード** いずれかの lookup メソッドが呼び出されたときに、バッファ内のデータと一致するか、それより大きい値のローを検索します。

現在のローの値へのアクセス

Table オブジェクトは、次のいずれかの位置に常に置かれています。

- テーブルの最初のローの前
- テーブルのいずれかのローの上
- テーブルの最後のローの後ろ

Table オブジェクトがローの上に置かれている場合は、そのデータ型に適したメソッド・セットを使用して、各カラムの値を取得したり、変更したりできます。

カラム値の取得

Table オブジェクトは、カラム値を取得するメソッド・セットを提供します。これらのメソッドは、カラム ID を引数として取ります。

例

次のコードは、lname カラムの値を取得します。このカラムの値は文字列です。

```
int lname = t.GetOrdinal( "lname" );
string lastname = t.GetString( lname );
```

次のコードは、cust_id カラムの値を取得します。このカラムの値は整数です。

```
int cust_id = t.GetOrdinal( "cust_id" );
int id = t.GetInt( cust_id );
```

カラム値の変更

値を取り出すメソッド以外に、値を設定するメソッドもあります。値を設定するメソッドは、カラム ID と値を引数として取ります。

例

たとえば、次のコードは、lname カラムの値を Kaminski に設定します。

```
t.SetString( lname, "Kaminski" );
```

これらのプロパティへの値の割り当てによって、データベース内のデータの値が変更されることはありません。現在の位置がテーブルの最初のローの前でも、テーブルの最後のローの後ろであっても、プロパティに値を割り当てることができます。ただし、これらの位置に現在のローが置かれている場合に、たとえば、変数へのプロパティの割り当てなどによって、データにアクセスしようとするとエラーになります。

```
// This code is incorrect
t.MoveBeforeFirst();
id = t.GetInt( cust_id );
```

値のキャスト

選択するメソッドは、割り当てるデータ型に一致させてください。データ型に互換性がある場合は、Ultra Light が自動的にデータベースのデータ型をキャストするため、getString メソッドを使用して整数値を文字列変数にフェッチしたりできます。「データ型の明示的な変換」 『Ultra Light データベース管理とリファレンス』を参照してください。

find と lookup を使用したローの検索

Ultra Light には、データを操作するための操作モードがいくつかあります。これらのモードのうちの 2 つ (検索モードとルックアップ・モード) は、検索に使用されます。Table オブジェクトには、テーブル内の特定のローを検索するために、これらのモードに対応するメソッドがあります。

> **注意**
> find メソッドや lookup メソッドを使用して検索されるカラムは、テーブルを開くのに使用されたインデックスにあることが必要です。

- **find メソッド**　Table オブジェクトを開いたときに指定したソート順に基づいて、指定された検索値と正確に一致する最初のローに移動します。検索値が見つからない場合は、最初のローの前、または最後のローの後ろに位置設定されます。
- **lookup メソッド**　Table オブジェクトを開いたときに指定したソート順に基づいて、指定された検索値と一致するか、それより大きい値の最初のローに移動します。

♦ **ローを検索するには、次の手順に従います。**

1. 検索モードまたはルックアップ・モードを開始します。

   モードを開始するには、テーブル・オブジェクトのメソッドを呼び出します。たとえば、次のコードは検索モードを開始します。

   ```
   t.FindBegin();
   ```

2. 検索値を設定します。

   検索値は、現在のローの値を設定することで設定します。これらの値の設定は、データベースではなく、現在のローを保持しているバッファにのみ影響します。たとえば、次のコードは、バッファの値を Kaminski に設定します。

   ```
   int lname = t.GetOrdinal( "lname" );
   t.SetString( lname, "Kaminski" );
   ```

3. ローを検索します。

   適切なメソッドを使用して検索を実行します。たとえば、次の指示は、現在のインデックスで指定された値と正確に一致する最初のローを検索します。

   マルチカラム・インデックスの場合、最初のカラムの値が常に使用され、他のカラムは省略できます。

   ```
   tCustomer.FindFirst();
   ```

4. ローの次のインスタンスを検索します。

   適切なメソッドを使用して検索を実行します。検索操作の場合は、FindNext でインデックス内のパラメータの次のインスタンスを検索します。ルックアップ操作では、MoveNext で次のインスタンスを検索します。

「ULTable クラス」 548 ページを参照してください。

ローの更新

ローの更新手順を次に示します。

♦ **ローを更新するには、次の手順に従います。**

1. 更新するローに移動します。

   ローに移動するには、テーブルをスクロールするか、find メソッドまたは lookup メソッドを使用してテーブルを検索します。

2. 更新モードを開始します。

   たとえば、次の指示は、テーブル t 上で更新モードを開始します。

   ```
   t.BeginUpdate();
   ```

3. 更新するローの新しい値を設定します。

   たとえば、次の指示は、バッファ内の id カラムを 3 に設定します。

   ```
   t.SetInt( id , 3);
   ```

4. Update を実行します。

   ```
   t.Update();
   ```

更新操作が終了すると、更新したローが現在のローになります。Table オブジェクトを開いたときに指定したインデックスのカラム値を変更した場合は、現在のローの定義が解除されます。

デフォルトでは、Ultra Light.NET は AutoCommit モードで動作するため、更新は永続的な記憶領域のローに即時適用されます。AutoCommit モードを無効にした場合は、コミット操作を実行するまで、更新は適用されません。「トランザクションの管理」 26 ページを参照してください。

警告
ローのプライマリ・キー値は更新できません。代わりに、ローを削除して新しいローを追加してください。

ローの挿入

ローの挿入手順は、ローの更新手順とほぼ同じです。ただし、挿入操作の場合は、テーブル内のローをあらかじめ指定する必要はありません。テーブルへのローの挿入順序に意味はありません。

例

次のコードでは、新しいローが挿入されます。

```
t.InsertBegin();
t.SetInt( id, 3 );
t.SetString( lname, "Carlo" );
t.Insert();
```

カラムの値を設定しない場合、そのカラムにデフォルト値があるときはデフォルト値が使用されます。カラムにデフォルトがない場合は、次のエントリが使用されます。

- NULL 入力可のカラムの場合は NULL
- NULL 入力不可の数値カラムの場合は 0
- NULL 入力不可の文字カラムの場合は空の文字列
- 明示的に値を NULL に設定するには、setDBNull メソッドを使用します。

更新操作の場合は、コミットを実行したときに、永続的な記憶領域のデータベースに挿入が適用されます。AutoCommit モードでは、挿入メソッドの一部としてコミットが実行されます。

ローの削除

ローの削除手順は、ローの挿入や更新よりも簡単です。挿入モードや更新モードに対応する削除モードはありません。

次のプロシージャは、ローを削除します。

♦ **ローを削除するには、次の手順に従います。**

1. 削除するローに移動します。
2. Table.Delete メソッドを実行します。

```
t.Delete();
```

トランザクションの管理

Ultra Light のトランザクション処理は、データベース内のデータの整合性を保証します。トランザクションは、作業の論理単位です。トランザクション全体が実行されるか、トランザクション内の文がどれも実行されないかのいずれかです。

デフォルトでは、Ultra Light.NET は AutoCommit モードで動作するため、挿入、更新、削除はそれぞれ独立したトランザクションとして実行されます。操作が完了すると、データベースに変更が加えられます。

複数文のトランザクションを使用するには、ULConnection.BeginTransaction を呼び出して ULTransaction クラスのオブジェクトを作成する必要があります。たとえば、2 つの口座間で資金を移動するアプリケーションでは、振込元口座からの引き落としと振込先口座への振り込みとを合わせて 1 つの操作として完了する必要があります。完了できない場合は、両方とも完了しないでください。

接続に有効なトランザクションがある場合は、ULTransaction.Commit 文を実行して、トランザクションを完了し、データベースへの変更をコミットする必要があります。一連の更新を中止する場合は、ULTransaction.Rollback 文を実行して、トランザクションのすべての操作を取り消してロールバックする必要があります。トランザクションがコミットまたはロールバックされると、次に ULConnection.BeginTransaction を呼び出すまで、接続は AutoCommit モードに戻ります。

たとえば、次のコード・フラグメントは、(デフォルトのオートコミット動作を回避しながら) 複数の操作を伴うトランザクションを設定する方法を示しています。

```
// Assuming an already open connection named  conn
ULTransaction txn = conn.BeginTransaction(IsolationLevel.ReadUncommitted);
// Carry out transaction operations here
txn.Commit();
```

Ultra Light の独立性レベル
Ultra Light は、IsolationLevel 列挙体の IsolationLevel.ReadUncommitted メンバのみをサポートします。

一部の SQL 文 (特にデータベースの構造を変更する文) は、保留中のトランザクションをすべてコミットします。処理中のトランザクションを自動的にコミットする SQL 文には、CREATE TABLE や ALTER TABLE などがあります。

「ULConnection クラス」 139 ページと「ULTransaction クラス」 586 ページを参照してください。

スキーマ情報へのアクセス

テーブル API のオブジェクトは、テーブル、カラム、インデックス、同期の各パブリケーションを表します。各オブジェクトには、そのオブジェクトの構造情報へアクセスするための Schema プロパティがあります。

API によるスキーマの変更はできません。スキーマに関する情報の取得のみが可能です。

次のスキーマ・オブジェクトと情報にアクセスできます。

- **DatabaseSchema**　データベース内のテーブルの数と名前、日付と時刻のフォーマットなどのグローバル・プロパティを公開します。

  ULDatabaseSchema オブジェクトを取得するには、ULConnection.Schema にアクセスします。

  「ULConnection クラス」 139 ページを参照してください。

- **TableSchema**　このテーブル内のカラムとインデックスの数と名前。

  ULTableSchema オブジェクトを取得するには、ULTable.Schema にアクセスします。

- **IndexSchema**　インデックス内のカラムに関する情報。インデックスには直接に対応するデータがないため、個別の Index クラスはなく、ULIndexSchema クラスのみが存在します。

  ULIndexSchema オブジェクトを取得するには、ULTableSchema.GetIndex、ULTableSchema.GetOptimalIndex、ULTableSchema.GetPrimaryKey のいずれかのメソッドを呼び出します。

- **PublicationSchema**　パブリケーションに含まれるテーブルとカラムのリスト。パブリケーションには PublicationSchema オブジェクトは含まれますが、Publication オブジェクトは含まれません。

  ULPublicationSchema オブジェクトを取得するには、ULDatabaseSchema.GetPublicationSchema メソッドを呼び出します。

「ULTableSchema クラス」 571 ページを参照してください。

エラー処理

.NET の標準エラー処理機能を使用して、エラーを処理できます。ほとんどの Ultra Light メソッドは、ULException エラーをスローします。ULException.NativeError を使用して、エラーに割り当てられた ULSQLCode 値を取得できます。ULException には Message プロパティがあり、エラーの説明文の取得に使用できます。ULSQLCode エラーは、エラー・タイプを示す負の番号です。

エラー・コードのリストについては、エラー・メッセージを参照してください。

同期後は、接続の SyncResult プロパティを使用して詳細なエラー情報を取得できます。たとえば、次の例は、同期時に発生したエラーをレポートする方法を示しています。

```
public void Sync()
 {
      try
      {
        _conn.Synchronize( this );
        _inSync = false;
      }
      catch( ULException uEx ){
        if( uEx.NativeError == ULSQLCode.SQLE_COMMUNICATIONS_ERROR )
        {
          MessageBox.Show(
              "StreamErrorCode = " +
              _conn.SyncResult.StreamErrorCode.ToString() +
              "¥r¥n"
            + "StreamErrorContext = " +
              _conn.SyncResult.StreamErrorContext + "¥r¥n"
            + "StreamErrorID = " +
              _conn.SyncResult.StreamErrorID + "¥r¥n"
            + "StreamErrorSystem = " +
              _conn.SyncResult.StreamErrorSystem + "¥r¥n"
            );
        }
        else
        {
          MessageBox.Show(uEx.Message);
        }
      }
      catch(System.Exception ex )
      {
        MessageBox.Show(ex.Message);
      }
 }
```

参照

- 「ULSyncProgressListener インタフェース」 538 ページ
- 「ULSyncResult クラス」 542 ページ

ユーザの認証

新しいユーザは既存の接続から追加する必要があります。Ultra Light のすべてのデータベースは、デフォルトのユーザ ID DBA とパスワード sql を使用して作成されるため、まずこのユーザとして接続します。

ユーザ ID を直接変更することはできません。新しいユーザ ID を追加して、既存のユーザ ID を削除します。Ultra Light では、Ultra Light データベースごとに ID を 4 つまでサポートします。

「ユーザの認証」 29 ページを参照してください。

♦ ユーザを追加する、または既存のユーザのパスワードを変更するには、次の手順に従います。

1. 既存のユーザのユーザ ID とパスワードを使用してデータベースに接続します。
2. ULConnection.GrantConnectTo メソッドを使用して、指定のパスワードによるデータベースへのアクセスをユーザに付与します。

   新規ユーザを追加する場合も、既存のユーザのパスワードを変更する場合も、この手順は同じです。

   「ULConnection クラス」 139 ページを参照してください。

♦ 既存のユーザを削除するには、次の手順に従います。

1. 既存のユーザのユーザ ID とパスワードを使用してデータベースに接続します。
2. Connection.RevokeConnectFrom メソッドを使用して既存のユーザを削除します。

Ultra Light アプリケーションの同期

Ultra Light データベースを中央の統合データベースと同期します。同期には、SQL Anywhere に付属の Mobile Link 同期ソフトウェアが必要です。

この項では、同期の概要について簡単に説明し、Ultra Light.Net コンポーネントを使用するうえで重要となるいくつかの機能について説明します。

同期の詳細については、「Ultra Light クライアント」『Ultra Light データベース管理とリファレンス』を参照してください。

また、CustDB サンプル・アプリケーションには、同期の実例もあります。詳細については、SQL Anywhere 11 インストール環境の *Samples¥UltraLite.NET¥CustDB* サブディレクトリを参照してください。

Ultra Light.NET は、TCP/IP、HTTP、HTTPS、TLS (トランスポート・レイヤ・セキュリティ) の各同期をサポートしています。同期は、Ultra Light アプリケーションによって開始されます。いずれの場合でも、SyncParms オブジェクトのプロパティを使用して同期を制御します。

別途ライセンスが必要な必須コンポーネント
ECC 暗号化と FIPS 認定の暗号化には、別途ライセンスが必要です。強力な暗号化テクノロジはすべて、輸出規制対象品目です。

「別途ライセンスが必要なコンポーネント」『SQL Anywhere 11 - 紹介』を参照してください。

C# アプリケーションでの同期の開始

次のコードは、C# で記述されているアプリケーションで同期を開始する方法を示します。

```
private void Sync()
{
    // Sync
    try
    {
        // setup to synchronize a publication named "high_priority"
        conn.SyncParms.Publications = "high_priority";

        // Set the synchronization parameters
        conn.SyncParms.Version     = "Version1";
        conn.SyncParms.StreamParms = "";
        conn.SyncParms.Stream      = ULStreamType.TCPIP;
        conn.SyncParms.UserName    = "51";
        conn.Synchronize();
    }
    catch (System.Exception t)
        {
          MessageBox.Show("Exception: " + t.Message);
        }
}
```

アプリケーションへの ActiveSync 同期の追加

この項では、ActiveSync を Ultra Light.NET アプリケーションに追加する方法について説明します。また、エンド・ユーザのコンピュータ上の ActiveSync で使用するアプリケーションの登録方法についても説明します。

ActiveSync 同期を開始できるのは、ActiveSync 自体だけです。デバイスがクレードルにある場合や、[ActiveSync] ウィンドウで同期コマンドが選択された場合に、ActiveSync は同期を開始します。

ActiveSync が同期を開始すると、Ultra Light アプリケーションがまだ実行されていない場合は ActiveSync 用 Mobile Link プロバイダが Ultra Light アプリケーションを起動し、メッセージを送信します。Mobile Link プロバイダからのメッセージを受信および処理するには、ULActiveSyncListener オブジェクトをアプリケーションに実装します。アプリケーションは、SetActiveSyncListener メソッドを使用してリスナ・オブジェクトを指定する必要があります。ここで、**MyAppClassName** は、アプリケーションのユニークな Windows クラス名です。

```
dbMgr.SetActiveSyncListener( "MyAppClassName", listener );
```

詳細とサンプル・コードについては、「ULActiveSyncListener インタフェース」 57 ページを参照してください。

Ultra Light は、ActiveSync メッセージを受信すると、指定されたリスナの ActiveSyncInvoked メソッドを別のスレッド上で呼び出します。マルチスレッドの問題を回避するには、ActiveSyncInvoked メソッドでユーザ・インタフェースにイベントを通知してください。

マルチスレッド・アプリケーションを使用する場合は、スレッドごとに別々の接続を使用し、C# の **lock** キーワードまたは Visual Basic .NET の **SyncLock** キーワードを使用して、アプリケーションの共有オブジェクトにアクセスします。ActiveSyncInvoked メソッドは、使用する接続の SyncParms.Stream に ULStreamType.ACTIVE_SYNC を指定してから ULConnection.Synchronize を呼び出します。

アプリケーションを登録するときは、次のパラメータを設定します。

- **クラス名**　Connection.SetActiveSyncListener メソッドでアプリケーションが使用したクラス名と同じクラス名

チュートリアル：Ultra Light.NET アプリケーションのビルド

目次

Ultra Light.NET 開発のチュートリアル 34
レッスン 1：Visual Studio プロジェクトの作成 35
レッスン 2：Ultra Light データベースの作成 39
レッスン 3 ：データベースへの接続 41
レッスン 4：データの挿入、更新、削除 43
レッスン 5：アプリケーションのビルドと配置 48
C# チュートリアルのコード・リスト 50
Visual Basic チュートリアルのコード・リスト 53

Ultra Light.NET 開発のチュートリアル

このチュートリアルでは、Microsoft Visual Studio で Ultra Light アプリケーションをビルドする手順について説明します。また、このチュートリアルでは iAnywhere.Data.UltraLite ネームスペースに用意されている ADO.NET インタフェースを使用します。

このチュートリアルには、Visual Basic アプリケーションと Visual C# アプリケーションのコードが含まれています。

前提知識と経験

このチュートリアルは、次のことを前提にしています。

- C# プログラミング言語または Visual Basic プログラミング言語に精通している。
- Microsoft Visual Studio がコンピュータにインストール済みである。また、Visual Studio に精通している。このチュートリアルは Visual Studio 2008 を使用してテストされており、Visual Studio のその他のバージョンとは多少異なる Visual Studio アクションやプロシージャに基づいている場合があります。
- Sybase Central の Ultra Light プラグインを使用して Ultra Light データベースを作成する方法を知っている。

  「データベース作成ウィザードを使用したデータベースの作成」『Ultra Light データベース管理とリファレンス』を参照してください。

目的

このチュートリアルの目的は、Visual Studio 環境での Ultra Light アプリケーションの開発プロセスについて、知識と経験を得ることです。

インストールの注意

Ultra Light ソフトウェアをインストールした Windows コンピュータに Visual Studio がインストール済みの場合、Ultra Light のインストール・プロセスは Visual Studio の存在を検出し、必要な統合手順を実行します。Visual Studio を Ultra Light の後にインストールする場合、または Visual Studio の新しいバージョンをインストールする場合、次の手順に従って、コマンド・プロンプトから手動で Ultra Light を Visual Studio に統合する必要があります。

- Visual Studio が実行されていないことを確認します。
- Visual Studio 2005 以降の場合は、*install-dir¥UltraLite¥UltraLite.NET¥Assembly¥v2¥* フォルダから *installULNet.exe* を実行します。

レッスン 1：Visual Studio プロジェクトの作成

次の手順では、新しい Visual Studio アプリケーションを作成および設定します。プログラミング言語として Visual Basic と C# のどちらも使用できます。

このチュートリアルでは、C# アプリケーションを設計している場合はファイルがディレクトリ *C:¥tutorial¥uldotnet¥CSApp* にあり、Visual Basic アプリケーションを設計している場合はファイルがディレクトリ *C:¥tutorial¥uldotnet¥VBApp* にあることを前提とします。別の名前のディレクトリを使用する場合は、チュートリアルを通じてそのディレクトリを使用してください。

♦ Visual Studio プロジェクトを作成するには、次の手順に従います。

1. Visual Studio プロジェクトを作成します。
   - Visual Studio で、**[ファイル] - [新規] - [プロジェクト]** を選択します。
   - **[新しいプロジェクト]** ウィンドウが表示されます。左ウィンドウ枠で、**[Visual Basic]** フォルダまたは **[Visual C#]** フォルダを選択します。プロジェクト・タイプとして **[スマート デバイス]** を選択します。

     右ウィンドウ枠で、**[スマート デバイス プロジェクト]** を選択し、プログラミング言語として Visual Basic と C# のどちらを使用するかに応じて、プロジェクトに **VBApp** または **CSApp** という名前を付けます。
   - *C:¥tutorial¥uldotnet* の場所を入力し、**[OK]** をクリックします。
   - ターゲット・プラットフォームとして **[Windows Mobile 5.0 Pocket PC SDK]** を選択します。**[OK]** をクリックします。
2. プロジェクトに参照を追加します。
   - iAnywhere.Data.UltraLite アセンブリと、関連するリソースをプロジェクトに追加します。
     a. **[プロジェクト] - [参照の追加]** を選択します。
     b. 利用可能な参照のリストから **[iAnywhere.Data.UltraLite (CE)]** を選択します。**[選択]** をクリックして選択済みコンポーネントのリストに追加します。

        この参照がリストに表示されない場合は、**[参照]** をクリックし、SQL Anywhere インストール環境の *UltraLite¥UltraLite.NET¥ce¥Assembly¥v2¥* サブディレクトリで探します。*iAnywhere.Data.UltraLite.dll* を選択し、**[OK]** をクリックします。
     c. 利用可能な参照のリストから **[iAnywhere.Data.UltraLite (CE) JA]** を選択します。**[選択]** をクリックして選択済みコンポーネントのリストに追加します。

        この参照がリストに表示されない場合は、**[参照]** をクリックし、SQL Anywhere インストール環境の *UltraLite¥UltraLite.NET¥ce¥xx* サブディレクトリで探します。この場合、xx は言語を示す 2 文字の略語 (英語の場合は **en** など) です。*iAnywhere.Data.UltraLite.resources.dll* を選択し、**[開く]** をクリックします。
     d. **[OK]** をクリックし、アセンブリとリソースをプロジェクトに追加します。
   - Ultra Light コンポーネントをプロジェクトにリンクします。

     ここで、リンクをコンポーネントに追加したことと、コンポーネントを開いていないことを確認します。

a. **[プロジェクト] - [既存項目の追加]** を選択し、SQL Anywhereインストール環境の *UltraLite ¥UltraLite.NET¥ce* サブディレクトリを参照します。

b. **[ファイルの種類]** リストで **[実行ファイル]** を選択します。

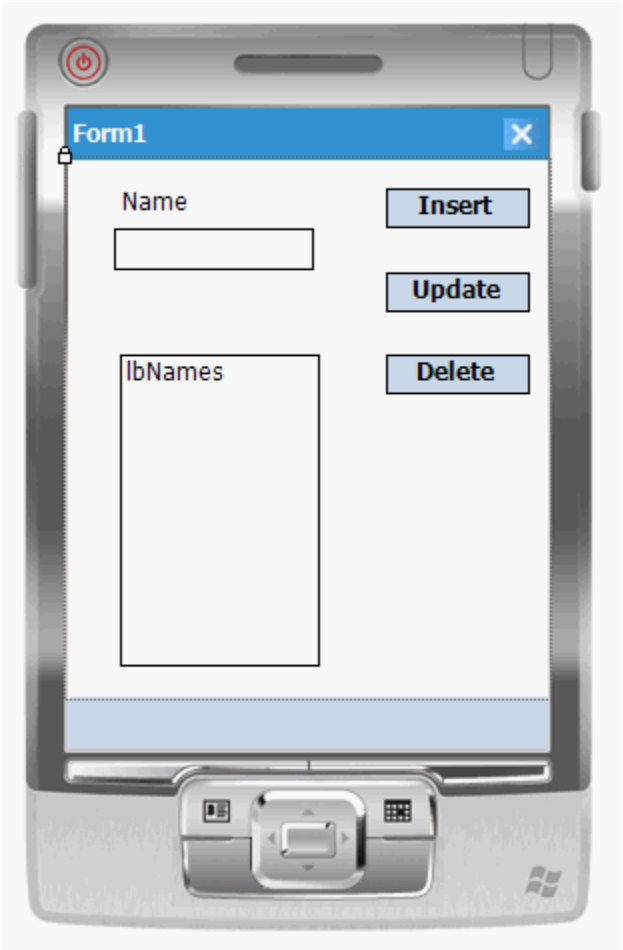
c. 使用している Windows Mobile デバイスのプロセッサに対応するフォルダを開きます。Visual Studio 2005 以降の場合は、*arm.50* フォルダを開きます。*ulnet11.dll* を選択し、**[追加]** ボタンの矢印をクリックして、**[リンクとして追加]** を選択します。

3. アプリケーションのフォームを作成します。

Visual Studio のツールボックス・パネルが表示されていない場合は、メイン・メニューから **[表示] - [ツールボックス]** を選択します。オブジェクトを選択して目的のロケーションのフォームにドラッグすることで、次のビジュアル・コンポーネントをフォームに追加します。

型	設計 - 名前	外観 - テキスト
Button	btnInsert	Insert
Button	btnUpdate	Update
Button	btnDelete	Delete
TextBox	txtName	(テキストなし)
ListBox	lbNames	(テキストなし)
Label	laName	Name

フォームは、次の図のようになります。



4. ソリューションをビルドおよび配置します。

   ソリューションをビルドおよび配置し、Visual Studio プロジェクトが正しく設定されたことを確認します。

   a. **[ビルド] - [ソリューションのビルド]** を選択します。プロジェクトが正常にビルドされたことを確認します。Visual Basic アプリケーションをビルドしている場合、次の警告は無視してかまいません。

      Referenced assembly 'iAnywhere.Data.UltraLite.resources' is a localized satellite assembly

   b. **[デバッグ] - [デバッグの開始]** を選択します。

      これによって、アプリケーションがデバイスまたはエミュレータに配置され、起動されます。アプリケーションは、エミュレータまたはプロジェクト名に応じたデバイスのロケーション (*¥Program Files¥VBApp* または *¥Program Files¥CSApp*) に配置されます。

      配置には時間がかかる場合があります。

   c. アプリケーションがエミュレータまたはターゲット・デバイスに配置され、設計したフォーム (**Form1**) が正しく表示されることを確認します。

d. エミュレータまたはターゲット・デバイスのアプリケーションを終了します。

レッスン 2：Ultra Light データベースの作成

(デスクトップ PC で実行する) 次の手順では、Sybase Central を使用して Ultra Light データベースを作成します。

「データベース作成ウィザードを使用したデータベースの作成」『Ultra Light データベース管理とリファレンス』を参照してください。

♦ **データベースを作成するには、次の手順に従います。**

1. **[スタート] - [プログラム] - [SQL Anywhere 11] - [Sybase Central]** を選択します。
2. Sybase Central の Ultra Light プラグインを使用して、アプリケーションと同じディレクトリにデータベースを作成します。通常、Sybase Central によって設定されるデフォルトのデータベース特性のままで問題ありません。特性は次のとおりです。
   - **データベース・ファイル名**　*VBApp.udb* または *CSApp.udb* (アプリケーション・タイプによって異なります)。
   - **照合順**　デフォルトの照合を使用します。
   - **文字列の比較で大文字と小文字を区別する**　このオプションはオフにしてください。
   - **テーブル名**　**Names** と入力します。
   - **カラム**　次の属性を持つカラムを **Names** テーブルに作成します。

カラム名	データ型 (サイズ)	NULL	ユニーク	デフォルト値
ID	integer	いいえ	はい (プライマリ・キー)	グローバル・オートインクリメント
Name	varchar(30)	いいえ	いいえ	なし

   - **プライマリ・キー**　ID カラムをプライマリ・キーとして指定します。
3. Sybase Central を終了し、データベース・ファイルが指定のディレクトリに作成されていることを確認します。
4. 初期化済みの (空の) データベース・ファイルを Visual Studio プロジェクトにリンクし、データベース・ファイルがアプリケーション・コードとともにデバイスに配備されるようにします。
   - **[Visual Studio] - [プロジェクト] - [既存項目の追加]** を選択します。
   - **[オブジェクトの種類]** が **[すべてのファイル]** に設定されていることを確認します。データベース・ファイルを作成したディレクトリを参照し、アプリケーション・タイプに応じてファイル *VBApp.udb* または *CSApp.udb* を選択します。
   - **[追加]** ボタンの矢印をクリックし、**[リンクとして追加]** をクリックします。
   - Solution Explorer フレームで、プロジェクトに追加したデータベース・ファイル名を右クリックし、**[プロパティ]** を選択します。

プロパティのパネルで、**[ビルド アクション]** プロパティを **[コンテンツ]** に設定し、**[出力ディレクトリにコピー]** プロパティを **[常にコピーする]** に設定します。

レッスン 3 ：データベースへの接続

次の手順では、Ultra Light データベースへの接続を確立する Ultra Light.NET アプリケーションにコントロールを追加します。

♦ **アプリケーションに Ultra Light 接続を追加するには、次の手順に従います。**

1. フォームをダブルクリックして、ソース・ファイル (*Form1.cs* または *Form1.vb*) を開きます。
2. コードを追加して、iAnywhere.Data.UltraLite ネームスペースをインポートします。

   ファイルの先頭行に次の文を追加します。

   ```
   //Visual C#
   using iAnywhere.Data.UltraLite;

   'Visual Basic
   Imports iAnywhere.Data.UltraLite
   ```
3. グローバル変数をフォーム宣言に追加します。

   Visual C# の場合は、フォーム・コンポーネントを記述したコードと、最初のメソッド宣言までの間に、次のコードを追加します。

   ```
   //Visual C#
   private ULConnection Conn;
   private int[] ids;
   ```

   Visual Basic の場合は、Form1 クラスの先頭に次のコードを追加します。

   ```
   'Visual Basic
   Dim Conn As ULConnection
   Dim ids() As Integer
   ```

   これらの変数は次のように使用します。

   - **ULConnection**　Connection オブジェクトは、データベースへの接続時に実行されるすべてのアクションのルート・オブジェクトです。
   - **ids**　ids 配列を使用して、クエリの実行後に返される ID カラム値を格納します。

     ListBox コントロール自体によって順序番号にアクセスできますが、ローが削除されると、これらの番号は ID カラムの値とは異なります。このため、ID カラム値を個別に格納してください。
4. フォームの空白領域をダブル・クリックして、Form1_Load メソッドを作成します。

   このメソッドでは次のタスクを実行します。

   - ulConnectionParms1 コントロールで設定されている接続パラメータを使用して、データベースへの接続を開きます。
   - RefreshListBox メソッドを呼び出します (このチュートリアルの後半で定義)。
   - エラーが発生したら、エラー・メッセージを印刷 (表示) します。SQL Anywhere エラーの場合は、エラー・コードも出力されます。エラー・メッセージを参照してください。

   C# の場合は、次のコードを Form1_Load メソッドに追加します。

```
//Visual C#
try {
    String ConnString = "dbf=¥¥Program Files¥¥CSApp¥¥CSApp.udb";
    Conn = new ULConnection( ConnString );
    Conn.Open();
    Conn.DatabaseID = 1;
    RefreshListBox();
}
catch ( System.Exception t ) {
    MessageBox.Show( "Exception: " + t.Message);
}
```

Visual Basic の場合は、次のコードを Form1_Load メソッドに追加します。

```
'Visual Basic
Try
    Dim ConnString as String = "dbf=¥Program Files¥VBApp¥VBApp.udb"
    Conn = New ULConnection( ConnString )
    Conn.Open()
    Conn.DatabaseID = 1
    RefreshListBox()
Catch
    MsgBox("Exception: " + err.Description)
End Try
```

5. プロジェクトをビルドします。

   [ビルド] - [ソリューションのビルド] を選択します。この段階で、1 つのエラーがレポートされます。たとえば C# の場合は、RefreshListBox が宣言されていないことを示す「error CS0103: The name 'RefreshListBox' does not exist in the class or namespace 'CSApp.Form1'」というエラーがレポートされます。次のレッスンではこの関数を追加します。

   他のエラーが表示された場合は、先へ進む前にエラーを修正します。C# における大文字と小文字の違いなどの一般的なエラーをチェックします。たとえば、**UltraLite** および **ULConnection** は大文字と小文字が正確に一致している必要があります。Visual Basic では、レッスン 3 で説明した **Imports iAnywhere.Data.Ultralite** 文を含めることが重要です。

レッスン 4：データの挿入、更新、削除

このレッスンでは、コードをアプリケーションに追加し、データベースのデータを変更します。次の手順では、動的 SQL を使用します。テーブル API を使用して同じテクニックを実行することもできます。

「テーブル API によるデータ・アクセスと操作」 20 ページを参照してください。

次の手順では、リストボックスを管理するサポート・メソッドを作成します。このメソッドは、残りの手順で作成されるデータ操作メソッドに必要です。

♦ **リストボックスを管理するコードを追加するには、次の手順に従います。**

1. フォームを右クリックし、**[コードの表示]** を選択します。
2. Form1 クラスのメソッドを追加し、リストボックスを更新してデータを移植します。このメソッドでは、次のタスクを実行します。
   - リストボックスをクリアします。
   - ULCommand オブジェクトをインスタンス化し、データベース内の Names テーブルのデータを返す SELECT クエリに割り当てます。
   - クエリを実行し、ULDataReader として結果セットを返します。
   - 結果セットの行数と同じ長さの整数配列をインスタンス化します。
   - ULDataReader で返される名前をリストボックスに移植し、ULDataReader で返される ID を整数配列に移植します。
   - ULDataReader を閉じます。
   - エラーが発生した場合は、エラー・メッセージが出力されます。SQL エラーの場合は、エラー・コードも出力されます。

     エラー・メッセージを参照してください。

   C# の場合は、Form1 クラスのメソッドとして次のコードをアプリケーションに追加します。

```
//Visual C#
private void RefreshListBox(){
    try{
        long NumRows;
        int I = 0;
        lbNames.Items.Clear();
        using( ULCommand cmd = Conn.CreateCommand() ){
            cmd.CommandText = "SELECT ID, Name FROM Names";
            using( ULDataReader dr = cmd.ExecuteReader()){
                dr.MoveBeforeFirst();
                NumRows = dr.RowCount;
                ids = new int[ NumRows ];
                while (dr.MoveNext())
                {
                    lbNames.Items.Add(
                    dr.GetString(1));
                    ids[ I ] = dr.GetInt32(0);
                    I++;
                }
            }
```

```
            txtName.Text = " ";
        }
    }
    catch( Exception err ){
        MessageBox.Show(
        "Exception in RefreshListBox: " + err.Message );
    }
}
```

Visual Basic の場合は、Form1 クラスのメソッドとして次のコードをアプリケーションに追加します。

```
'Visual Basic
Private Sub RefreshListBox()
    Try
        Dim cmd As ULCommand = Conn.CreateCommand()
        Dim I As Integer = 0
        lbNames.Items.Clear()
        cmd.CommandText = "SELECT ID, Name FROM Names"
        Dim dr As ULDataReader = cmd.ExecuteReader()
        ReDim ids(dr.RowCount)
        While (dr.MoveNext)
            lbNames.Items.Add(dr.GetString(1))
            ids(I) = dr.GetInt32(0)
            I = I + 1
        End While
        dr.Close()
        txtName.Text = " "
    Catch ex As Exception
        MsgBox(ex.ToString)
    End Try
End Sub
```

3. プロジェクトをビルドします。

   プロジェクトのビルドによってエラーが発生しないようにします。

♦ INSERT、UPDATE、DELETE を実装するには、次の手順に従います。

1. [フォーム デザイン] タブで **[Insert]** をダブルクリックして、btnInsert_Click メソッドを作成します。このメソッドでは、次のタスクを実行します。

   - ULCommand オブジェクトをインスタンス化し、テキスト・ボックス内の値をデータベースに挿入する INSERT 文に割り当てます。
   - 文を実行します。
   - ULCommand オブジェクトを破棄します。
   - リストボックスをリフレッシュします。
   - エラーが発生した場合は、エラー・メッセージが出力されます。SQL エラーの場合は、エラー・コードも出力されます。

     エラー・メッセージを参照してください。

   C# の場合は、次のコードを btnInsert_Click メソッドに追加します。

```
//Visual C#
try {
    long RowsInserted;
    using( ULCommand cmd = Conn.CreateCommand() ) {
```



```
        cmd.CommandText =
            "INSERT INTO Names(name) VALUES (?)";
        cmd.Parameters.Add("", txtName.Text);
        RowsInserted = cmd.ExecuteNonQuery();
    }
    RefreshListBox();
}
catch( Exception err ) {
    MessageBox.Show("Exception: " + err.Message );
}
```

Visual Basic の場合は、次のコードを btnInsert_Click メソッドに追加します。

```
'Visual Basic
Try
    Dim RowsInserted As Long
    Dim cmd As ULCommand = Conn.CreateCommand()
    cmd.CommandText = "INSERT INTO Names(name) VALUES (?)"
    cmd.Parameters.Add("", txtName.Text)
    RowsInserted = cmd.ExecuteNonQuery()
    cmd.Dispose()
    RefreshListBox()
Catch
    MsgBox("Exception: " + Err.Description)
End Try
```

2. [フォーム デザイン] タブで **[Update]** をダブルクリックして、btnUpdate_Click メソッドを作成します。このメソッドでは、次のタスクを実行します。

   - ULCommand オブジェクトをインスタンス化し、関連 ID に基づいて、テキスト・ボックス内の値をデータベースに挿入する UPDATE 文に割り当てます。
   - 文を実行します。
   - ULCommand オブジェクトを破棄します。
   - リストボックスをリフレッシュします。
   - エラーが発生した場合は、エラー・メッセージが出力されます。SQL エラーの場合は、エラー・コードも出力されます。

     エラー・メッセージを参照してください。

   C# の場合は、次のコードを btnUpdate_Click メソッドに追加します。

```
//Visual C#
try {
    long RowsUpdated;
    int updateID = ids[ lbNames.SelectedIndex ];
    using( ULCommand cmd = Conn.CreateCommand() ){
        cmd.CommandText =
            "UPDATE Names SET name = ? WHERE id = ?" ;
        cmd.Parameters.Add("", txtName.Text );
        cmd.Parameters.Add("", updateID);
        RowsUpdated = cmd.ExecuteNonQuery();
    }
    RefreshListBox();
}
catch( Exception err ) {
    MessageBox.Show(
        "Exception: " + err.Message);
}
```

Visual Basic の場合は、次のコードを btnUpdate_Click メソッドに追加します。

```
'Visual Basic
Try
   Dim RowsUpdated As Long
   Dim updateID As Integer = ids(lbNames.SelectedIndex)
   Dim cmd As ULCommand = Conn.CreateCommand()
   cmd.CommandText = "UPDATE Names SET name = ? WHERE id = ?"
   cmd.Parameters.Add("", txtName.Text)
   cmd.Parameters.Add("", updateID)
   RowsUpdated = cmd.ExecuteNonQuery()
   cmd.Dispose()
   RefreshListBox()
Catch
   MsgBox("Exception: " + Err.Description)
End Try
```

3. [フォーム デザイン] タブで **[Delete]** をダブルクリックして、btnDelete_Click メソッドを作成します。コードを追加し、次のタスクを実行します。

   - ULCommand オブジェクトをインスタンス化し、DELETE 文に割り当てます。DELETE 文は、整数配列 ids からの関連 ID に基づいて、選択した行をデータベースから削除します。
   - 文を実行します。
   - ULCommand オブジェクトを破棄します。
   - リストボックスをリフレッシュします。
   - エラーが発生した場合は、エラー・メッセージが表示されます。SQL エラーの場合は、エラー・コードも表示されます。

     エラー・メッセージを参照してください。

   C# の場合は、次のコードを btnDelete_Click メソッドに追加します。

```
//Visual C#
try{
   long RowsDeleted;
   int deleteID = ids[lbNames.SelectedIndex];
   using( ULCommand cmd = Conn.CreateCommand() ){
      cmd.CommandText =
         "DELETE From Names WHERE id = ?" ;
      cmd.Parameters.Add("", deleteID);
      RowsDeleted = cmd.ExecuteNonQuery ();
   }
   RefreshListBox();
}
catch( Exception err ) {
   MessageBox.Show("Exception: " + err.Message );
}
```

   Visual Basic の場合は、次のコードを btnDelete_Click メソッドに追加します。

```
'Visual Basic
Try
   Dim RowsDeleted As Long
   Dim deleteID As Integer = ids(lbNames.SelectedIndex)
   Dim cmd As ULCommand = Conn.CreateCommand()
   cmd.CommandText = "DELETE From Names WHERE id = ?"
   cmd.Parameters.Add("", deleteID)
   RowsDeleted = cmd.ExecuteNonQuery()
   cmd.Dispose()
```



```
    RefreshListBox()
Catch
    MsgBox("Exception: " + Err.Description)
End Try
```

4. アプリケーションをビルドし、正常にコンパイルされることを確認します。

レッスン 5：アプリケーションのビルドと配置

次の手順では、アプリケーションをビルドし、リモート・デバイスまたはエミュレータに配置します。

♦ **アプリケーションを配置するには、次の手順に従います。**

1. ソリューションをビルドします。

   アプリケーションがエラーなくビルドされることを確認します。

2. 配置ターゲットを選択します。

   配置ターゲットは、アプリケーションに含める *ulnet11.dll* のバージョンと一致している必要があります。

3. **[デバッグ] - [開始]** を選択します。

   これにより、アプリケーションが格納された実行可能ファイルがビルドされ、エミュレータに配置されます。このプロセスには時間がかかることがあります。アプリケーションを実行する前に .NET Compact Framework を配置する必要がある場合は、特に時間がかかります。

配置のトラブルシューティングのチェックリスト

エラーがレポートされた場合は、次のチェックリストを使用して、配置が正常に完了したかどうかをチェックします。

- アプリケーションが *¥Program Files¥appname* に配置されていることを確認します。*appname* は、レッスン 1 でアプリケーションに付けた名前 (CSApp または VBApp) です。
- アプリケーション・コード内のデータベース・ファイルへのパスが正しいことを確認します。「レッスン 3 ：データベースへの接続」 41 ページを参照してください。
- データベース・ファイルをプロジェクトに追加するときに [リンク ファイル] を選択し、[ビルド アクション] を [コンテンツ] に設定し、[出力ディレクトリにコピー] を [常にコピーする] に設定したことを確認します。これらのオプションを適切に設定しなかった場合は、ファイルがデバイスに配置されません。
- ターゲット・プラットフォームに対して適切なバージョンの *ulnet11.dll* への参照を追加したこと、または Windows Mobile インストーラを実行したことを確認します。Windows Mobile 5.0 より前のバージョンの Windows Mobile で、エミュレータと実際のデバイスの切り替えを行う場合は、使用するライブラリのバージョンを変更する必要があります。「レッスン 1：Visual Studio プロジェクトの作成」 35 ページを参照してください。
- エミュレータの状態を保存しないでエミュレータを終了できます。アプリケーションの再配置を行うと、必要なすべてのファイルがエミュレータにコピーされ、バージョン上の問題がないことが確認されます。

♦ **アプリケーションをテストするには、次の手順に従います。**

1. データベースにデータを挿入します。

テキスト・ボックスに名前を入力し、**[Insert]** をクリックします。名前がリストボックスに表示されます。

2. データベース内のデータを更新します。

   リストボックスから名前を選択します。テキスト・ボックスに新しい名前を入力します。**[Update]** をクリックします。新しい名前が、リストボックスの古い名前の代わりに表示されます。

3. データベースからデータを削除します。

   リストから名前を選択します。**[Delete]** をクリックします。名前はリストに表示されなくなります。

このチュートリアルは、これで終了です。

C# チュートリアルのコード・リスト

この章で説明したチュートリアル・プログラムの完全なコードを次に示します。

```
using iAnywhere.Data.UltraLite;
using System;
using System.Linq;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Text;
using System.Windows.Forms;

namespace CSApp
{
   public partial class Form1 : Form
   {
      public Form1()
      {
         InitializeComponent();
      }
      private ULConnection Conn;
      private int[] ids;

      private void Form1_Load(object sender, EventArgs e)
      {
         try
         {
            String ConnString = "dbf=¥¥Program Files¥¥CSApp¥¥CSApp.udb";
            Conn = new ULConnection(ConnString);
            Conn.Open();
            Conn.DatabaseID = 1;
            RefreshListBox();
         }
         catch (System.Exception t)
         {
            MessageBox.Show("Exception: " + t.Message);
         }
      }
      private void RefreshListBox()
      {
         try
         {
            long NumRows;
            int I = 0;
            lbNames.Items.Clear();
            using (ULCommand cmd = Conn.CreateCommand())
            {
               cmd.CommandText = "SELECT ID, Name FROM Names";
               using (ULDataReader dr = cmd.ExecuteReader())
               {
                  dr.MoveBeforeFirst();
                  NumRows = dr.RowCount;
                  ids = new int[NumRows];
                  while (dr.MoveNext())
                  {
                     lbNames.Items.Add(
                     dr.GetString(1));
                     ids[i] = dr.GetInt32(0);
                     I++;
                  }
```



```
            }
            txtName.Text = " ";
        }
    }
    catch (Exception err)
    {
        MessageBox.Show(
        "Exception in RefreshListBox: " + err.Message);
    }
}

private void btnInsert_Click(object sender, EventArgs e)
{
    try
    {
        long RowsInserted;
        using (ULCommand cmd = Conn.CreateCommand())
        {
            cmd.CommandText =
                "INSERT INTO Names(name) VALUES (?)";
            cmd.Parameters.Add("", txtName.Text);
            RowsInserted = cmd.ExecuteNonQuery();
        }
        RefreshListBox();
    }
    catch (Exception err)
    {
        MessageBox.Show("Exception: " + err.Message);
    }
}

private void btnUpdate_Click(object sender, EventArgs e)
{
    try
    {
        long RowsUpdated;
        int updateID = ids[lbNames.SelectedIndex];
        using (ULCommand cmd = Conn.CreateCommand())
        {
            cmd.CommandText =
                "UPDATE Names SET name = ? WHERE id = ?";
            cmd.Parameters.Add("", txtName.Text);
            cmd.Parameters.Add("", updateID);
            RowsUpdated = cmd.ExecuteNonQuery();
        }
        RefreshListBox();
    }
    catch (Exception err)
    {
        MessageBox.Show(
            "Exception: " + err.Message);
    }
}

private void btnDelete_Click(object sender, EventArgs e)
{
    try
    {
        long RowsDeleted;
        int deleteID = ids[lbNames.SelectedIndex];
        using (ULCommand cmd = Conn.CreateCommand())
        {
            cmd.CommandText =
                "DELETE From Names WHERE id = ?";
```

```
                cmd.Parameters.Add("", deleteID);
                RowsDeleted = cmd.ExecuteNonQuery();
            }
            RefreshListBox();
        }
        catch (Exception err)
        {
            MessageBox.Show("Exception: " + err.Message);
        }
    }
  }
}
```


Visual Basic チュートリアルのコード・リスト

```
Imports iAnywhere.Data.UltraLite
Public Class Form1
    Dim Conn As ULConnection
    Dim ids() As Integer
    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As
      System.EventArgs) Handles MyBase.Load
        Try
            Dim ConnString As String = "dbf=¥Program Files¥VBApp¥VBApp.udb"
            Conn = New ULConnection(ConnString)
            Conn.Open()
            Conn.DatabaseID = 1
            RefreshListBox()
        Catch
            MsgBox("Exception: " + Err.Description)
        End Try
    End Sub
    Private Sub RefreshListBox()
        Try
            Dim cmd As ULCommand = Conn.CreateCommand()
            Dim I As Integer = 0
            lbNames.Items.Clear()
            cmd.CommandText = "SELECT ID, Name FROM Names"
            Dim dr As ULDataReader = cmd.ExecuteReader()
            ReDim ids(dr.RowCount)
            While (dr.MoveNext)
                lbNames.Items.Add(dr.GetString(1))
                ids(I) = dr.GetInt32(0)
                I = I + 1
            End While
            dr.Close()
            txtName.Text = " "
        Catch ex As Exception
            MsgBox(ex.ToString)
        End Try
    End Sub

    Private Sub btnInsert_Click(ByVal sender As System.Object, ByVal e As
      System.EventArgs) Handles btnInsert.Click
        Try
            Dim RowsInserted As Long
            Dim cmd As ULCommand = Conn.CreateCommand()
            cmd.CommandText = "INSERT INTO Names(name) VALUES (?)"
            cmd.Parameters.Add("", txtName.Text)
            RowsInserted = cmd.ExecuteNonQuery()
            cmd.Dispose()
            RefreshListBox()
        Catch
            MsgBox("Exception: " + Err.Description)
        End Try
    End Sub

    Private Sub btnUpdate_Click(ByVal sender As System.Object, ByVal e As
      System.EventArgs) Handles btnUpdate.Click
        Try
            Dim RowsUpdated As Long
            Dim updateID As Integer = ids(lbNames.SelectedIndex)
            Dim cmd As ULCommand = Conn.CreateCommand()
            cmd.CommandText = "UPDATE Names SET name = ? WHERE id = ?"
            cmd.Parameters.Add("", txtName.Text)
            cmd.Parameters.Add("", updateID)
```

```
            RowsUpdated = cmd.ExecuteNonQuery()
            cmd.Dispose()
            RefreshListBox()
        Catch
            MsgBox("Exception: " + Err.Description)
        End Try
    End Sub

    Private Sub btnDelete_Click(ByVal sender As System.Object, ByVal e As
      System.EventArgs) Handles btnDelete.Click
        Try
            Dim RowsDeleted As Long
            Dim deleteID As Integer = ids(lbNames.SelectedIndex)
            Dim cmd As ULCommand = Conn.CreateCommand()
            cmd.CommandText = "DELETE From Names WHERE id = ?"
            cmd.Parameters.Add("", deleteID)
            RowsDeleted = cmd.ExecuteNonQuery()
            cmd.Dispose()
            RefreshListBox()
        Catch
            MsgBox("Exception: " + Err.Description)
        End Try
    End Sub
End Class
```


Ultra Light .NET 2.0 API リファレンス

目次

ULActiveSyncListener インタフェース 57
ULAuthStatusCode 列挙体 61
ULBulkCopy クラス 62
ULBulkCopyColumnMapping クラス 74
ULBulkCopyColumnMappingCollection クラス 81
ULBulkCopyOptions 列挙体 90
ULCommand クラス 91
ULCommandBuilder クラス 129
ULConnection クラス 139
ULConnectionParms クラス 189
ULConnectionParms.UnusedEventHandler デリゲート 202
ULConnectionStringBuilder クラス 203
ULCreateParms クラス 221
ULCursorSchema クラス 233
ULDataAdapter クラス 242
ULDatabaseManager クラス 255
ULDatabaseSchema クラス 265
ULDataReader クラス 276
ULDateOrder 列挙体 316
ULDbType 列挙体 317
ULDBValid 列挙体 321
ULException クラス 322
ULFactory クラス 326
ULFileTransfer クラス 332
ULFileTransferProgressData クラス 348
ULFileTransferProgressListener インタフェース 351
ULIndexSchema クラス 353
ULInfoMessageEventArgs クラス 362
ULInfoMessageEventHandler デリゲート 365
ULMetaDataCollectionNames クラス 366
ULParameter クラス 375

ULParameterCollection クラス **392**
ULPublicationSchema クラス **412**
ULResultSet クラス **415**
ULResultSetSchema クラス **442**
ULRowsCopiedEventArgs クラス **445**
ULRowsCopiedEventHandler デリゲート **448**
ULRowUpdatedEventArgs クラス **449**
ULRowUpdatedEventHandler デリゲート **453**
ULRowUpdatingEventArgs クラス **454**
ULRowUpdatingEventHandler デリゲート **457**
ULRuntimeType 列挙体 **458**
ULServerSyncListener インタフェース **459**
ULSQLCode 列挙体 **462**
ULSqlPassthroughProgressListener インタフェース **477**
ULSqlProgressData クラス **479**
ULSqlProgressState 列挙体 **481**
ULStreamErrorCode 列挙体 **482**
ULStreamType 列挙体 **512**
ULSyncParms クラス **514**
ULSyncProgressData クラス **528**
ULSyncProgressListener インタフェース **538**
ULSyncProgressState 列挙体 **540**
ULSyncResult クラス **542**
ULTable クラス **548**
ULTableSchema クラス **571**
ULTransaction クラス **586**

ネームスペース

iAnywhere.Data.UltraLite ネームスペース

ULActiveSyncListener インタフェース

UL 拡張：ActiveSync イベントを受信するリスナ・インタフェースです。

構文

Visual Basic
Public Interface **ULActiveSyncListener**

C#
public interface **ULActiveSyncListener**

参照

- 「ULActiveSyncListener メンバ」 57 ページ

ULActiveSyncListener メンバ

パブリック・メソッド

メンバ名	説明
「ActiveSyncInvoked メソッド」 57 ページ	ActiveSync 用の Mobile Link プロバイダがアプリケーションを呼び出して同期を実行するときに起動されます。

参照

- 「ULActiveSyncListener インタフェース」 57 ページ

ActiveSyncInvoked メソッド

ActiveSync 用の Mobile Link プロバイダがアプリケーションを呼び出して同期を実行するときに起動されます。

構文

Visual Basic
```
Public Sub ActiveSyncInvoked( _
  ByVal launchedByProvider As Boolean _
)
```

C#
```
public void ActiveSyncInvoked(
  bool launchedByProvider
);
```

パラメータ

- **launchedByProvider**　ActiveSync 同期を実行するために、Mobile Link プロバイダがアプリケーションを起動した場合は true です。同期の完了後、アプリケーションは自動的に停止し

なければなりません。ActiveSync 用 Mobile Link プロバイダが呼び出したときに、すでにアプリケーションが実行されていた場合は false です。

備考

このメソッドは、別のスレッドによって呼び出されます。マルチスレッドの問題を回避するには、このメソッドがユーザ・インタフェースにイベントを通知する必要があります。マルチスレッドを使用する場合は、スレッドごとに別々の接続を使用し、lock キーワードを使用して、アプリケーションの共有オブジェクトにアクセスすることをおすすめします。

同期が完了したら、アプリケーションは ULDatabaseManager.SignalSyncIsComplete() を呼び出して、ActiveSync 用の Mobile Link プロバイダに対して通知する必要があります。

例

次のコード・フラグメントは、ActiveSync 要求を受信して、UI スレッドで同期を実行する方法を示しています。

```
' Visual Basic
Imports iAnywhere.Data.UltraLite

Public Class MainWindow
  Inherits System.Windows.Forms.Form
  Implements ULActiveSyncListener
  Private conn As ULConnection

  Public Sub New(ByVal args() As String)

    MyBase.New()

    'This call is required by the Windows Form Designer.
    InitializeComponent()

    'Add any initialization after the InitializeComponent() call
    ULConnection.DatabaseManager.SetActiveSyncListener( _
        "myCompany.myapp", Me _
      )
    'Create Connection
    ...
  End Sub

  Protected Overrides Sub OnClosing( _
      ByVal e As System.ComponentModel.CancelEventArgs _
    )
    ULConnection.DatabaseManager.SetActiveSyncListener( _
      Nothing, Nothing _
    )
    MyBase.OnClosing(e)
  End Sub

  Public Sub ActiveSyncInvoked( _
      ByVal launchedByProvider As Boolean _
    ) Implements ULActiveSyncListener.ActiveSyncInvoked
    Me.Invoke(New EventHandler(AddressOf Me.ActiveSyncAction))
  End Sub

  Public Sub ActiveSyncAction( _
      ByVal sender As Object, ByVal e As EventArgs _
    )
    ' Do active sync
```



```
    conn.Synchronize()
    ULConnection.DatabaseManager.SignalSyncIsComplete()
  End Sub
End Class


// C#
using iAnywhere.Data.UltraLite;
public class Form1 : System.Windows.Forms.Form, ULActiveSyncListener
{
  private System.Windows.Forms.MainMenu mainMenu1;
  private ULConnection conn;

  public Form1()
  {
    //
    // Required for Windows Form Designer support
    //
    InitializeComponent();

    //
    // TODO: Add any constructor code after
    // InitializeComponent call
    //
    ULConnection.DatabaseManager.SetActiveSyncListener(
        "myCompany.myapp", this
      );
    // Create connection
    ...
  }

  protected override void Dispose( bool disposing )
  {
    base.Dispose( disposing );
  }

  protected override void OnClosing(
      System.ComponentModel.CancelEventArgs e
    )
  {
    ULConnection.DatabaseManager.SetActiveSyncListener(
        null, null
      );
    base.OnClosing(e);
  }

  public void ActiveSyncInvoked(bool launchedByProvider)
  {
    this.Invoke( new EventHandler( ActiveSyncHandler ) );
  }

  internal void ActiveSyncHandler(object sender, EventArgs e)
  {
    conn.Synchronize();
    ULConnection.DatabaseManager.SignalSyncIsComplete();
  }
}
```

参照

- 「ULActiveSyncListener インタフェース」 57 ページ
- 「ULActiveSyncListener メンバ」 57 ページ
- 「SignalSyncIsComplete メソッド」 263 ページ

ULAuthStatusCode 列挙体

UL 拡張： Mobile Link ユーザ認証の実行中にレポートされる可能性のあるステータス・コードを列挙します。

構文

Visual Basic
Public Enum **ULAuthStatusCode**

C#
public enum **ULAuthStatusCode**

メンバ

メンバ名	説明	値
EXPIRED	ユーザ ID またはパスワードの有効期限が切れているため、認証に失敗しました (EXPIRED = 3)。	3
IN_USE	ユーザ ID がすでに使用されているため、認証に失敗しました (IN_USE = 5)。	5
INVALID	ユーザ ID またはパスワードが正しくないため、認証に失敗しました (INVALID = 4)。	4
UNKNOWN	認証ステータスが不明です。接続がまだ同期を実行していない可能性があります (UNKNOWN = 0)。	0
VALID	ユーザ ID とパスワードは、同期時には有効でした (VALID = 1)。	1
VALID_BUT_EXPIRES_SOON	ユーザ ID とパスワードは、同期時には有効でしたが、まもなく有効期限が切れます (VALID_BUT_EXPIRES_SOON = 2)。	2

参照

- 「AuthStatus プロパティ」 543 ページ

ULBulkCopy クラス

別のソースのデータを使用して、Ultra Light テーブルを効率的にバルク・ロードします。このクラスは継承できません。

構文

Visual Basic
Public NotInheritable Class **ULBulkCopy**
  Implements IDisposable

C#
public sealed class **ULBulkCopy** : IDisposable

備考

制限：ULBulkCopy クラスは、.NET Compact Framework 2.0 では使用できません。

参照

- 「ULBulkCopy メンバ」 62 ページ

ULBulkCopy メンバ

パブリック・コンストラクタ

メンバ名	説明
「ULBulkCopy コンストラクタ」 63 ページ	ULBulkCopy オブジェクトを、指定された ULConnection オブジェクトで初期化します。

パブリック・プロパティ

メンバ名	説明
「BatchSize プロパティ」 66 ページ	各バッチの中のローの数を取得または設定します。各バッチが終了すると、バッチ内のローがサーバに送信されます。
「BulkCopyTimeOut プロパティ」 67 ページ	タイムアウトするまでに完了するオペレーションの秒数を取得または設定します。
「ColumnMappings プロパティ」 68 ページ	ULBulkCopyColumnMapping 項目のコレクションを返します。カラム・マッピングは、データ・ソース内のカラムと、送信先のカラムの間の関係を定義します。
「DestinationTableName プロパティ」 68 ページ	サーバ上の送信先テーブルの名前を取得または設定します。

メンバ名	説明
「NotifyAfter プロパティ」 69 ページ	通知イベントが生成されるまでに処理されるローの数を指定します。

パブリック・メソッド

メンバ名	説明
「Close メソッド」 69 ページ	ULBulkCopy インスタンスを閉じます。
「Dispose メソッド」 70 ページ	ULBulkCopy インスタンスを破棄します。
「WriteToServer メソッド」 70 ページ	指定された System.Data.DataRow オブジェクトの配列内のすべてのローを、ULBulkCopy オブジェクトの DestinationTableName フィールドで指定される送信先テーブルにコピーします。

パブリック・イベント

メンバ名	説明
「ULRowsCopied イベント」 73 ページ	NotifyAfter で指定される数のローが処理されるたびに、このイベントが発生します。

参照

- 「ULBulkCopy クラス」 62 ページ

ULBulkCopy コンストラクタ

ULBulkCopy オブジェクトを、指定された ULConnection オブジェクトで初期化します。

参照

- 「ULConnection クラス」 139 ページ

ULBulkCopy(ULConnection) コンストラクタ

構文

Visual Basic

```
Public Sub New( _
  ByVal connection As ULConnection _
)
```

C#
```
public ULBulkCopy(
  ULConnection connection
);
```

パラメータ

- **connection** バルク・コピー・オペレーションの実行に使用する、すでに開いている ULConnection。接続が開いていない場合は、WriteToServer に例外がスローされます。

備考

制限：ULBulkCopy クラスは、.NET Compact Framework 2.0 では使用できません。

参照

- 「ULBulkCopy クラス」 62 ページ
- 「ULBulkCopy メンバ」 62 ページ
- 「ULBulkCopy コンストラクタ」 63 ページ
- 「ULConnection クラス」 139 ページ

ULBulkCopy(String) コンストラクタ

ULBulkCopy オブジェクトを、指定された接続文字列で初期化します。

構文

Visual Basic
```
Public Sub New( _
  ByVal connectionString As String _
)
```

C#
```
public ULBulkCopy(
  string connectionString
);
```

パラメータ

- **connectionString** ULBulkCopy インスタンスによって使用されるために開かれる接続を定義する文字列。接続文字列は keyword=value のペアがセミコロンで区切られたリストです。

  パラメータのリストについては、「ConnectionString プロパティ」 146 ページを参照してください。

備考

この構文は、connectionString を使用して WriteToServer の実行中に接続を開きます。WriteToServer が終了すると、接続が閉じます。

接続文字列は、ULConnectionParms オブジェクトを使用して指定できます。

制限：ULBulkCopy クラスは、.NET Compact Framework 2.0 では使用できません。

実装：System.IDisposable

参照

- 「ULBulkCopy クラス」 62 ページ
- 「ULBulkCopy メンバ」 62 ページ
- 「ULBulkCopy コンストラクタ」 63 ページ
- 「ULConnectionParms クラス」 189 ページ
- IDisposable

ULBulkCopy(String, ULBulkCopyOptions) コンストラクタ

ULBulkCopy オブジェクトを、指定された接続文字列とコピー・オプションで初期化します。

構文

Visual Basic

```
Public Sub New( _
  ByVal connectionString As String, _
  ByVal copyOptions As ULBulkCopyOptions _
)
```

C#

```
public  ULBulkCopy(
  string  connectionString,
  ULBulkCopyOptions copyOptions
);
```

パラメータ

- **connectionString**　ULBulkCopy インスタンスによって使用されるために開かれる接続を定義する文字列。接続文字列は keyword=value のペアがセミコロンで区切られたリストです。

  パラメータのリストについては、「ConnectionString プロパティ」 146 ページを参照してください。

- **copyOptions**　目的のテーブルにデータ・ソース・ローをコピーする方法を決定する、ULBulkCopyOptions 列挙体の値の組み合わせ。

備考

この構文は、connectionString を使用して WriteToServer の実行中に接続を開きます。WriteToServer が終了すると、接続が閉じます。

制限：ULBulkCopy クラスは、.NET Compact Framework 2.0 では使用できません。

参照

- 「ULBulkCopy クラス」 62 ページ
- 「ULBulkCopy メンバ」 62 ページ
- 「ULBulkCopy コンストラクタ」 63 ページ
- 「ULBulkCopyOptions 列挙体」 90 ページ

ULBulkCopy(ULConnection, ULBulkCopyOptions, ULTransaction) コンストラクタ

ULBulkCopy オブジェクトを、指定された ULConnection、コピー・オプション、ULTransaction で初期化します。

構文

Visual Basic

```
Public Sub New( _
  ByVal connection As ULConnection, _
  ByVal copyOptions As ULBulkCopyOptions, _
  ByVal externalTransaction As ULTransaction _
)
```

C#

```
public  ULBulkCopy(
  ULConnection connection,
  ULBulkCopyOptions copyOptions,
  ULTransaction externalTransaction
);
```

パラメータ

- **connection**　バルク・コピー・オペレーションの実行に使用する、すでに開いている ULConnection。接続が開いていない場合は、WriteToServer に例外がスローされます。
- **copyOptions**　目的のテーブルにデータ・ソース・ローをコピーする方法を決定する、ULBulkCopyOptions 列挙体の値の組み合わせ。
- **externalTransaction**　バルク・コピーが発生する、既存の ULTransaction インスタンス。externalTransaction が NULL 参照 (Visual Basic の Nothing) でないと、バルク・コピー操作が内部で実行されます。外部トランザクションと ULBulkCopyOptions.UseInternalTransaction オプションの両方を指定すると、エラーになります。

備考

制限：ULBulkCopy クラスは、.NET Compact Framework 2.0 では使用できません。

参照

- 「ULBulkCopy クラス」 62 ページ
- 「ULBulkCopy メンバ」 62 ページ
- 「ULBulkCopy コンストラクタ」 63 ページ
- 「ULConnection クラス」 139 ページ
- 「ULTransaction クラス」 586 ページ

BatchSize プロパティ

各バッチの中のローの数を取得または設定します。各バッチが終了すると、バッチ内のローがサーバに送信されます。

構文

Visual Basic
Public Property **BatchSize** As Integer

C#
public int **BatchSize** { get; set; }

プロパティ値

各バッチの中のロー数。デフォルトは 0 です。

備考

0 に設定すると、すべてのローが 1 つのバッチで送信されます。

0 未満の値を設定すると、エラーになります。

バッチの進行中にこの値が変更された場合、現在のバッチはそのまま完了し、それ以降のバッチが新しい値を使用します。

参照

- 「ULBulkCopy クラス」 62 ページ
- 「ULBulkCopy メンバ」 62 ページ

BulkCopyTimeOut プロパティ

タイムアウトするまでに完了するオペレーションの秒数を取得または設定します。

構文

Visual Basic
Public Property **BulkCopyTimeout** As Integer

C#
public int **BulkCopyTimeout** { get; set; }

プロパティ値

デフォルト値は 30 秒です。

備考

値が 0 の場合、制限はありません。この場合、待機時間が無限になる可能性があるため、値は 0 にしないでください。

オペレーションがタイムアウトすると、現在のトランザクション内のすべてのローがロールバックされ、SAException が発生します。

0 未満の値を設定すると、エラーになります。

参照

- 「ULBulkCopy クラス」 62 ページ
- 「ULBulkCopy メンバ」 62 ページ

ColumnMappings プロパティ

ULBulkCopyColumnMapping 項目のコレクションを返します。カラム・マッピングは、データ・ソース内のカラムと、送信先のカラムの間の関係を定義します。

構文

Visual Basic
Public Readonly Property **ColumnMappings** As ULBulkCopyColumnMappingCollection

C#
public ULBulkCopyColumnMappingCollection **ColumnMappings** { get;}

プロパティ値

デフォルトでは、空のコレクションです。

備考

WriteToServer の実行中は、プロパティを変更できません。

WriteToServer の実行時に ColumnMappings が空の場合、ソース内の先頭のカラムが送信先の先頭のカラムにマッピングされ、2 番目は 2 番目にマッピングされます。以降についても同様です。この処理は、カラムの型が変換可能な場合、ソース・カラム以上の送信先カラムがある場合、余分な送信先カラムが NULL 入力可のカラムである場合に行われます。

参照

- 「ULBulkCopy クラス」 62 ページ
- 「ULBulkCopy メンバ」 62 ページ
- 「ULBulkCopyColumnMapping クラス」 74 ページ

DestinationTableName プロパティ

サーバ上の送信先テーブルの名前を取得または設定します。

構文

Visual Basic
Public Property **DestinationTableName** As String

C#
public string **DestinationTableName** { get; set; }

プロパティ値

デフォルト値は NULL 参照 (Visual Basic の Nothing) です。

備考

WriteToServer の実行時に値が変更されても、変更は反映されません。

WriteToServer への呼び出しの前に値が設定されていない場合、InvalidOperationException が発生します。

値を NULL (Visual Basic の場合は Nothing) または空の文字列に設定すると、エラーになります。

参照

- 「ULBulkCopy クラス」 62 ページ
- 「ULBulkCopy メンバ」 62 ページ

NotifyAfter プロパティ

通知イベントが生成されるまでに処理されるローの数を指定します。

構文

Visual Basic
Public Property **NotifyAfter** As Integer

C#
public int **NotifyAfter** { get; set; }

プロパティ値

通知イベントが生成されるまでに処理されるローの数を表す整数。プロパティが設定されていない場合は 0。

備考

WriteToServer の実行時に加えられる NotifyAfter への変更は、次の通知まで反映されません。

0 未満の値を設定すると、エラーになります。

NotifyAfter と BulkCopyTimeOut の値は相互に排他的なため、データベースにローが送信されなかったり、コミットされない場合であっても、イベントは起動します。

参照

- 「ULBulkCopy クラス」 62 ページ
- 「ULBulkCopy メンバ」 62 ページ
- 「BulkCopyTimeOut プロパティ」 67 ページ

Close メソッド

ULBulkCopy インスタンスを閉じます。

構文

Visual Basic
Public Sub **Close()**

C#
public void **Close();**

参照

- 「ULBulkCopy クラス」 62 ページ
- 「ULBulkCopy メンバ」 62 ページ

Dispose メソッド

ULBulkCopy インスタンスを破棄します。

構文

Visual Basic
NotOverridable Public Sub **Dispose()**

C#
public void **Dispose();**

参照

- 「ULBulkCopy クラス」 62 ページ
- 「ULBulkCopy メンバ」 62 ページ

WriteToServer メソッド

指定された System.Data.DataRow オブジェクトの配列内のすべてのローを、ULBulkCopy オブジェクトの DestinationTableName フィールドで指定される送信先テーブルにコピーします。

WriteToServer(DataRow[]) メソッド

指定された System.Data.DataRow オブジェクトの配列内のすべてのローを、ULBulkCopy オブジェクトの DestinationTableName フィールドで指定される送信先テーブルにコピーします。

構文

Visual Basic
```
Public Sub WriteToServer( _
  ByVal rows As DataRow() _
)
```

C#
```
public void WriteToServer(
  DataRow[] rows
);
```


パラメータ

- **rows**　送信先テーブルにコピーされる System.Data.DataRow オブジェクトの配列。

参照

- 「ULBulkCopy クラス」 62 ページ
- 「ULBulkCopy メンバ」 62 ページ
- 「WriteToServer メソッド」 70 ページ
- DataRow
- 「DestinationTableName プロパティ」 68 ページ

WriteToServer(DataTable) メソッド

指定された System.Data.DataTable のすべてのローを、ULBulkCopy オブジェクトの DestinationTableName で指定される送信先テーブルにコピーします。

構文

Visual Basic

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

C#

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

パラメータ

- **table**　ローが送信先テーブルにコピーされる System.Data.DataTable。

参照

- 「ULBulkCopy クラス」 62 ページ
- 「ULBulkCopy メンバ」 62 ページ
- 「WriteToServer メソッド」 70 ページ
- 「DestinationTableName プロパティ」 68 ページ
- DataTable

WriteToServer(IDataReader) メソッド

指定された System.Data.IDataReader のすべてのローを、ULBulkCopy オブジェクトの DestinationTableName で指定される送信先テーブルにコピーします。

構文

Visual Basic

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

```
C#
public void WriteToServer(
  IDataReader reader
);
```

パラメータ

- **reader**　ローが送信先テーブルにコピーされる System.Data.IDataReader。

参照

- 「ULBulkCopy クラス」 62 ページ
- 「ULBulkCopy メンバ」 62 ページ
- 「WriteToServer メソッド」 70 ページ
- IDataReader
- 「DestinationTableName プロパティ」 68 ページ

WriteToServer(DataTable, DataRowState) メソッド

指定されたローの状態を持つ指定された System.Data.DataTable のすべてのローを、ULBulkCopy オブジェクトの DestinationTableName で指定される送信先テーブルにコピーします。

構文

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

C#
public void WriteToServer(
  DataTable table,
  DataRowState rowState
);
```

パラメータ

- **table**　ローが送信先テーブルにコピーされる System.Data.DataTable。
- **rowState**　System.Data.DataRowState 列挙体の値。ロー・ステータスに一致するローのみ、送信先にコピーされます。

備考

rowState が指定されている場合、ロー・ステータスが同じローだけがコピーされます。

参照

- 「ULBulkCopy クラス」 62 ページ
- 「ULBulkCopy メンバ」 62 ページ
- 「WriteToServer メソッド」 70 ページ
- 「DestinationTableName プロパティ」 68 ページ
- DataTable
- DataRowState

ULRowsCopied イベント

NotifyAfter で指定される数のローが処理されるたびに、このイベントが発生します。

構文

Visual Basic
Public Event **ULRowsCopied** As ULRowsCopiedEventHandler

C#
public event ULRowsCopiedEventHandler **ULRowsCopied**;

備考

ULRowsCopied イベントの受信は、ローがコミットされたことを意味するわけではありません。このイベントから Close メソッドを呼び出すことはできません。

参照

- 「ULBulkCopy クラス」 62 ページ
- 「ULBulkCopy メンバ」 62 ページ
- 「NotifyAfter プロパティ」 69 ページ

ULBulkCopyColumnMapping クラス

ULBulkCopy インスタンスのデータ・ソース内のカラムと、インスタンスの送信先テーブル内のカラムの間のマッピングを定義します。このクラスは継承できません。

構文

Visual Basic
Public NotInheritable Class **ULBulkCopyColumnMapping**

C#
public sealed class **ULBulkCopyColumnMapping**

備考

制限：ULBulkCopyColumnMapping クラスは、.NET Compact Framework 2.0 では使用できません。

参照

- 「ULBulkCopyColumnMapping メンバ」 74 ページ
- 「ULBulkCopy クラス」 62 ページ

ULBulkCopyColumnMapping メンバ

パブリック・コンストラクタ

メンバ名	説明
「ULBulkCopyColumnMapping コンストラクタ」 75 ページ	「ULBulkCopyColumnMapping クラス」 74 ページの新しいインスタンスを初期化します。

パブリック・プロパティ

メンバ名	説明
「DestinationColumn プロパティ」 78 ページ	マッピングされる送信先データベース・テーブルのカラムの名前を指定します。
「DestinationOrdinal プロパティ」 79 ページ	マッピングされる送信先データベース・テーブルにおけるカラムの順序を指定します。
「SourceColumn プロパティ」 79 ページ	データ・ソースにマッピングされるカラムの名前を指定します。
「SourceOrdinal プロパティ」 80 ページ	データ・ソース内のソース・カラムの順序位置を指定します。

参照

- 「ULBulkCopyColumnMapping クラス」 74 ページ
- 「ULBulkCopy クラス」 62 ページ

ULBulkCopyColumnMapping コンストラクタ

「ULBulkCopyColumnMapping クラス」 74 ページの新しいインスタンスを初期化します。

ULBulkCopyColumnMapping() コンストラクタ

新しいカラム・マッピングを作成します。

構文

Visual Basic
Public Sub **New()**

C#
public **ULBulkCopyColumnMapping();**

備考

制限：ULBulkCopyColumnMapping クラスは、.NET Compact Framework 2.0 では使用できません。

参照

- 「ULBulkCopyColumnMapping クラス」 74 ページ
- 「ULBulkCopyColumnMapping メンバ」 74 ページ
- 「ULBulkCopyColumnMapping コンストラクタ」 75 ページ

ULBulkCopyColumnMapping(Int32, Int32) コンストラクタ

カラムの順序または名前を使用してソース・カラムと送信先カラムを参照する、新しいカラム・マッピングを作成します。

構文

Visual Basic
```
Public Sub New( _
  ByVal sourceColumnOrdinal As Integer, _
  ByVal destinationColumnOrdinal As Integer _
)
```

C#
```
public ULBulkCopyColumnMapping(
  int sourceColumnOrdinal,
  int destinationColumnOrdinal
);
```

パラメータ

- **sourceColumnOrdinal**　データ・ソース内のソース・カラムの順序位置。データ・ソースの先頭カラムの順序位置は 0 です。
- **destinationColumnOrdinal**　送信先テーブル内の送信先カラムの順序位置。テーブルの先頭カラムの順序位置は 0 です。

備考

制限：ULBulkCopyColumnMapping クラスは、.NET Compact Framework 2.0 では使用できません。

参照

- 「ULBulkCopyColumnMapping クラス」 74 ページ
- 「ULBulkCopyColumnMapping メンバ」 74 ページ
- 「ULBulkCopyColumnMapping コンストラクタ」 75 ページ

ULBulkCopyColumnMapping(Int32, String) コンストラクタ

ソース・カラムを参照するカラムの順序と、送信先カラムを参照するカラム名を使用して、新しいカラム・マッピングを作成します。

構文

Visual Basic

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

C#

```
public  ULBulkCopyColumnMapping(
  int sourceColumnOrdinal,
  string  destinationColumn
);
```

パラメータ

- **sourceColumnOrdinal**　データ・ソース内のソース・カラムの順序位置。データ・ソースの先頭カラムの順序位置は 0 です。
- **destinationColumn**　送信先テーブル内の送信先カラムの名前。

備考

制限：ULBulkCopyColumnMapping クラスは、.NET Compact Framework 2.0 では使用できません。

参照

- 「ULBulkCopyColumnMapping クラス」 74 ページ
- 「ULBulkCopyColumnMapping メンバ」 74 ページ
- 「ULBulkCopyColumnMapping コンストラクタ」 75 ページ

ULBulkCopyColumnMapping(String, Int32) コンストラクタ

ソース・カラムを参照するカラム名と、送信先カラムを参照するカラムの順序を使用して、新しいカラム・マッピングを作成します。

構文

Visual Basic

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

C#

```
public ULBulkCopyColumnMapping(
  string sourceColumn,
  int destinationColumnOrdinal
);
```

パラメータ

- **sourceColumn**　データ・ソース内のソース・カラムの名前。
- **destinationColumnOrdinal**　送信先テーブル内の送信先カラムの順序位置。テーブルの先頭カラムの順序位置は 0 です。

備考

制限：ULBulkCopyColumnMapping クラスは、.NET Compact Framework 2.0 では使用できません。

参照

- 「ULBulkCopyColumnMapping クラス」 74 ページ
- 「ULBulkCopyColumnMapping メンバ」 74 ページ
- 「ULBulkCopyColumnMapping コンストラクタ」 75 ページ

ULBulkCopyColumnMapping(String, String) コンストラクタ

カラムの名前を使用してソース・カラムと送信先カラムを参照する、新しいカラム・マッピングを作成します。

構文

Visual Basic

```
Public Sub New( _
  ByVal sourceColumn As String, _
  ByVal destinationColumn As String _
)
```

C#

```
public ULBulkCopyColumnMapping(
  string sourceColumn,
  string destinationColumn
);
```

パラメータ

- **sourceColumn**　データ・ソース内のソース・カラムの名前。
- **destinationColumn**　送信先テーブル内の送信先カラムの名前。

備考

制限：ULBulkCopyColumnMapping クラスは、.NET Compact Framework 2.0 では使用できません。

参照

- 「ULBulkCopyColumnMapping クラス」 74 ページ
- 「ULBulkCopyColumnMapping メンバ」 74 ページ
- 「ULBulkCopyColumnMapping コンストラクタ」 75 ページ

DestinationColumn プロパティ

マッピングされる送信先データベース・テーブルのカラムの名前を指定します。

構文

Visual Basic
Public Property **DestinationColumn** As String

C#
public string **DestinationColumn** { get; set; }

プロパティ値

送信先テーブルのカラムの名前を指定する文字列。DestinationOrdinal に優先度がない場合は NULL 参照 (Visual Basic の Nothing)。

備考

DestinationColumn プロパティと DestinationOrdinal プロパティは、相互に排他的です。直前に設定された値が優先されます。

DestinationColumn プロパティを設定すると、DestinationOrdinal プロパティは -1 に設定されます。DestinationOrdinal プロパティを設定すると、DestinationColumn プロパティは NULL 参照 (Visual Basic の Nothing) に設定されます。

DestinationColumn を NULL または空の文字列に設定すると、エラーになります。

参照

- 「ULBulkCopyColumnMapping クラス」 74 ページ
- 「ULBulkCopyColumnMapping メンバ」 74 ページ
- 「DestinationOrdinal プロパティ」 79 ページ
- 「DestinationOrdinal プロパティ」 79 ページ

DestinationOrdinal プロパティ

マッピングされる送信先データベース・テーブルにおけるカラムの順序を指定します。

構文

Visual Basic
Public Property **DestinationOrdinal** As Integer

C#
public int **DestinationOrdinal** { get; set; }

プロパティ値

マッピングされるカラムの送信先テーブルにおける順序を指定する整数。プロパティが設定されていない場合は -1。

備考

DestinationColumn プロパティと DestinationOrdinal プロパティは、相互に排他的です。直前に設定された値が優先されます。

DestinationColumn プロパティを設定すると、DestinationOrdinal プロパティは -1 に設定されます。DestinationOrdinal プロパティを設定すると、DestinationColumn プロパティは NULL 参照 (Visual Basic の Nothing) に設定されます。

参照

- 「ULBulkCopyColumnMapping クラス」 74 ページ
- 「ULBulkCopyColumnMapping メンバ」 74 ページ
- 「DestinationColumn プロパティ」 78 ページ
- 「DestinationColumn プロパティ」 78 ページ

SourceColumn プロパティ

データ・ソースにマッピングされるカラムの名前を指定します。

構文

Visual Basic
Public Property **SourceColumn** As String

C#
public string **SourceColumn** { get; set; }

プロパティ値

データ・ソースのカラムの名前を指定する文字列。SourceOrdinal に優先度がない場合は NULL 参照 (Visual Basic の Nothing)。

備考

SourceColumn プロパティと SourceOrdinal プロパティは、相互に排他的です。直前に設定された値が優先されます。

SourceColumn プロパティを設定すると、SourceOrdinal プロパティは -1 に設定されます。SourceOrdinal プロパティを設定すると、SourceColumn プロパティは NULL 参照 (Visual Basic の Nothing) に設定されます。

SourceColumn を NULL または空の文字列に設定すると、エラーになります。

参照

- 「ULBulkCopyColumnMapping クラス」 74 ページ
- 「ULBulkCopyColumnMapping メンバ」 74 ページ
- 「SourceOrdinal プロパティ」 80 ページ
- 「SourceOrdinal プロパティ」 80 ページ

SourceOrdinal プロパティ

データ・ソース内のソース・カラムの順序位置を指定します。

構文

Visual Basic
Public Property **SourceOrdinal** As Integer

C#
public int **SourceOrdinal** { get; set; }

プロパティ値

データ・ソースのカラムの順序を指定する整数。プロパティが設定されていない場合は -1。

備考

SourceColumn プロパティと SourceOrdinal プロパティは、相互に排他的です。直前に設定された値が優先されます。

SourceColumn プロパティを設定すると、SourceOrdinal プロパティは -1 に設定されます。SourceOrdinal プロパティを設定すると、SourceColumn プロパティは NULL 参照 (Visual Basic の Nothing) に設定されます。

参照

- 「ULBulkCopyColumnMapping クラス」 74 ページ
- 「ULBulkCopyColumnMapping メンバ」 74 ページ
- 「SourceColumn プロパティ」 79 ページ

ULBulkCopyColumnMappingCollection クラス

System.Collections.CollectionBase から継承した ULBulkCopyColumnMapping オブジェクトのコレクションです。このクラスは継承できません。

構文

Visual Basic
Public NotInheritable Class **ULBulkCopyColumnMappingCollection**
  Inherits CollectionBase

C#
public sealed class **ULBulkCopyColumnMappingCollection**: CollectionBase

備考

制限：ULBulkCopyColumnMappingCollection クラスは、.NET Compact Framework 2.0 では使用できません。

参照

- 「ULBulkCopyColumnMappingCollection メンバ」 81 ページ
- 「ULBulkCopyColumnMapping クラス」 74 ページ

ULBulkCopyColumnMappingCollection メンバ

パブリック・プロパティ

メンバ名	説明
Capacity (CollectionBase から継承)	CollectionBase に含めることのできる要素数を取得または設定します。
Count (CollectionBase から継承)	CollectionBase インスタンスに含まれている要素数を取得します。このプロパティは上書きできません。
「Item プロパティ」 82 ページ	指定されたインデックス位置の ULBulkCopyColumnMapping オブジェクトを取得します。

パブリック・メソッド

メンバ名	説明
「Add メソッド」 83 ページ	指定された ULBulkCopyColumnMapping をコレクションに追加します。

メンバ名	説明
Clear (CollectionBase から継承)	CollectionBase インスタンスからすべてのオブジェクトを削除します。このプロパティは上書きできません。
「Contains メソッド」 87 ページ	コレクション内に指定された ULBulkCopyColumnMapping オブジェクトが存在するかどうかを返します。
「CopyTo メソッド」 87 ページ	ULBulkCopyColumnMappingCollection の要素を、特定のインデックスを先頭に、ULBulkCopyColumnMapping 項目の配列にコピーします。
GetEnumerator (CollectionBase から継承)	CollectionBase インスタンスで反復処理する列挙子を返します。
「IndexOf メソッド」 88 ページ	コレクション内で指定された ULBulkCopyColumnMapping のインデックスを返します。
「Remove メソッド」 88 ページ	指定された ULBulkCopyColumnMapping 要素を ULBulkCopyColumnMappingCollection から削除します。
「RemoveAt メソッド」 89 ページ	指定されたインデックス位置のマッピングをコレクションから削除します。

参照

- 「ULBulkCopyColumnMappingCollection クラス」 81 ページ
- 「ULBulkCopyColumnMapping クラス」 74 ページ

Item プロパティ

指定されたインデックス位置の ULBulkCopyColumnMapping オブジェクトを取得します。

構文

Visual Basic

```
Public Readonly Property Item( _
   ByVal index As Integer _
) As ULBulkCopyColumnMapping
```

C#

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

パラメータ

- **index**　検索する ULBulkCopyColumnMapping オブジェクトの、0 から始まるインデックス。

プロパティ値

ULBulkCopyColumnMapping オブジェクトが返されます。

参照

- 「ULBulkCopyColumnMappingCollection クラス」 81 ページ
- 「ULBulkCopyColumnMappingCollection メンバ」 81 ページ
- 「ULBulkCopyColumnMapping クラス」 74 ページ

Add メソッド

指定された ULBulkCopyColumnMapping をコレクションに追加します。

Add(ULBulkCopyColumnMapping) メソッド

指定された ULBulkCopyColumnMapping をコレクションに追加します。

構文

Visual Basic

```
Public Function Add( _
  ByVal bulkCopyColumnMapping As ULBulkCopyColumnMapping _
) As ULBulkCopyColumnMapping
```

C#

```
public ULBulkCopyColumnMapping Add(
  ULBulkCopyColumnMapping bulkCopyColumnMapping
);
```

パラメータ

- **bulkCopyColumnMapping**　コレクションに追加される、マッピングを記述する ULBulkCopyColumnMapping オブジェクト。

備考

制限：ULBulkCopyColumnMappingCollection クラスは、.NET Compact Framework 2.0 では使用できません。

参照

- 「ULBulkCopyColumnMappingCollection クラス」 81 ページ
- 「ULBulkCopyColumnMappingCollection メンバ」 81 ページ
- 「Add メソッド」 83 ページ
- 「ULBulkCopyColumnMapping クラス」 74 ページ

Add(Int32, Int32) メソッド

ソース・カラムと送信先カラムの両方を指定する順序を使用して、新しい ULBulkCopyColumnMapping インスタンスを作成し、マッピングをコレクションに追加します。

構文

Visual Basic
```
Public Function Add( _
  ByVal sourceColumnOrdinal As Integer, _
  ByVal destinationColumnOrdinal As Integer _
) As ULBulkCopyColumnMapping
```

C#
```
public ULBulkCopyColumnMapping Add(
  int sourceColumnOrdinal,
  int destinationColumnOrdinal
);
```

パラメータ

- **sourceColumnOrdinal**　データ・ソース内のソース・カラムの順序位置。データ・ソースの先頭カラムの順序位置は 0 です。
- **destinationColumnOrdinal**　送信先テーブル内の送信先カラムの順序位置。テーブルの先頭カラムの順序位置は 0 です。

備考

制限：ULBulkCopyColumnMappingCollection クラスは、.NET Compact Framework 2.0 では使用できません。

参照

- 「ULBulkCopyColumnMappingCollection クラス」 81 ページ
- 「ULBulkCopyColumnMappingCollection メンバ」 81 ページ
- 「Add メソッド」 83 ページ
- 「ULBulkCopyColumnMapping クラス」 74 ページ

Add(Int32, String) メソッド

カラムの順序を使用してソース・カラムを参照し、カラム名を使用して送信先カラムを参照する、新しい ULBulkCopyColumnMapping を作成し、このマッピングをコレクションに追加します。

構文

Visual Basic
```
Public Function Add( _
  ByVal sourceColumnOrdinal As Integer, _
  ByVal destinationColumn As String _
) As ULBulkCopyColumnMapping
```

C#
```
public ULBulkCopyColumnMapping Add(
  int sourceColumnOrdinal,
  string  destinationColumn
);
```

パラメータ

- **sourceColumnOrdinal**　データ・ソース内のソース・カラムの順序位置。データ・ソースの先頭カラムの順序位置は 0 です。
- **destinationColumn**　送信先テーブル内の送信先カラムの名前。

備考

制限：ULBulkCopyColumnMappingCollection クラスは、.NET Compact Framework 2.0 では使用できません。

参照

- 「ULBulkCopyColumnMappingCollection クラス」 81 ページ
- 「ULBulkCopyColumnMappingCollection メンバ」 81 ページ
- 「Add メソッド」 83 ページ
- 「ULBulkCopyColumnMapping クラス」 74 ページ

Add(String, Int32) メソッド

カラム名を使用してソース・カラムを参照し、カラムの順序を使用して送信先カラムを参照する、新しい ULBulkCopyColumnMapping を作成し、このマッピングをコレクションに追加します。

カラムの順序または名前を使用してソース・カラムと送信先カラムを参照する、新しいカラム・マッピングを作成します。

構文

Visual Basic
```
Public Function Add( _
  ByVal sourceColumn As String, _
  ByVal destinationColumnOrdinal As Integer _
) As ULBulkCopyColumnMapping
```

C#
```
public ULBulkCopyColumnMapping Add(
  string  sourceColumn,
  int destinationColumnOrdinal
);
```

パラメータ

- **sourceColumn**　データ・ソース内のソース・カラムの名前。
- **destinationColumnOrdinal**　送信先テーブル内の送信先カラムの順序位置。テーブルの先頭カラムの順序位置は 0 です。

備考

制限：ULBulkCopyColumnMappingCollection クラスは、.NET Compact Framework 2.0 では使用できません。

参照

- 「ULBulkCopyColumnMappingCollection クラス」 81 ページ
- 「ULBulkCopyColumnMappingCollection メンバ」 81 ページ
- 「Add メソッド」 83 ページ
- 「ULBulkCopyColumnMapping クラス」 74 ページ

Add(String, String) メソッド

ソース・カラムと送信先カラムの両方を指定するカラム名を使用して、新しい ULBulkCopyColumnMapping を作成し、マッピングをコレクションに追加します。

構文

Visual Basic

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

C#

```
public ULBulkCopyColumnMapping Add(
   string  sourceColumn,
   string  destinationColumn
);
```

パラメータ

- **sourceColumn**　データ・ソース内のソース・カラムの名前。
- **destinationColumn**　送信先テーブル内の送信先カラムの名前。

備考

制限：ULBulkCopyColumnMappingCollection クラスは、.NET Compact Framework 2.0 では使用できません。

参照

- 「ULBulkCopyColumnMappingCollection クラス」 81 ページ
- 「ULBulkCopyColumnMappingCollection メンバ」 81 ページ
- 「Add メソッド」 83 ページ
- 「ULBulkCopyColumnMapping クラス」 74 ページ

Contains メソッド

コレクション内に指定された ULBulkCopyColumnMapping オブジェクトが存在するかどうかを返します。

構文

Visual Basic

```
Public Function Contains( _
  ByVal value As ULBulkCopyColumnMapping _
) As Boolean
```

C#

```
public bool Contains(
  ULBulkCopyColumnMapping value
);
```

パラメータ

- **value**　有効な ULBulkCopyColumnMapping オブジェクト。

戻り値

指定のマッピングがコレクション内にある場合は true、ない場合は false。

参照

- 「ULBulkCopyColumnMappingCollection クラス」 81 ページ
- 「ULBulkCopyColumnMappingCollection メンバ」 81 ページ
- 「ULBulkCopyColumnMapping クラス」 74 ページ

CopyTo メソッド

ULBulkCopyColumnMappingCollection の要素を、特定のインデックスを先頭に、ULBulkCopyColumnMapping 項目の配列にコピーします。

構文

Visual Basic

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

C#

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

パラメータ

- **array**　ULBulkCopyColumnMappingCollection の要素のコピー先である、1 次元の ULBulkCopyColumnMapping 配列。配列のインデックスは 0 から始まります。
- **index**　コピーが開始される、配列内の 0 から始まるインデックス。

参照

- 「ULBulkCopyColumnMappingCollection クラス」 81 ページ
- 「ULBulkCopyColumnMappingCollection メンバ」 81 ページ
- 「ULBulkCopyColumnMapping クラス」 74 ページ

IndexOf メソッド

コレクション内で指定された ULBulkCopyColumnMapping のインデックスを返します。

構文

Visual Basic

```
Public Function IndexOf( _
  ByVal value As ULBulkCopyColumnMapping _
) As Integer
```

C#

```
public int IndexOf(
  ULBulkCopyColumnMapping value
);
```

パラメータ

- **value**　検索対象の ULBulkCopyColumnMapping オブジェクト。

戻り値

カラム・マッピングの 0 から始まるインデックス、またはコレクション内にカラム・マッピングが見つからない場合は -1 が返されます。

参照

- 「ULBulkCopyColumnMappingCollection クラス」 81 ページ
- 「ULBulkCopyColumnMappingCollection メンバ」 81 ページ
- 「ULBulkCopyColumnMapping クラス」 74 ページ

Remove メソッド

指定された ULBulkCopyColumnMapping 要素を ULBulkCopyColumnMappingCollection から削除します。

構文

Visual Basic
```
Public Sub Remove( _
  ByVal value As ULBulkCopyColumnMapping _
)
```

C#
```
public void Remove(
  ULBulkCopyColumnMapping value
);
```

パラメータ

- **value**　コレクションから削除する ULBulkCopyColumnMapping オブジェクト。

参照

- 「ULBulkCopyColumnMappingCollection クラス」 81 ページ
- 「ULBulkCopyColumnMappingCollection メンバ」 81 ページ
- 「ULBulkCopyColumnMapping クラス」 74 ページ

RemoveAt メソッド

指定されたインデックス位置のマッピングをコレクションから削除します。

構文

Visual Basic
```
Public Sub RemoveAt( _
  ByVal index As Integer _
)
```

C#
```
public void RemoveAt(
  int index
);
```

パラメータ

- **index**　コレクションから削除する ULBulkCopyColumnMapping オブジェクトの、0 から始まるインデックス。

参照

- 「ULBulkCopyColumnMappingCollection クラス」 81 ページ
- 「ULBulkCopyColumnMappingCollection メンバ」 81 ページ

ULBulkCopyOptions 列挙体

ULBulkCopy クラスのインスタンスで使用する、1 つ以上のオプションを指定するビット単位フラグです。

構文

Visual Basic
Public Enum **ULBulkCopyOptions**

C#
public enum **ULBulkCopyOptions**

備考

ULBulkCopyOptions 列挙体は、ULBulkCopy インスタンスの構築時に使用され、WriteToServer メソッドの動作方法を指定します。

制限：ULBulkCopyOptions クラスは、.NET Compact Framework 2.0 では使用できません。

メンバ

メンバ名	説明	値
Default	これだけを指定すると、デフォルトの動作が使用されます。	0
KeepIdentity	これを指定すると、IDENTITY カラムにコピーされる元の値が保持されます。デフォルトでは、送信先テーブルでは新しい ID 番号が生成されます。	1
UseInternalTransaction	これを指定すると、バルク・コピー・オペレーションの各バッチがトランザクションの中で実行されます。これが指定されない場合、トランザクションは使用されません。このオプションを指定し、コンストラクタに ULTransaction オブジェクトも指定すると、System.ArgumentException が発生します。	2

参照

- 「ULBulkCopy クラス」 62 ページ

ULCommand クラス

IN パラメータあり、または IN パラメータなしで事前にコンパイルされた SQL 文またはクエリを表します。このオブジェクトを使用すると、文またはクエリを効率的に何回も実行できます。このクラスは継承できません。

構文

Visual Basic

```
Public NotInheritable Class ULCommand
  Inherits DbCommand
  Implements ICloneable
```

C#

```
public sealed class ULCommand: DbCommand,
  ICloneable
```

備考

ULCommand オブジェクトは、直接作成したり、ULConnection.CreateCommand メソッドを使用して作成したりできます。このメソッドによって、コマンドは特定の接続で文を実行するための適切なトランザクションを使用できます。

ULCommand.Transaction メソッドは、現在のトランザクションがコミットまたはロールバックされた後でリセットする必要があります。

ULCommand には、Ultra Light.NET データベースでコマンドを実行するための次のメソッドがあります。

メソッド	説明
ULCommand.ExecuteNonQuery	SQL INSERT 文、DELETE 文、UPDATE 文のように、結果セットを返さない文を実行します。
ULCommand.ExecuteReader()	SQL SELECT 文を実行し、ULDataReader で結果セットを返します。読み込み専用の結果セットの作成には、このメソッドを使用します。
ULCommand.ExecuteResultSet()	**UL 拡張：**SQL SELECT 文を実行し、ULResultSet で結果セットを返します。可変の結果セットの作成には、このメソッドを使用します。
ULCommand.ExecuteScalar	SQL SELECT 文を実行し、単一の値を返します。
ULCommand.ExecuteTable()	**UL 拡張：**直接の操作用に、ULTable にデータベース・テーブルを取り出します。ULCommand.CommandText はテーブルの名前として解釈され、ULCommand.IndexName はテーブルのソート順の指定に使用できます。ULCommand.CommandType は System.Data.CommandType.TableDirect であることが必要です。

ULCommand.CommandText を含め、ほとんどのプロパティをリセットできるほか、ULCommand オブジェクトを再利用できます。

リソース管理の理由により、コマンドを使用し終わったら、そのコマンドを明示的に破棄することをおすすめします。C# では、使用中の文を使って自動的に System.ComponentModel.Component.Dispose() メソッドを呼び出すことができます。または、明示的に System.ComponentModel.Component.Dispose() メソッドを呼び出すことができます。Visual Basic では、常に明示的に System.ComponentModel.Component.Dispose() メソッドを呼び出します。

継承：System.Data.Common.DbCommand

実装：System.Data.IDbCommand、System.IDisposable

参照

- 「ULCommand メンバ」 92 ページ
- 「CreateCommand メソッド」 160 ページ
- 「Transaction プロパティ」 104 ページ
- 「ExecuteNonQuery メソッド」 118 ページ
- 「ExecuteReader() メソッド」 119 ページ
- 「ExecuteResultSet() メソッド」 122 ページ
- 「ExecuteScalar メソッド」 124 ページ
- 「ExecuteTable() メソッド」 125 ページ
- 「ULTable クラス」 548 ページ
- 「CommandText プロパティ」 99 ページ
- CommandType.TableDirect
- Component.Dispose
- 「CommandType プロパティ」 100 ページ
- 「IndexName プロパティ」 102 ページ
- DbCommand
- IDbCommand
- IDisposable
- 「ULResultSet クラス」 415 ページ
- 「ULDataReader クラス」 276 ページ

ULCommand メンバ

パブリック・コンストラクタ

メンバ名	説明
「ULCommand コンストラクタ」 95 ページ	「ULCommand クラス」 91 ページの新しいインスタンスを初期化します。

パブリック・プロパティ

メンバ名	説明
「CommandText プロパティ」 99 ページ	ULCommand.CommandType が System.Data.CommandType.TableDirect である場合の、SQL 文のテキストまたはテーブルの名前を指定します。パラメータ化された文の場合、疑問符 (?) プレースホルダを使用してパラメータを渡します。
「CommandTimeout プロパティ」 100 ページ	この機能は Ultra Light.NET ではサポートされていません。
「CommandType プロパティ」 100 ページ	実行されるコマンドのタイプを指定します。
「Connection プロパティ」 101 ページ	ULCommand オブジェクトを実行する接続オブジェクトです。
「DesignTimeVisible プロパティ」 102 ページ	カスタマイズされた Windows Form Designer 制御で ULCommand を参照できるようにするかどうかを指定します。
「IndexName プロパティ」 102 ページ	**UL 拡張：** ULCommand.CommandType が System.Data.CommandType.TableDirect であるときに、テーブルを開く (ソートする) インデックスの名前を指定します。
「Parameters プロパティ」 103 ページ	現在の文のパラメータを指定します。
「Plan プロパティ」 104 ページ	**UL 拡張：** Ultra Light.NET がクエリの実行に使用するアクセス・プランを返します。このプロパティは、主に開発中の使用を目的とします。
「Transaction プロパティ」 104 ページ	ULCommand が実行される ULTransaction を指定します。
「UpdatedRowSource プロパティ」 105 ページ	ULDataAdapter の Update メソッドによって使用されるときにコマンドの結果が DataRow に適用される方法を指定します。

パブリック・メソッド

メンバ名	説明
「BeginExecuteNonQuery メソッド」 105 ページ	コールバック・プロシージャと状態情報を指定して、この ULCommand で記述される SQL 文の非同期実行を開始します。

メンバ名	説明
「BeginExecuteReader メソッド」 107 ページ	この ULCommand で記述される SQL 文の非同期実行を開始し、結果セットを取得します。
「Cancel メソッド」 110 ページ	このメソッドは Ultra Light.NET ではサポートされていません。
「CreateParameter メソッド」 111 ページ	ULCommand オブジェクトにパラメータを指定するために ULParameter オブジェクトを提供します。
「EndExecuteNonQuery メソッド」 111 ページ	SQL 文の非同期実行を終了します。
「EndExecuteReader メソッド」 115 ページ	SQL 文の非同期実行を終了し、要求された ULDataReader を返します。
「ExecuteNonQuery メソッド」 118 ページ	SQL INSERT 文、DELETE 文、UPDATE 文のように、結果セットを返さない文を実行します。
「ExecuteReader メソッド」 118 ページ	SQL SELECT 文を実行し、結果セットを返します。
「ExecuteResultSet メソッド」 121 ページ	**UL 拡張：**SQL SELECT 文を実行し、ULResultSet として結果セットを返します。
「ExecuteScalar メソッド」 124 ページ	SQL SELECT 文を実行し、単一の値を返します。
「ExecuteTable メソッド」 125 ページ	**UL 拡張：**直接の操作用に、ULTable にデータベース・テーブルを取り出します。ULCommand.CommandText はテーブルの名前として解釈され、ULCommand.IndexName はテーブルのソート順の指定に使用できます。
「Prepare メソッド」 127 ページ	このコマンドの SQL 文を事前にコンパイルして格納します。

参照

- 「ULCommand クラス」 91 ページ
- 「CreateCommand メソッド」 160 ページ
- 「Transaction プロパティ」 104 ページ
- 「ExecuteNonQuery メソッド」 118 ページ
- 「ExecuteReader() メソッド」 119 ページ
- 「ExecuteResultSet() メソッド」 122 ページ
- 「ExecuteScalar メソッド」 124 ページ
- 「ExecuteTable() メソッド」 125 ページ
- 「ULTable クラス」 548 ページ
- 「CommandText プロパティ」 99 ページ
- CommandType.TableDirect
- Component.Dispose
- 「CommandType プロパティ」 100 ページ
- 「IndexName プロパティ」 102 ページ
- DbCommand
- IDbCommand
- IDisposable
- 「ULResultSet クラス」 415 ページ
- 「ULDataReader クラス」 276 ページ

ULCommand コンストラクタ

「ULCommand クラス」 91 ページの新しいインスタンスを初期化します。

ULCommand() コンストラクタ

ULCommand オブジェクトを初期化します。

構文

Visual Basic
Public Sub **New()**

C#
public **ULCommand();**

備考

ULCommand オブジェクトに ULCommand.CommandText、ULCommand.Connection、ULCommand.Transaction のプロパティが設定されていないと、文を実行できません。

参照

- 「ULCommand クラス」 91 ページ
- 「ULCommand メンバ」 92 ページ
- 「ULCommand コンストラクタ」 95 ページ
- 「CreateCommand メソッド」 160 ページ
- 「ULCommand(String) コンストラクタ」 96 ページ
- 「ULCommand(String, ULConnection) コンストラクタ」 97 ページ
- 「ULCommand(String, ULConnection, ULTransaction) コンストラクタ」 98 ページ
- 「CommandText プロパティ」 99 ページ
- 「Connection プロパティ」 101 ページ
- 「Transaction プロパティ」 104 ページ

ULCommand(String) コンストラクタ

ULCommand オブジェクトを、指定されたコマンド・テキストで初期化します。

構文

Visual Basic

```
Public Sub New( _
  ByVal cmdText As String _
)
```

C#

```
public  ULCommand(
  string  cmdText
);
```

パラメータ

- **cmdText**　ULCommand.CommandType が System.Data.CommandType.TableDirect である場合の、SQL 文のテキストまたはテーブルの名前。パラメータ化された文の場合、疑問符 (?) プレースホルダを使用してパラメータを渡します。

備考

ULCommand オブジェクトに ULCommand.Connection と ULCommand.Transaction が設定されていないと、文を実行できません。

参照

- 「ULCommand クラス」 91 ページ
- 「ULCommand メンバ」 92 ページ
- 「ULCommand コンストラクタ」 95 ページ
- 「CreateCommand メソッド」 160 ページ
- 「ULCommand() コンストラクタ」 95 ページ
- 「ULCommand(String, ULConnection) コンストラクタ」 97 ページ
- 「ULCommand(String, ULConnection, ULTransaction) コンストラクタ」 98 ページ
- 「Connection プロパティ」 101 ページ
- 「Transaction プロパティ」 104 ページ
- 「CommandType プロパティ」 100 ページ
- CommandType.TableDirect

ULCommand(String, ULConnection) コンストラクタ

ULCommand オブジェクトを、指定されたコマンド・テキストと接続で初期化します。

構文

Visual Basic

```
Public Sub New( _
  ByVal cmdText As String, _
  ByVal connection As ULConnection _
)
```

C#

```
public  ULCommand(
  string  cmdText,
  ULConnection connection
);
```

パラメータ

- **cmdText**　ULCommand.CommandType が System.Data.CommandType.TableDirect である場合の、SQL 文のテキストまたはテーブルの名前。パラメータ化された文の場合、疑問符 (?) プレースホルダを使用してパラメータを渡します。
- **connection**　現在の接続を表す ULConnection オブジェクト。

備考

ULCommand オブジェクトに ULCommand.Transaction が設定されていないと、文を実行できないことがあります。

参照

- 「ULCommand クラス」 91 ページ
- 「ULCommand メンバ」 92 ページ
- 「ULCommand コンストラクタ」 95 ページ
- 「CreateCommand メソッド」 160 ページ
- 「ULCommand() コンストラクタ」 95 ページ
- 「ULCommand(String) コンストラクタ」 96 ページ
- 「ULCommand(String, ULConnection, ULTransaction) コンストラクタ」 98 ページ
- 「Transaction プロパティ」 104 ページ
- 「CommandType プロパティ」 100 ページ
- CommandType.TableDirect
- 「ULConnection クラス」 139 ページ

ULCommand(String, ULConnection, ULTransaction) コンストラクタ

ULCommand オブジェクトを、指定されたコマンド・テキスト、接続、トランザクションで初期化します。

構文

Visual Basic

```
Public Sub New( _
  ByVal cmdText As String, _
  ByVal connection As ULConnection, _
  ByVal transaction As ULTransaction _
)
```

C#

```
public  ULCommand(
  string  cmdText,
  ULConnection connection,
  ULTransaction transaction
);
```

パラメータ

- **cmdText**　ULCommand.CommandType が System.Data.CommandType.TableDirect である場合の、SQL 文のテキストまたはテーブルの名前。パラメータ化された文の場合、疑問符 (?) プレースホルダを使用してパラメータを渡します。
- **connection**　現在の接続を表す ULConnection オブジェクト。
- **transaction**　ULCommand が実行される ULTransaction。

参照

- 「ULCommand クラス」 91 ページ
- 「ULCommand メンバ」 92 ページ
- 「ULCommand コンストラクタ」 95 ページ
- 「CreateCommand メソッド」 160 ページ
- 「ULCommand() コンストラクタ」 95 ページ
- 「ULCommand(String) コンストラクタ」 96 ページ
- 「ULCommand(String, ULConnection) コンストラクタ」 97 ページ
- 「CommandType プロパティ」 100 ページ
- CommandType.TableDirect
- 「ULTransaction クラス」 586 ページ

CommandText プロパティ

ULCommand.CommandType が System.Data.CommandType.TableDirect である場合の、SQL 文のテキストまたはテーブルの名前を指定します。パラメータ化された文の場合、疑問符 (?) プレースホルダを使用してパラメータを渡します。

構文

Visual Basic
Public Overrides Property **CommandText** As String

C#
public override string **CommandText** { get; set; }

プロパティ値

SQL 文のテキストまたはテーブルの名前を指定する文字列。デフォルトは、空の文字列 (無効なコマンド) です。

備考

SELECT 文は、パフォーマンス上の理由により、デフォルトで読み込み専用とマーク付けされます。クエリを使用して更新を行う場合、SELECT 文は "FOR UPDATE" で終わる必要があります。

例

次の例は、パラメータ化されたプレースホルダの使用方法を示します。

```
' Visual Basic
myCmd.CommandText = "SELECT * FROM Customers WHERE CustomerID = ?"

// C#
myCmd.CommandText = "SELECT * FROM Customers WHERE CustomerID = ?";
```

参照

- 「ULCommand クラス」 91 ページ
- 「ULCommand メンバ」 92 ページ
- 「ExecuteNonQuery メソッド」 118 ページ
- 「ExecuteReader() メソッド」 119 ページ
- 「ExecuteResultSet() メソッド」 122 ページ
- 「ExecuteScalar メソッド」 124 ページ
- 「ExecuteTable() メソッド」 125 ページ
- 「CommandType プロパティ」 100 ページ
- CommandType.TableDirect

CommandTimeout プロパティ

この機能は Ultra Light.NET ではサポートされていません。

構文

Visual Basic
Public Overrides Property **CommandTimeout** As Integer

C#
public override int **CommandTimeout** { get; set; }

プロパティ値

値は常に 0 です。

参照

- 「ULCommand クラス」 91 ページ
- 「ULCommand メンバ」 92 ページ

CommandType プロパティ

実行されるコマンドのタイプを指定します。

構文

Visual Basic
Public Overrides Property **CommandType** As CommandType

C#
public override CommandType **CommandType** { get; set; }

プロパティ値

System.Data.CommandType 値の 1 つ。デフォルトは System.Data.CommandType.Text です。

備考

サポートされているコマンド・タイプは、次のとおりです。

- System.Data.CommandType.TableDirect - **UL 拡張：** この CommandType を指定するときは、ULCommand.CommandText はデータベース・テーブルの名前でなければなりません。また、ULCommand.IndexName を使用して、テーブルを開く (ソートする) インデックスを指定することもできます。テーブルにアクセスするには、ULCommand.ExecuteTable() または ULCommand.ExecuteReader() を使用します。
- System.Data.CommandType.Text - この CommandType を指定するときは、ULCommand.CommandText は SQL 文またはクエリでなければなりません。クエリ以外の SQL 文を実行するには、ULCommand.ExecuteNonQuery を使用します。また、クエリを実行するには、ULCommand.ExecuteReader() または ULCommand.ExecuteScalar を使用します。

参照

- 「ULCommand クラス」 91 ページ
- 「ULCommand メンバ」 92 ページ
- CommandType
- CommandType.Text
- CommandType.TableDirect
- 「CommandText プロパティ」 99 ページ
- 「IndexName プロパティ」 102 ページ
- 「ExecuteTable() メソッド」 125 ページ
- 「ExecuteReader() メソッド」 119 ページ
- CommandType.Text
- 「CommandText プロパティ」 99 ページ
- 「ExecuteNonQuery メソッド」 118 ページ
- 「ExecuteReader() メソッド」 119 ページ
- 「ExecuteScalar メソッド」 124 ページ

Connection プロパティ

ULCommand オブジェクトを実行する接続オブジェクトです。

構文

Visual Basic
Public Property **Connection** As ULConnection

C#
public ULConnection **Connection** { get; set; }

プロパティ値

コマンドを実行する ULConnection オブジェクト。

備考

ULCommand オブジェクトは、開かれた接続がないと、実行できません。

デフォルト値は NULL 参照 (Visual Basic の Nothing) です。

これは、System.Data.IDbCommand.Connection と System.Data.Common.DbCommand.Connection が厳密に型指定されたものです。

参照

- 「ULCommand クラス」 91 ページ
- 「ULCommand メンバ」 92 ページ
- 「ULConnection クラス」 139 ページ
- IDbCommand.Connection
- DbCommand.Connection

DesignTimeVisible プロパティ

カスタマイズされた Windows Form Designer 制御で ULCommand を参照できるようにするかどうかを指定します。

構文

Visual Basic
Public Overrides Property **DesignTimeVisible** As Boolean

C#
public override bool **DesignTimeVisible** { get; set; }

プロパティ値

ULCommand インスタンスを参照できるようにする場合は true、このインスタンスを参照できないようにする場合は false。デフォルトは false です。

参照

- 「ULCommand クラス」 91 ページ
- 「ULCommand メンバ」 92 ページ

IndexName プロパティ

UL 拡張：ULCommand.CommandType が System.Data.CommandType.TableDirect であるときに、テーブルを開く (ソートする) インデックスの名前を指定します。

構文

Visual Basic
Public Property **IndexName** As String

C#
public string **IndexName** { get; set; }

プロパティ値

インデックスの名前を指定する文字列。デフォルトは NULL 参照 (Visual Basic の Nothing) です。この場合、テーブルはプライマリ・キーを使用して開かれます。

参照

- 「ULCommand クラス」 91 ページ
- 「ULCommand メンバ」 92 ページ
- 「ExecuteTable() メソッド」 125 ページ
- 「ExecuteReader() メソッド」 119 ページ
- 「CommandType プロパティ」 100 ページ
- CommandType.TableDirect

Parameters プロパティ

現在の文のパラメータを指定します。

構文

Visual Basic
Public Readonly Property **Parameters** As ULParameterCollection

C#
public ULParameterCollection **Parameters** { get;}

プロパティ値

SQL 文のパラメータを保持する ULParameterCollection。デフォルト値は空のコレクションです。

備考

ULCommand.CommandText で疑問符を使用してパラメータを示します。コレクション内のパラメータは、疑問符のプレースホルダと同じ順序で指定されます。たとえば、コレクション内の最初のパラメータは、SQL 文の最初の疑問符に対応し、コレクション内の 2 番目のパラメータは、SQL 文の 2 番目の疑問符に対応します。ULCommand.CommandText 内の疑問符の数は、少なくともコレクション内のパラメータの数と同じでなければなりません。

これは、System.Data.IDbCommand.Parameters と System.Data.Common.DbCommand.Parameters が厳密に型指定されたものです。

参照

- 「ULCommand クラス」 91 ページ
- 「ULCommand メンバ」 92 ページ
- 「ULParameterCollection クラス」 392 ページ
- 「ULParameter クラス」 375 ページ
- IDbCommand.Connection
- DbCommand.Connection
- 「CommandText プロパティ」 99 ページ

Plan プロパティ

UL 拡張：Ultra Light.NET がクエリの実行に使用するアクセス・プランを返します。このプロパティは、主に開発中の使用を目的とします。

構文

Visual Basic
Public Readonly Property **Plan** As String

C#
public string **Plan** { get;}

プロパティ値

クエリ実行プランのテキストベースの記述が含まれる文字列。

参照

- 「ULCommand クラス」 91 ページ
- 「ULCommand メンバ」 92 ページ

Transaction プロパティ

ULCommand が実行される ULTransaction を指定します。

構文

Visual Basic
Public Property **Transaction** As ULTransaction

C#
public ULTransaction **Transaction** { get; set; }

プロパティ値

ULCommand が実行される ULTransaction。これは、ULCommand.Connection によって指定された接続の現在のトランザクションでなければなりません。デフォルトは NULL 参照 (Visual Basic の Nothing) です。

備考

トランザクションがコミットまたはロールバックされた後でコマンドを再使用する場合は、このプロパティをリセットする必要があります。

これは、System.Data.IDbCommand.Transaction と System.Data.Common.DbCommand.Transaction が厳密に型指定されたものです。

参照

- 「ULCommand クラス」 91 ページ
- 「ULCommand メンバ」 92 ページ
- 「BeginTransaction() メソッド」 153 ページ
- 「ULTransaction クラス」 586 ページ
- 「Connection プロパティ」 101 ページ
- IDbCommand.Transaction
- DbCommand.Transaction

UpdatedRowSource プロパティ

ULDataAdapter の Update メソッドによって使用されるときにコマンドの結果が DataRow に適用される方法を指定します。

構文

Visual Basic
Public Overrides Property **UpdatedRowSource** As UpdateRowSource

C#
public override UpdateRowSource **UpdatedRowSource** { get; set; }

プロパティ値

System.Data.UpdateRowSource 値の 1 つ。デフォルト値は System.Data.UpdateRowSource.Both です。

参照

- 「ULCommand クラス」 91 ページ
- 「ULCommand メンバ」 92 ページ
- UpdateRowSource
- UpdateRowSource.Both

BeginExecuteNonQuery メソッド

コールバック・プロシージャと状態情報を指定して、この ULCommand で記述される SQL 文の非同期実行を開始します。

BeginExecuteNonQuery() メソッド

構文

Visual Basic
Public Function **BeginExecuteNonQuery()** As IAsyncResult

C#
public IAsyncResult **BeginExecuteNonQuery();**

参照

- 「ULCommand クラス」 91 ページ
- 「ULCommand メンバ」 92 ページ
- 「BeginExecuteNonQuery メソッド」 105 ページ

BeginExecuteNonQuery(AsyncCallback, Object) メソッド

コールバック・プロシージャと状態情報を指定して、この ULCommand で記述される SQL 文の非同期実行を開始します。

構文

Visual Basic

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

C#

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

パラメータ

- **callback**　コマンドの実行が終了すると起動される System.AsyncCallback デリゲート。コールバックが必要ないことを示すには、NULL (Microsoft Visual Basic の場合は Nothing) を渡します。
- **stateObject**　コールバック・プロシージャに渡される、ユーザ定義のステータス・オブジェクト。コールバック・プロシージャからこのオブジェクトを取得するには、System.IAsyncResult.AsyncState プロパティを使用します。

戻り値

ポーリング、結果の待機、または両方に使用できる System.IAsyncResult が返されます。この値は、影響を受けたローの数を返す EndExecuteNonQuery(IAsyncResult) を起動する場合にも必要です。

参照

- 「ULCommand クラス」 91 ページ
- 「ULCommand メンバ」 92 ページ
- 「BeginExecuteNonQuery メソッド」 105 ページ
- 「EndExecuteNonQuery メソッド」 111 ページ
- IAsyncResult.AsyncState
- AsyncCallback
- IAsyncResult

BeginExecuteReader メソッド

この ULCommand で記述される SQL 文の非同期実行を開始し、結果セットを取得します。

BeginExecuteReader() メソッド

この ULCommand で記述される SQL 文の非同期実行を開始し、結果セットを取得します。

構文

Visual Basic
Public Function **BeginExecuteReader()** As IAsyncResult

C#
public IAsyncResult **BeginExecuteReader();**

戻り値

ポーリング、結果の待機、または両方に使用できる System.IAsyncResult が返されます。この値は、返されたローを取得するために使用する ULDataReader インスタンスを返す、EndExecuteReader(IAsyncResult) を起動する場合にも必要です。

参照

- 「ULCommand クラス」 91 ページ
- 「ULCommand メンバ」 92 ページ
- 「BeginExecuteReader メソッド」 107 ページ
- 「EndExecuteReader メソッド」 115 ページ
- IAsyncResult
- 「ULDataReader クラス」 276 ページ

BeginExecuteReader(CommandBehavior) メソッド

CommandBehavior 値の 1 つを使用してこの ULCommand で記述される SQL 文の非同期実行を開始し、結果セットを取得します。

構文

Visual Basic
```
Public Function BeginExecuteReader( _
  ByVal cmdBehavior As CommandBehavior _
) As IAsyncResult
```

C#
```
public IAsyncResult BeginExecuteReader(
  CommandBehavior cmdBehavior
);
```

パラメータ

- **cmdBehavior**　クエリの結果の記述と、接続への影響の記述の System.Data.CommandBehavior フラグのビット単位の組み合わせ。Ultra Light.NET で有効なのは、System.Data.CommandBehavior.Default フラグ、System.Data.CommandBehavior.CloseConnection フラグ、System.Data.CommandBehavior.SchemaOnly フラグだけです。

戻り値

ポーリング、結果の待機、または両方に使用できる System.IAsyncResult が返されます。この値は、返されたローを取得するために使用する ULDataReader インスタンスを返す、EndExecuteReader(IAsyncResult) を起動する場合にも必要です。

参照

- 「ULCommand クラス」 91 ページ
- 「ULCommand メンバ」 92 ページ
- 「BeginExecuteReader メソッド」 107 ページ
- 「EndExecuteReader メソッド」 115 ページ
- CommandBehavior
- CommandBehavior.Default
- CommandBehavior.CloseConnection
- CommandBehavior.SchemaOnly
- IAsyncResult
- 「EndExecuteReader メソッド」 115 ページ
- 「ULDataReader クラス」 276 ページ

BeginExecuteReader(AsyncCallback, Object) メソッド

コールバック・プロシージャと状態情報を指定して、この ULCommand で記述される SQL 文の非同期実行を開始し、結果セットを取得します。

構文

Visual Basic

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

C#

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

パラメータ

- **callback**　コマンドの実行が終了すると起動される System.AsyncCallback デリゲート。コールバックが必要ないことを示すには、NULL (Microsoft Visual Basic の場合は Nothing) を渡します。

- **stateObject**　コールバック・プロシージャに渡される、ユーザ定義のステータス・オブジェクト。コールバック・プロシージャからこのオブジェクトを取得するには、System.IAsyncResult.AsyncState プロパティを使用します。

戻り値

ポーリング、結果の待機、または両方に使用できる System.IAsyncResult が返されます。この値は、返されたローを取得するために使用する ULDataReader インスタンスを返す、EndExecuteReader(IAsyncResult) を起動する場合にも必要です。

参照

- 「ULCommand クラス」 91 ページ
- 「ULCommand メンバ」 92 ページ
- 「BeginExecuteReader メソッド」 107 ページ
- 「EndExecuteReader メソッド」 115 ページ
- AsyncCallback
- IAsyncResult.AsyncState
- IAsyncResult.AsyncState
- 「EndExecuteReader メソッド」 115 ページ
- IAsyncResult
- 「ULDataReader クラス」 276 ページ

BeginExecuteReader(AsyncCallback, Object, CommandBehavior) メソッド

コールバック・プロシージャと状態情報を指定して、CommandBehavior 値の 1 つを使用してこの ULCommand で記述される SQL 文の非同期実行を開始し、結果セットを取得します。

構文

Visual Basic

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

C#

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

パラメータ

- **callback**　コマンドの実行が終了すると起動される System.AsyncCallback デリゲート。コールバックが必要ないことを示すには、NULL (Microsoft Visual Basic の場合は Nothing) を渡します。

- **stateObject**　コールバック・プロシージャに渡される、ユーザ定義のステータス・オブジェクト。コールバック・プロシージャからこのオブジェクトを取得するには、System.IAsyncResult.AsyncState プロパティを使用します。
- **cmdBehavior**　クエリの結果の記述と、接続への影響の記述の System.Data.CommandBehavior フラグのビット単位の組み合わせ。Ultra Light.NET で有効なのは、System.Data.CommandBehavior.Default フラグ、System.Data.CommandBehavior.CloseConnection フラグ、System.Data.CommandBehavior.SchemaOnly フラグだけです。

戻り値

ポーリング、結果の待機、または両方に使用できる System.IAsyncResult が返されます。この値は、返されたローを取得するために使用する ULDataReader インスタンスを返す、EndExecuteReader(IAsyncResult) を起動する場合にも必要です。

参照

- 「ULCommand クラス」 91 ページ
- 「ULCommand メンバ」 92 ページ
- 「BeginExecuteReader メソッド」 107 ページ
- AsyncCallback
- IAsyncResult.AsyncState
- CommandBehavior
- CommandBehavior.Default
- CommandBehavior.CloseConnection
- CommandBehavior.SchemaOnly
- IAsyncResult
- 「EndExecuteReader メソッド」 115 ページ
- 「ULDataReader クラス」 276 ページ

Cancel メソッド

このメソッドは Ultra Light.NET ではサポートされていません。

構文

Visual Basic
Public Overrides Sub **Cancel()**

C#
public override void **Cancel();**

備考

このメソッドによって行われる処理はありません。Ultra Light.NET コマンドは、実行中に中断できません。

参照

- 「ULCommand クラス」 91 ページ
- 「ULCommand メンバ」 92 ページ

CreateParameter メソッド

ULCommand オブジェクトにパラメータを指定するために ULParameter オブジェクトを提供します。

構文

Visual Basic
Public Function **CreateParameter()** As ULParameter

C#
public ULParameter **CreateParameter();**

戻り値

ULParameter オブジェクトとして返される新しいパラメータ。

備考

一部の SQL 文では、疑問符 (?) によって文のテキストに示されているパラメータを使用できます。CreateParameter メソッドは、ULParameter オブジェクトを提供します。ULParameter にプロパティを設定して、そのパラメータの値を指定できます。

これは、System.Data.IDbCommand.CreateParameter と System.Data.Common.DbCommand.CreateParameter が厳密に型指定されたものです。

参照

- 「ULCommand クラス」 91 ページ
- 「ULCommand メンバ」 92 ページ
- 「ULParameter クラス」 375 ページ
- IDbCommand.CreateParameter
- DbCommand.CreateParameter

EndExecuteNonQuery メソッド

SQL 文の非同期実行を終了します。

構文

Visual Basic
Public Function **EndExecuteNonQuery(_**
  ByVal *asyncResult* As IAsyncResult _
) As Integer

```
C#
public int EndExecuteNonQuery(
  IAsyncResult asyncResult
);
```

パラメータ

- **asyncResult** BeginExecuteNonQuery への呼び出しによって返される System.IAsyncResult。

戻り値

影響を受けたローの数 (ExecuteNonQuery と同じ動作)。

備考

BeginExecuteNonQuery を呼び出すごとに、EndExecuteNonQuery を呼び出す必要があります。呼び出しは、BeginExecuteNonQuery が返されてから行います。ADO.NET はスレッドに対応していないため、各自で BeginExecuteNonQuery が返されたことを確認する必要があります。EndExecuteNonQuery に渡される System.IAsyncResult は、完了した BeginExecuteNonQuery 呼び出しから返されるものと同じでなければなりません。EndExecuteNonQuery を呼び出して、BeginExecuteReader への呼び出しを終了すると、エラーになります。逆についても同様です。

コマンドの実行中にエラーが発生すると、EndExecuteNonQuery が呼び出されるときに例外がスローされます。

実行の完了を待機するには、4 通りの方法があります。

Call EndExecuteNonQuery EndExecuteNonQuery を呼び出すと、コマンドが完了するまでブロックします。次に例を示します。

```
' Visual Basic
Dim cmd As ULCommand = new ULCommand( _
    "UPDATE Departments" _
    + " SET DepartmentName = 'Engineering'" _
    + " WHERE DepartmentID=100", _
    conn _
  )
Dim res As IAsyncResult res = _
  cmd.BeginExecuteNonQuery()
' perform other work
' this will block until the command completes
Dim rowCount As Integer = _
  cmd.EndExecuteNonQuery( res )

// C#
ULCommand cmd = new ULCommand(
    "UPDATE Departments"
    + " SET DepartmentName = 'Engineering'"
    + " WHERE DepartmentID=100",
    conn
  );
IAsyncResult res = cmd.BeginExecuteNonQuery();
// perform other work
// this will block until the command completes
int rowCount = cmd.EndExecuteNonQuery( res );
```

Poll the IsCompleted property of the IAsyncResult IAsyncResult の IsCompleted プロパティをポーリングできます。次に例を示します。


```
' Visual Basic
Dim cmd As ULCommand = new ULCommand( _
    "UPDATE Departments" _
    + " SET DepartmentName = 'Engineering'" _
    + " WHERE DepartmentID=100", _
    conn _
  )
Dim res As IAsyncResult res = _
  cmd.BeginExecuteNonQuery()
While( !res.IsCompleted )
  ' do other work
End While
' this will block until the command completes
Dim rowCount As Integer = _
  cmd.EndExecuteNonQuery( res )

// C#
ULCommand cmd = new ULCommand(
    "UPDATE Departments"
    + " SET DepartmentName = 'Engineering'"
    + " WHERE DepartmentID=100",
    conn
  );
IAsyncResult res = cmd.BeginExecuteNonQuery();
while( !res.IsCompleted ) {
  // do other work
}
// this will block until the command completes
int rowCount = cmd.EndExecuteNonQuery( res );
```

Use the IAsyncResult.AsyncWaitHandle property to get a synchronization object
IAsyncResult.AsyncWaitHandle プロパティを使用して同期オブジェクトを取得し、その状態で待機できます。次に例を示します。

```
' Visual Basic
Dim cmd As ULCommand = new ULCommand( _
    "UPDATE Departments" _
    + " SET DepartmentName = 'Engineering'" _
    + " WHERE DepartmentID=100", _
    conn _
  )
Dim res As IAsyncResult res = _
  cmd.BeginExecuteNonQuery()
' perform other work
Dim wh As WaitHandle = res.AsyncWaitHandle
wh.WaitOne()
' this will not block because the command is finished
Dim rowCount As Integer = _
  cmd.EndExecuteNonQuery( res )

// C#
ULCommand cmd = new ULCommand(
    "UPDATE Departments"
    + " SET DepartmentName = 'Engineering'"
    + " WHERE DepartmentID=100",
    conn
  );
IAsyncResult res = cmd.BeginExecuteNonQuery();
// perform other work
WaitHandle wh = res.AsyncWaitHandle;
wh.WaitOne();
```

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

Specify a callback function when calling BeginExecuteNonQuery BeginExecuteNonQuery の呼び出し時にコールバック関数を指定できます。次に例を示します。

```
' Visual Basic
Private Sub callbackFunction(ByVal ar As IAsyncResult)
  Dim cmd As ULCommand = _
    CType(ar.AsyncState, ULCommand)
  ' this won't block since the command has completed
  Dim rowCount As Integer = _
    cmd.EndExecuteNonQuery( res )
End Sub


' elsewhere in the code
Private Sub DoStuff()
  Dim cmd As ULCommand = new ULCommand( _
      "UPDATE Departments" _
      + " SET DepartmentName = 'Engineering'" _
      + " WHERE DepartmentID=100", _
      conn _
    )
  Dim res As IAsyncResult = _
    cmd.BeginExecuteNonQuery( _
       callbackFunction, cmd _
      )
  ' perform other work.  The callback function
  ' will be called when the command completes
End Sub


// C#
private void callbackFunction( IAsyncResult ar )
{
  ULCommand cmd = (ULCommand) ar.AsyncState;
  // this won't block since the command has completed
  int rowCount = cmd.EndExecuteNonQuery();
}


// elsewhere in the code
private void DoStuff()
{
  ULCommand cmd = new ULCommand(
      "UPDATE Departments"
      + " SET DepartmentName = 'Engineering'"
      + " WHERE DepartmentID=100",
      conn
    );
  IAsyncResult res = cmd.BeginExecuteNonQuery(
      callbackFunction, cmd
    );
  // perform other work.  The callback function
  // will be called when the command completes
}
```

コールバック関数は別のスレッドで実行するため、スレッド化されたプログラム内でのユーザ・インタフェースの更新に関する通常の注意が適用されます。

参照

- 「ULCommand クラス」 91 ページ
- 「ULCommand メンバ」 92 ページ
- 「BeginExecuteNonQuery() メソッド」 105 ページ
- IAsyncResult

EndExecuteReader メソッド

SQL 文の非同期実行を終了し、要求された ULDataReader を返します。

構文

Visual Basic

```
Public Function EndExecuteReader( _
  ByVal asyncResult As IAsyncResult _
) As ULDataReader
```

C#

```
public ULDataReader EndExecuteReader(
  IAsyncResult asyncResult
);
```

パラメータ

- **asyncResult**　BeginExecuteReader への呼び出しによって返される System.IAsyncResult。

戻り値

要求されたローの取り出しに使用する ULDataReader オブジェクト (ExecuteReader と同じ動作)。

備考

BeginExecuteReader を呼び出すごとに、EndExecuteReader を呼び出す必要があります。呼び出しは、BeginExecuteReader が返されてから行います。ADO.NET はスレッドに対応していないため、各自で BeginExecuteReader が返されたことを確認する必要があります。EndExecuteReader に渡される System.IAsyncResult は、完了した BeginExecuteReader 呼び出しから返されるものと同じでなければなりません。EndExecuteReade を呼び出して、BeginExecuteNonQuery への呼び出しを終了すると、エラーになります。逆についても同様です。

コマンドの実行中にエラーが発生すると、EndExecuteReader が呼び出されるときに例外がスローされます。

実行の完了を待機するには、4 通りの方法があります。

Call EndExecuteReader EndExecuteReader を呼び出すと、コマンドが完了するまでブロックします。次に例を示します。

```
' Visual Basic
Dim cmd As ULCommand = new ULCommand( _
   "SELECT * FROM Departments", conn _
  )
Dim res As IAsyncResult res = _
  cmd.BeginExecuteReader()
```

```
' perform other work
' this will block until the command completes
Dim reader As ULDataReader = _
  cmd.EndExecuteReader( res )

// C#
ULCommand cmd = new ULCommand(
    "SELECT * FROM Departments", conn
  );
IAsyncResult res = cmd.BeginExecuteReader();
// perform other work
// this will block until the command completes
ULDataReader reader = cmd.EndExecuteReader( res );
```

Poll the IsCompleted property of the IAsyncResult IAsyncResult の IsCompleted プロパティをポーリングできます。次に例を示します。

```
' Visual Basic
Dim cmd As ULCommand = new ULCommand( _
    "SELECT * FROM Departments", conn _
  )
Dim res As IAsyncResult res = _
  cmd.BeginExecuteReader()
While( !res.IsCompleted )
  ' do other work
End While
' this will block until the command completes
Dim reader As ULDataReader = _
  cmd.EndExecuteReader( res )

// C#
ULCommand cmd = new ULCommand(
    "SELECT * FROM Departments", conn
  );
IAsyncResult res = cmd.BeginExecuteReader();
while( !res.IsCompleted ) {
  // do other work
}
// this will block until the command completes
ULDataReader reader = cmd.EndExecuteReader( res );
```

Use the IAsyncResult.AsyncWaitHandle property to get a synchronization object IAsyncResult.AsyncWaitHandle プロパティを使用して同期オブジェクトを取得し、その状態で待機できます。次に例を示します。

```
' Visual Basic
Dim cmd As ULCommand = new ULCommand( _
    "SELECT * FROM Departments", conn _
  )
Dim res As IAsyncResult res = _
  cmd.BeginExecuteReader()
' perform other work
Dim wh As WaitHandle = res.AsyncWaitHandle
wh.WaitOne()
' this will not block because the command is finished
Dim reader As ULDataReader = _
  cmd.EndExecuteReader( res )

// C#
ULCommand cmd = new ULCommand(
```



```
    "SELECT * FROM Departments", conn
  );
IAsyncResult res = cmd.BeginExecuteReader();
// perform other work
WaitHandle wh = res.AsyncWaitHandle;
wh.WaitOne();
// this will not block because the command is finished
ULDataReader reader = cmd.EndExecuteReader( res );
```

Specify a callback function when calling BeginExecuteReader BeginExecuteReader の呼び出し時にコールバック関数を指定できます。次に例を示します。

```
' Visual Basic
Private Sub callbackFunction(ByVal ar As IAsyncResult)
  Dim cmd As ULCommand = _
    CType(ar.AsyncState, ULCommand)
  ' this won't block since the command has completed
  Dim reader As ULDataReader = cmd.EndExecuteReader()
End Sub


' elsewhere in the code
Private Sub DoStuff()
  Dim cmd As ULCommand = new ULCommand( _
      "SELECT * FROM Departments", conn _
    )
  Dim res As IAsyncResult = _
    cmd.BeginExecuteReader( _
       callbackFunction, cmd _
    )
  ' perform other work.  The callback function
  ' will be called when the command completes
End Sub


// C#
private void callbackFunction( IAsyncResult ar )
{
  ULCommand cmd = (ULCommand) ar.AsyncState;
  // this won't block since the command has completed
  ULDataReader reader = cmd.EndExecuteReader();
}


// elsewhere in the code
private void DoStuff()
{
  ULCommand cmd = new ULCommand(
      "SELECT * FROM Departments", conn
    );
  IAsyncResult res = cmd.BeginExecuteReader(
      callbackFunction, cmd
    );
  // perform other work.  The callback function
  // will be called when the command completes
}
```

コールバック関数は別のスレッドで実行するため、スレッド化されたプログラム内でのユーザ・インタフェースの更新に関する通常の注意が適用されます。

参照

- 「ULCommand クラス」 91 ページ
- 「ULCommand メンバ」 92 ページ
- 「BeginExecuteReader() メソッド」 107 ページ
- 「ULDataReader クラス」 276 ページ
- IAsyncResult

ExecuteNonQuery メソッド

SQL INSERT 文、DELETE 文、UPDATE 文のように、結果セットを返さない文を実行します。

構文

Visual Basic
Public Overrides Function **ExecuteNonQuery()** As Integer

C#
public override int **ExecuteNonQuery();**

戻り値

影響を受けたローの数。

備考

この文は、必要に応じて ULCommand.CommandText と ULCommand.Parameters が指定された、現在の ULCommand オブジェクトです。

UPDATE、INSERT、DELETE 文の場合、戻り値はコマンドの影響を受けるローの数です。その他すべての文のタイプとロールバックの場合、戻り値は -1 です。

ULCommand.CommandType を System.Data.CommandType.TableDirect にすることはできません。

参照

- 「ULCommand クラス」 91 ページ
- 「ULCommand メンバ」 92 ページ
- 「CommandText プロパティ」 99 ページ
- 「Parameters プロパティ」 103 ページ
- 「CommandType プロパティ」 100 ページ
- CommandType.TableDirect
- 「Connection プロパティ」 101 ページ
- 「Transaction プロパティ」 104 ページ
- 「CommandText プロパティ」 99 ページ

ExecuteReader メソッド

SQL SELECT 文を実行し、結果セットを返します。

ExecuteReader() メソッド

SQL SELECT 文を実行し、結果セットを返します。

構文

Visual Basic
Public Function **ExecuteReader()** As ULDataReader

C#
public ULDataReader **ExecuteReader();**

戻り値

ULDataReader オブジェクトとして返される結果セット。

備考

この文は、必要に応じて ULCommand.CommandText と ULCommand.Parameters が指定された、現在の ULCommand オブジェクトです。ULDataReader オブジェクトは、読み込み専用の結果セットです。編集可能な結果セットには、ULCommand.ExecuteResultSet()、ULCommand.ExecuteTable()、または ULDataAdapter を使用します。

ULCommand.CommandType が System.Data.CommandType.TableDirect である場合、ExecuteReader は ULCommand.ExecuteTable() を実行し、ULDataReader としての ULTable ダウンキャストを返します。

SELECT 文は、パフォーマンス上の理由により、デフォルトで読み込み専用とマーク付けされます。クエリを使用して更新を行う場合、SELECT 文は "FOR UPDATE" で終わる必要があります。

これは、System.Data.IDbCommand.ExecuteReader() と System.Data.Common.DbCommand.ExecuteReader() が厳密に型指定されたものです。

参照

- 「ULCommand クラス」 91 ページ
- 「ULCommand メンバ」 92 ページ
- 「ExecuteReader メソッド」 118 ページ
- 「ExecuteReader(CommandBehavior) メソッド」 120 ページ
- 「CommandText プロパティ」 99 ページ
- 「Parameters プロパティ」 103 ページ
- 「ULDataReader クラス」 276 ページ
- 「ExecuteResultSet() メソッド」 122 ページ
- 「ExecuteTable() メソッド」 125 ページ
- 「ULDataAdapter クラス」 242 ページ
- 「CommandType プロパティ」 100 ページ
- CommandType.TableDirect
- 「ExecuteTable() メソッド」 125 ページ
- 「ULTable クラス」 548 ページ
- 「ULDataReader クラス」 276 ページ
- IDbCommand.ExecuteReader
- DbCommand.ExecuteReader
- 「Connection プロパティ」 101 ページ
- 「Transaction プロパティ」 104 ページ

ExecuteReader(CommandBehavior) メソッド

コマンドの動作を指定して SQL SELECT 文を実行し、結果セットを返します。

構文

Visual Basic

```
Public Function ExecuteReader( _
  ByVal cmdBehavior As CommandBehavior _
) As ULDataReader
```

C#

```
public ULDataReader ExecuteReader(
  CommandBehavior cmdBehavior
);
```

パラメータ

- **cmdBehavior**　クエリの結果の記述と、接続への影響の記述の System.Data.CommandBehavior フラグのビット単位の組み合わせ。Ultra Light.NET で有効なのは、System.Data.CommandBehavior.Default フラグ、System.Data.CommandBehavior.CloseConnection フラグ、System.Data.CommandBehavior.SchemaOnly フラグだけです。

戻り値

ULDataReader オブジェクトとして返される結果セット。

備考

この文は、必要に応じて ULCommand.CommandText と ULCommand.Parameters が指定された、現在の ULCommand オブジェクトです。ULDataReader オブジェクトは、読み込み専用の結果セットです。編集可能な結果セットには、ULCommand.ExecuteResultSet(CommandBehavior)、ULCommand.ExecuteTable(CommandBehavior)、または ULDataAdapter を使用します。

ULCommand.CommandType が System.Data.CommandType.TableDirect である場合、ExecuteReader は ULCommand.ExecuteTable(CommandBehavior) を実行し、ULDataReader としての ULTable ダウンキャストを返します。

SELECT 文は、パフォーマンス上の理由により、デフォルトで読み込み専用とマーク付けされます。クエリを使用して更新を行う場合、SELECT 文は "FOR UPDATE" で終わる必要があります。

これは、System.Data.IDbCommand.ExecuteReader(System.Data.CommandBehavior) と System.Data.Common.DbCommand.ExecuteReader(System.Data.CommandBehavior) が厳密に型指定されたものです。

参照

- 「ULCommand クラス」 91 ページ
- 「ULCommand メンバ」 92 ページ
- 「ExecuteReader メソッド」 118 ページ
- 「ExecuteReader() メソッド」 119 ページ
- 「CommandText プロパティ」 99 ページ
- 「Parameters プロパティ」 103 ページ
- 「ULDataReader クラス」 276 ページ
- 「ExecuteResultSet(CommandBehavior) メソッド」 123 ページ
- 「ExecuteTable(CommandBehavior) メソッド」 126 ページ
- 「ULDataAdapter クラス」 242 ページ
- 「CommandType プロパティ」 100 ページ
- CommandType.TableDirect
- 「ExecuteTable(CommandBehavior) メソッド」 126 ページ
- 「ULTable クラス」 548 ページ
- 「ULDataReader クラス」 276 ページ
- IDbCommand.ExecuteReader
- DbCommand.ExecuteReader
- CommandBehavior
- CommandBehavior.Default
- CommandBehavior.CloseConnection
- CommandBehavior.SchemaOnly
- 「Connection プロパティ」 101 ページ
- 「Transaction プロパティ」 104 ページ
- 「CommandText プロパティ」 99 ページ

ExecuteResultSet メソッド

UL 拡張：SQL SELECT 文を実行し、ULResultSet として結果セットを返します。

ExecuteResultSet() メソッド

UL 拡張：SQL SELECT 文を実行し、ULResultSet として結果セットを返します。

構文

Visual Basic
Public Function **ExecuteResultSet()** As ULResultSet

C#
public ULResultSet **ExecuteResultSet();**

戻り値

ULResultSet オブジェクトとして返される結果セット。

備考

この文は、必要に応じて ULCommand.CommandText と ULCommand.Parameters が指定された、現在の ULCommand オブジェクトです。ULResultSet オブジェクトは、位置付け更新や削除の実行対象となる編集可能な結果を表します。編集可能な結果セットには、ULCommand.ExecuteTable() または ULDataAdapter を使用します。

ULCommand.CommandType が System.Data.CommandType.TableDirect である場合、ExecuteReader は ULCommand.ExecuteTable() を実行し、ULResultSet としての ULTable ダウンキャストを返します。

ULCommand.ExecuteResultSet は、Dynamic SQL による位置付け更新と削除をサポートします。

例

```
cmd.CommandText = "SELECT id, season, price FROM OurProducts";
ULResultSet rs = cmd.ExecuteResultSet();
while( rs.Read() ) {
    string season = rs.GetString( 1 );
    double price = rs.GetDouble( 2 );
    if( season.Equals( "summer" ) && price < 100.0 ) {
        rs.SetDouble( 2, price * .5 );
        rs.Update();
    }
    if( season.Equals( "discontinued" ) ) {
        rs.Delete();
    }
}
rs.Close();
```


参照

- 「ULCommand クラス」 91 ページ
- 「ULCommand メンバ」 92 ページ
- 「ExecuteResultSet メソッド」 121 ページ
- 「ExecuteResultSet(CommandBehavior) メソッド」 123 ページ
- 「ULCommand クラス」 91 ページ
- 「CommandText プロパティ」 99 ページ
- 「Parameters プロパティ」 103 ページ
- 「CommandType プロパティ」 100 ページ
- CommandType.TableDirect
- 「ULResultSet クラス」 415 ページ
- 「ULTable クラス」 548 ページ

ExecuteResultSet(CommandBehavior) メソッド

UL 拡張：コマンドの動作を指定して SQL SELECT 文を実行し、ULResultSet として結果セットを返します。

構文

Visual Basic

```
Public Function ExecuteResultSet( _
  ByVal cmdBehavior As CommandBehavior _
) As ULResultSet
```

C#

```
public ULResultSet ExecuteResultSet(
  CommandBehavior cmdBehavior
);
```

パラメータ

- **cmdBehavior**　クエリの結果の記述と、接続への影響の記述の System.Data.CommandBehavior フラグのビット単位の組み合わせ。Ultra Light.NET で有効なのは、System.Data.CommandBehavior.Default フラグ、System.Data.CommandBehavior.CloseConnection フラグ、System.Data.CommandBehavior.SchemaOnly フラグだけです。

戻り値

ULResultSet オブジェクトとして返される結果セット。

備考

この文は、必要に応じて ULCommand.CommandText と ULCommand.Parameters が指定された、現在の ULCommand オブジェクトです。ULResultSet オブジェクトは、位置付け更新や削除の実行対象となる編集可能な結果を表します。編集可能な結果セットには、ULCommand.ExecuteTable(CommandBehavior) または ULDataAdapter を使用します。

ULCommand.CommandType が System.Data.CommandType.TableDirect である場合、ExecuteReader は ULCommand.ExecuteTable(CommandBehavior) を実行し、ULResultSet としての ULTable ダウンキャストを返します。

ULCommand.ExecuteResultSet は、Dynamic SQL による位置付け更新と削除をサポートします。

参照

- 「ULCommand クラス」 91 ページ
- 「ULCommand メンバ」 92 ページ
- 「ExecuteResultSet メソッド」 121 ページ
- 「ExecuteReader() メソッド」 119 ページ
- 「ULResultSet クラス」 415 ページ
- 「CommandText プロパティ」 99 ページ
- 「Parameters プロパティ」 103 ページ
- 「ExecuteTable(CommandBehavior) メソッド」 126 ページ
- 「ULDataAdapter クラス」 242 ページ
- 「CommandType プロパティ」 100 ページ
- CommandType.TableDirect
- 「ExecuteTable(CommandBehavior) メソッド」 126 ページ
- 「ULTable クラス」 548 ページ
- CommandBehavior
- CommandBehavior.Default
- CommandBehavior.CloseConnection
- CommandBehavior.SchemaOnly
- 「Connection プロパティ」 101 ページ
- 「Transaction プロパティ」 104 ページ

ExecuteScalar メソッド

SQL SELECT 文を実行し、単一の値を返します。

構文

Visual Basic
Public Overrides Function **ExecuteScalar()** As Object

C#
public override object **ExecuteScalar();**

戻り値

結果セットの最初のローの最初のカラム。結果セットが空の場合は NULL 参照 (Visual Basic の Nothing)。

備考

この文は、必要に応じて ULCommand.CommandText と ULCommand.Parameters が指定された、現在の ULCommand オブジェクトです。

複数のローとカラムを返すクエリでこのメソッドが呼び出されると、最初のローの最初のカラムのみが返されます。

ULCommand.CommandType が System.Data.CommandType.TableDirect である場合、ExecuteScalar は ULCommand.ExecuteTable() を実行し、最初のローの最初のカラムを返します。

SELECT 文は、パフォーマンス上の理由により、デフォルトで読み込み専用とマーク付けされます。クエリを使用して更新を行う場合、SELECT 文は "FOR UPDATE" で終わる必要があります。

参照

- 「ULCommand クラス」 91 ページ
- 「ULCommand メンバ」 92 ページ
- 「Connection プロパティ」 101 ページ
- 「Transaction プロパティ」 104 ページ
- 「CommandText プロパティ」 99 ページ
- 「IndexName プロパティ」 102 ページ
- CommandBehavior
- CommandBehavior.Default
- CommandBehavior.CloseConnection
- CommandBehavior.SchemaOnly
- 「CommandType プロパティ」 100 ページ
- 「Parameters プロパティ」 103 ページ
- CommandType.TableDirect

ExecuteTable メソッド

UL 拡張：直接の操作用に、ULTable にデータベース・テーブルを取り出します。ULCommand.CommandText はテーブルの名前として解釈され、ULCommand.IndexName はテーブルのソート順の指定に使用できます。

ExecuteTable() メソッド

UL 拡張：直接の操作用に、ULTable にデータベース・テーブルを取り出します。ULCommand.CommandText はテーブルの名前として解釈され、ULCommand.IndexName はテーブルのソート順の指定に使用できます。

構文

Visual Basic
Public Function **ExecuteTable()** As ULTable

C#
public ULTable **ExecuteTable();**

戻り値

ULTable オブジェクトとして返されるテーブル。

備考

ULCommand.CommandType は System.Data.CommandType.TableDirect に設定することが必要です。

ULCommand.IndexName が NULL 参照 (Visual Basic の Nothing) であると、プライマリ・キーを使用してテーブルが開かれます。そうでない場合は、ソート基準となるインデックスの名前として ULCommand.IndexName 値を使用してテーブルが開かれます。

参照

- 「ULCommand クラス」 91 ページ
- 「ULCommand メンバ」 92 ページ
- 「ExecuteTable メソッド」 125 ページ
- 「ExecuteTable(CommandBehavior) メソッド」 126 ページ
- 「Connection プロパティ」 101 ページ
- 「Transaction プロパティ」 104 ページ
- 「CommandText プロパティ」 99 ページ
- 「IndexName プロパティ」 102 ページ
- CommandBehavior
- CommandBehavior.Default
- CommandBehavior.CloseConnection
- CommandBehavior.SchemaOnly
- 「CommandType プロパティ」 100 ページ
- CommandType.TableDirect
- 「ULTable クラス」 548 ページ

ExecuteTable(CommandBehavior) メソッド

UL 拡張：直接の操作用に、コマンド動作を指定してデータベース・テーブルを取り出します。ULCommand.CommandText はテーブルの名前として解釈され、ULCommand.IndexName はテーブルのソート順の指定に使用できます。

構文

Visual Basic

```
Public Function ExecuteTable( _
  ByVal cmdBehavior As CommandBehavior _
) As ULTable
```

C#

```
public ULTable ExecuteTable(
  CommandBehavior cmdBehavior
);
```

パラメータ

- **cmdBehavior**　クエリの結果の記述と、接続への影響の記述の System.Data.CommandBehavior フラグのビット単位の組み合わせ。Ultra Light.NET で有効なのは、System.Data.CommandBehavior.Default フラグ、System.Data.CommandBehavior.CloseConnection フラグ、System.Data.CommandBehavior.SchemaOnly フラグだけです。

戻り値

ULTable オブジェクトとして返されるテーブル。

備考

ULCommand.CommandType は System.Data.CommandType.TableDirect に設定することが必要です。

ULCommand.IndexName が NULL 参照 (Visual Basic の Nothing) であると、プライマリ・キーを使用してテーブルが開かれます。そうでない場合は、ソート基準となるインデックスの名前として ULCommand.IndexName 値を使用してテーブルが開かれます。

参照

- 「ULCommand クラス」 91 ページ
- 「ULCommand メンバ」 92 ページ
- 「ExecuteTable メソッド」 125 ページ
- 「ExecuteTable() メソッド」 125 ページ
- 「Connection プロパティ」 101 ページ
- 「Transaction プロパティ」 104 ページ
- 「CommandText プロパティ」 99 ページ
- 「IndexName プロパティ」 102 ページ
- CommandBehavior
- CommandBehavior.Default
- CommandBehavior.CloseConnection
- CommandBehavior.SchemaOnly
- 「CommandType プロパティ」 100 ページ
- CommandType.TableDirect

Prepare メソッド

このコマンドの SQL 文を事前にコンパイルして格納します。

構文

Visual Basic
Public Overrides Sub **Prepare()**

C#
public override void **Prepare();**

備考

文を事前にコンパイルすると、パラメータ値のみが変更されているときに文を効率的に再使用できます。このコマンドの他のプロパティを変更すると、文の準備は解除されます。

準備が解除されたコマンドはいずれも、さまざまな Execute メソッドへの呼び出し時に準備が行われるため、Ultra Light.NET では明示的に文を準備する必要はありません。

参照

- 「ULCommand クラス」 91 ページ
- 「ULCommand メンバ」 92 ページ
- 「Connection プロパティ」 101 ページ
- 「Transaction プロパティ」 104 ページ
- 「CommandText プロパティ」 99 ページ

ULCommandBuilder クラス

System.Data.DataSet の変更内容を、関連するデータベース内のデータに一致させる単一テーブルのコマンドを、自動的に生成します。

構文

Visual Basic
Public Class **ULCommandBuilder**
  Inherits DbCommandBuilder

C#
public class **ULCommandBuilder**: DbCommandBuilder

備考

ULDataAdapter は、System.Data.DataSet の変更内容を関連するデータ・ソース内のデータに一致させるために必要な SQL 文を、自動的には生成しません。ただし、ULDataAdapter の SelectCommand プロパティを設定すると単一テーブルを更新する SQL 文を自動的に生成するような、ULCommandBuilder オブジェクトを作成することはできます。これにより、設定していない追加の SQL 文が ULCommandBuilder によって生成されます。

継承：System.ComponentModel.Component

実装：System.IDisposable

例

次の例では、ULCommand を ULDataAdapter と ULConnection とともに使用して、データ・ソースからローを選択します。この例では、接続文字列、クエリ文字列 (SQL SELECT 文)、データベース・テーブル名を表す文字列を受け取り、次に ULCommandBuilder を作成します。

```
' Visual Basic
Public Shared Function SelectULRows(ByVal connectionString As String, _
    ByVal queryString As String, ByVal tableName As String)

    Dim connection As ULConnection = New ULConnection(connectionString)
    Dim adapter As ULDataAdapter = New ULDataAdapter()

      adapter.SelectCommand = New ULCommand(queryString, connection)

      Dim builder As ULCommandBuilder = New ULCommandBuilder(adapter)

    connection.Open()

    Dim dataSet As DataSet = New DataSet()
    adapter.Fill(dataSet, tableName)

    'code to modify data in DataSet here

    'Without the ULCommandBuilder this line would fail
    adapter.Update(dataSet, tableName)

    Return dataSet

End Function
```

```
// C#
public static DataSet SelectULRows(string connectionString,
    string queryString, string tableName)
{
    using (ULConnection connection = new ULConnection(connectionString))
    {
        ULDataAdapter adapter = new ULDataAdapter();
        adapter.SelectCommand = new ULCommand(queryString, connection);
        ULCommandBuilder builder = new ULCommandBuilder(adapter);

        connection.Open();

        DataSet dataSet = new DataSet();
        adapter.Fill(dataSet, tableName);

          //code to modify data in DataSet here

        //Without the ULCommandBuilder this line would fail
        adapter.Update(dataSet, tableName);

        return dataSet;
    }
}
```

参照

- 「ULCommandBuilder メンバ」 130 ページ
- DataSet
- 「ULDataAdapter クラス」 242 ページ
- コンポーネント
- IDisposable
- 「ULCommand クラス」 91 ページ
- 「ULConnection クラス」 139 ページ

ULCommandBuilder メンバ

パブリック・コンストラクタ

メンバ名	説明
「ULCommandBuilder コンストラクタ」 132 ページ	「ULCommandBuilder クラス」 129 ページの新しいインスタンスを初期化します。

パブリック・プロパティ

メンバ名	説明
CatalogLocation (DbCommandBuilder から継承)	DbCommandBuilder のインスタンス用に CatalogLocation を設定または取得します。

メンバ名	説明
CatalogSeparator (DbCommandBuilder から継承)	DbCommandBuilder のインスタンスのカタログ・セパレータとして使用する文字列を設定または取得します。
ConflictOption (DbCommandBuilder から継承)	DbCommandBuilder によって使用される ConflictOption を指定します。
「DataAdapter プロパティ」 133 ページ	SQL 文が自動的に生成される ULDataAdapter オブジェクトを取得または設定します。
QuotePrefix (DbCommandBuilder から継承)	スペースや予約語などの文字列が名前に含まれているデータベース・オブジェクト (テーブルやカラムなど) を指定するときに使用する先頭の文字または文字列を取得または設定します。
QuoteSuffix (DbCommandBuilder から継承)	スペースや予約語などの文字列が名前に含まれているデータベース・オブジェクト (テーブルやカラムなど) を指定するときに使用する先頭の文字または文字列を取得または設定します。
SchemaSeparator (DbCommandBuilder から継承)	スキーマ識別子とその他の識別子とを区切るセパレータに使用する文字を取得または設定します。
SetAllValues (DbCommandBuilder から継承)	UPDATE 文のすべてのカラム値が含まれるか、変更された値のみが含まれるかを指定します。

パブリック・メソッド

メンバ名	説明
「GetDeleteCommand メソッド」 133 ページ	データ・ソースでの削除に必要な自動生成された DbCommand オブジェクトを取得します。
「GetInsertCommand メソッド」 135 ページ	データ・ソースでの挿入に必要な自動生成された DbCommand オブジェクトを取得します。
「GetUpdateCommand メソッド」 137 ページ	データ・ソースでの更新に必要な自動生成された DbCommand オブジェクトを取得します。
QuoteIdentifier (DbCommandBuilder から継承)	カタログの大文字／小文字が正しい引用符なしの識別子が指定されると、識別子に埋め込まれる引用符を適切にエスケープして、正しい形式の引用符付き識別子を返します。
RefreshSchema (DbCommandBuilder から継承)	この DbCommandBuilder に関連付けられたコマンドをクリアします。

メンバ名	説明
UnquoteIdentifier (DbCommandBuilder から継承)	引用符付き識別子が指定されると、識別子に埋め込まれた引用符を適切にアンエスケープして、正しい形式の引用符なしの識別子を返します。

参照

- 「ULCommandBuilder クラス」 129 ページ
- DataSet
- 「ULDataAdapter クラス」 242 ページ
- コンポーネント
- IDisposable
- 「ULCommand クラス」 91 ページ
- 「ULConnection クラス」 139 ページ

ULCommandBuilder コンストラクタ

「ULCommandBuilder クラス」 129 ページの新しいインスタンスを初期化します。

ULCommandBuilder() コンストラクタ

ULCommandBuilder オブジェクトを初期化します。

構文

Visual Basic
Public Sub **New()**

C#
public **ULCommandBuilder();**

参照

- 「ULCommandBuilder クラス」 129 ページ
- 「ULCommandBuilder メンバ」 130 ページ
- 「ULCommandBuilder コンストラクタ」 132 ページ
- 「ULCommandBuilder(ULDataAdapter) コンストラクタ」 132 ページ

ULCommandBuilder(ULDataAdapter) コンストラクタ

ULCommandBuilder オブジェクトを、指定された ULDataAdapter オブジェクトで初期化します。

構文

Visual Basic
Public Sub **New(_**


```
  ByVal adapter As ULDataAdapter _
)

C#
public  ULCommandBuilder(
  ULDataAdapter adapter
);
```

パラメータ

- **adapter**　ULDataAdapter オブジェクト。

参照

- 「ULCommandBuilder クラス」 129 ページ
- 「ULCommandBuilder メンバ」 130 ページ
- 「ULCommandBuilder コンストラクタ」 132 ページ
- 「ULDataAdapter クラス」 242 ページ

DataAdapter プロパティ

SQL 文が自動的に生成される ULDataAdapter オブジェクトを取得または設定します。

構文

Visual Basic
Public Property **DataAdapter** As ULDataAdapter

C#
public ULDataAdapter **DataAdapter** { get; set; }

プロパティ値

ULDataAdapter オブジェクト。

参照

- 「ULCommandBuilder クラス」 129 ページ
- 「ULCommandBuilder メンバ」 130 ページ
- 「ULDataAdapter クラス」 242 ページ

GetDeleteCommand メソッド

データ・ソースでの削除に必要な自動生成された DbCommand オブジェクトを取得します。

GetDeleteCommand(Boolean) メソッド

データベースで削除処理を実行するのに必要な、自動的に生成された ULCommand オブジェクトを取得します。

構文

Visual Basic
Public Function **GetDeleteCommand(_**
  ByVal *useColumnsForParameterNames* As Boolean _
) As ULCommand

C#
public ULCommand **GetDeleteCommand(**
  bool *useColumnsForParameterNames*
);

パラメータ

- **useColumnsForParameterNames**　true の場合、可能であれば、カラム名に一致するパラメータ名を生成します。false の場合、@p1、@p2 などを生成します。

戻り値

削除の実行に必要な、自動的に生成された ULCommand オブジェクト。

備考

SQL 文が最初に生成された後で、アプリケーションが ULDataAdapter.SelectCommand を変更する場合、DbCommandBuilder.RefreshSchema を明示的に呼び出す必要があります。この処理を行わないと、GetDeleteCommand メソッドは、正しくない可能性がある古い文の情報を使用し続けます。SQL 文が最初に生成されるのは、アプリケーションが DbDataAdapter.Update(System.Data.DataSet) または GetDeleteCommand メソッドを呼び出したときです。

参照

- 「ULCommandBuilder クラス」 129 ページ
- 「ULCommandBuilder メンバ」 130 ページ
- 「GetDeleteCommand メソッド」 133 ページ
- 「ULCommand クラス」 91 ページ
- DbCommandBuilder.RefreshSchema
- 「SelectCommand プロパティ」 249 ページ
- DbDataAdapter.Update
- DbCommandBuilder.DataAdapter

GetDeleteCommand() メソッド

データベースで削除処理を実行するのに必要な、自動的に生成された ULCommand オブジェクトを取得します。

構文

Visual Basic
Public Function **GetDeleteCommand()** As ULCommand

C#
public ULCommand **GetDeleteCommand();**

戻り値

削除の実行に必要な、自動的に生成された ULCommand オブジェクト。

備考

SQL 文が最初に生成された後で、アプリケーションが ULDataAdapter.SelectCommand を変更する場合、DbCommandBuilder.RefreshSchema を明示的に呼び出す必要があります。この処理を行わないと、GetDeleteCommand メソッドは、正しくない可能性がある古い文の情報を使用し続けます。SQL 文が最初に生成されるのは、アプリケーションが DbDataAdapter.Update(System.Data.DataSet) または GetDeleteCommand メソッドを呼び出したときです。

参照

- 「ULCommandBuilder クラス」 129 ページ
- 「ULCommandBuilder メンバ」 130 ページ
- 「GetDeleteCommand メソッド」 133 ページ
- 「ULCommand クラス」 91 ページ
- DbCommandBuilder.RefreshSchema
- 「SelectCommand プロパティ」 249 ページ
- DbDataAdapter.Update
- DbCommandBuilder.DataAdapter

GetInsertCommand メソッド

データ・ソースでの挿入に必要な自動生成された DbCommand オブジェクトを取得します。

GetInsertCommand(Boolean) メソッド

データベースで挿入処理を実行するのに必要な、自動的に生成された ULCommand オブジェクトを取得します。

構文

Visual Basic

```
Public Function GetInsertCommand( _
  ByVal useColumnsForParameterNames As Boolean _
) As ULCommand
```

C#

```
public ULCommand GetInsertCommand(
  bool useColumnsForParameterNames
);
```

パラメータ

- **useColumnsForParameterNames** true の場合、可能であれば、カラム名に一致するパラメータ名を生成します。false の場合、@p1、@p2 などを生成します。

戻り値

挿入の実行に必要な、自動的に生成された ULCommand オブジェクト。

備考

SQL 文が最初に生成された後で、アプリケーションが ULDataAdapter.SelectCommand を変更する場合、DbCommandBuilder.RefreshSchema を明示的に呼び出す必要があります。この処理を行わないと、GetInsertCommand メソッドは、正しくない可能性がある古い文の情報を使用し続けます。SQL 文が最初に生成されるのは、アプリケーションが DbDataAdapter.Update(System.Data.DataSet) または GetInsertCommand メソッドを呼び出したときです。

参照

- 「ULCommandBuilder クラス」 129 ページ
- 「ULCommandBuilder メンバ」 130 ページ
- 「GetInsertCommand メソッド」 135 ページ
- 「ULCommand クラス」 91 ページ
- DbCommandBuilder.RefreshSchema
- 「SelectCommand プロパティ」 249 ページ
- DbDataAdapter.Update
- DbCommandBuilder.DataAdapter

GetInsertCommand() メソッド

データベースで挿入処理を実行するのに必要な、自動的に生成された ULCommand オブジェクトを取得します。

構文

Visual Basic
Public Function **GetInsertCommand()** As ULCommand

C#
public ULCommand **GetInsertCommand();**

戻り値

挿入の実行に必要な、自動的に生成された ULCommand オブジェクト。

備考

SQL 文が最初に生成された後で、アプリケーションが ULDataAdapter.SelectCommand を変更する場合、DbCommandBuilder.RefreshSchema を明示的に呼び出す必要があります。この処理を行わないと、GetInsertCommand メソッドは、正しくない可能性がある古い文の情報を使用し続けます。SQL 文が最初に生成されるのは、アプリケーションが DbDataAdapter.Update(System.Data.DataSet) または GetInsertCommand メソッドを呼び出したときです。

参照

- 「ULCommandBuilder クラス」 129 ページ
- 「ULCommandBuilder メンバ」 130 ページ
- 「GetInsertCommand メソッド」 135 ページ
- 「ULCommand クラス」 91 ページ
- DbCommandBuilder.RefreshSchema
- 「SelectCommand プロパティ」 249 ページ
- 「ULCommand クラス」 91 ページ
- DbCommandBuilder.DataAdapter

GetUpdateCommand メソッド

データ・ソースでの更新に必要な自動生成された DbCommand オブジェクトを取得します。

GetUpdateCommand(Boolean) メソッド

データベースで更新処理を実行するのに必要な、自動的に生成された ULCommand オブジェクトを取得します。

構文

Visual Basic

```
Public Function GetUpdateCommand( _
  ByVal useColumnsForParameterNames As Boolean _
) As ULCommand
```

C#

```
public ULCommand GetUpdateCommand(
  bool useColumnsForParameterNames
);
```

パラメータ

- **useColumnsForParameterNames** true の場合、可能であれば、カラム名に一致するパラメータ名を生成します。false の場合、@p1、@p2 などを生成します。

戻り値

更新の実行に必要な、自動的に生成された ULCommand オブジェクト。

備考

SQL 文が最初に生成された後で、アプリケーションが ULDataAdapter.SelectCommand を変更する場合、DbCommandBuilder.RefreshSchema を明示的に呼び出す必要があります。この処理を行わないと、GetUpdateCommand メソッドは、正しくない可能性がある古い文の情報を使用し続けます。SQL 文が最初に生成されるのは、アプリケーションが DbDataAdapter.Update(System.Data.DataSet) または GetUpdateCommand メソッドを呼び出したときです。

参照

- 「ULCommandBuilder クラス」 129 ページ
- 「ULCommandBuilder メンバ」 130 ページ
- 「GetUpdateCommand メソッド」 137 ページ
- 「ULCommand クラス」 91 ページ
- DbCommandBuilder.RefreshSchema
- 「SelectCommand プロパティ」 249 ページ
- DbDataAdapter.Update
- DbCommandBuilder.DataAdapter

GetUpdateCommand() メソッド

データベースで更新処理を実行するのに必要な、自動的に生成された ULCommand オブジェクトを取得します。

構文

Visual Basic
Public Function **GetUpdateCommand()** As ULCommand

C#
public ULCommand **GetUpdateCommand();**

戻り値

更新の実行に必要な、自動的に生成された ULCommand オブジェクト。

備考

SQL 文が最初に生成された後で、アプリケーションが ULDataAdapter.SelectCommand を変更する場合、DbCommandBuilder.RefreshSchema を明示的に呼び出す必要があります。この処理を行わないと、GetUpdateCommand メソッドは、正しくない可能性がある古い文の情報を使用し続けます。SQL 文が最初に生成されるのは、アプリケーションが DbDataAdapter.Update(System.Data.DataSet) または GetUpdateCommand メソッドを呼び出したときです。

参照

- 「ULCommandBuilder クラス」 129 ページ
- 「ULCommandBuilder メンバ」 130 ページ
- 「GetUpdateCommand メソッド」 137 ページ
- 「ULCommand クラス」 91 ページ
- DbCommandBuilder.RefreshSchema
- 「SelectCommand プロパティ」 249 ページ
- DbDataAdapter.Update
- DbCommandBuilder.DataAdapter

ULConnection クラス

Ultra Light.NET データベースへの接続を表します。このクラスは継承できません。

構文

Visual Basic
Public NotInheritable Class **ULConnection**
  Inherits DbConnection

C#
public sealed class **ULConnection**: DbConnection

備考

Ultra Light.NET の Ultra Light エンジンのランタイムを使用するには、ULDatabaseManager.RuntimeType を適切な値に設定してから、他の Ultra Light.NET API を使用します。

既存のデータベースへの接続は、ULConnection.Open を使用して開きます。

接続は、他の操作の実行前に開きます。また、その接続での操作がすべて終了したら、接続を閉じてからアプリケーションを終了します。また、接続で開いた結果セットとテーブルをすべて閉じてから、その接続を閉じます。

データベースのスキーマには、開いている接続の ULConnection.Schema を使用してアクセスできます。

実装：System.Data.IDbConnection、System.IDisposable

参照

- 「ULConnection メンバ」 139 ページ
- 「RuntimeType プロパティ」 256 ページ
- 「Open メソッド」 177 ページ
- 「Schema プロパティ」 151 ページ
- IDbConnection
- IDisposable

ULConnection メンバ

パブリック・コンストラクタ

メンバ名	説明
「ULConnection コンストラクタ」 143 ページ	「ULConnection クラス」 139 ページの新しいインスタンスを初期化します。

パブリック・フィールド

メンバ名	説明
「INVALID_DATABASE_ID フィールド」 145 ページ	**UL 拡張：** ULConnection.DatabaseID が設定されていないことを示すデータベース ID の定数です。このフィールドは定数で、読み込み専用です。

パブリック・プロパティ

メンバ名	説明
「ConnectionString プロパティ」 146 ページ	Ultra Light.NET データベースへの接続を開くためのパラメータを指定します。接続文字列は、ULConnectionParms オブジェクトを使用して指定できます。
「ConnectionTimeout プロパティ」 147 ページ	この機能は Ultra Light.NET ではサポートされていません。
「DataSource プロパティ」 147 ページ	この機能は Ultra Light.NET ではサポートされていません。
「Database プロパティ」 148 ページ	接続を開くデータベースの名前を返します。
「DatabaseID プロパティ」 148 ページ	**UL 拡張：** グローバル・オートインクリメント・カラムに使用するデータベース ID の値を指定します。
「DatabaseManager プロパティ」 149 ページ	**UL 拡張：** シングルトンの ULDatabaseManager オブジェクトへのアクセスに使用します。
「GlobalAutoIncrementUsage プロパティ」 149 ページ	**UL 拡張：** 利用可能なグローバル・オートインクリメントの値の使用済み比率 (%) を返します。
「LastIdentity プロパティ」 150 ページ	**UL 拡張：** 直前に使用した identity の値を返します。
「Schema プロパティ」 151 ページ	**UL 拡張：** この接続に関連付けられている現在のデータベースのスキーマへのアクセスに使用します。
「ServerVersion プロパティ」 151 ページ	この機能は Ultra Light.NET ではサポートされていません。
「State プロパティ」 152 ページ	接続の現在のステータスを返します。
「SyncParms プロパティ」 152 ページ	**UL 拡張：** この接続の同期設定を指定します。
「SyncResult プロパティ」 153 ページ	**UL 拡張：** この接続の最後の同期結果を返します。

パブリック・メソッド

メンバ名	説明
「BeginTransaction メソッド」 153 ページ	データベース・トランザクションを起動します。
「CancelGetNotification メソッド」 155 ページ	指定された名前に一致する、すべてのキューに登録されている保留中の取得通知コールをキャンセルします。
「ChangeDatabase メソッド」 156 ページ	開いている ULConnection の現在のデータベースを変更します。
「ChangeEncryptionKey メソッド」 157 ページ	**UL 拡張：**データベースの暗号化キーを、指定された新しいキーに変更します。
「ChangePassword メソッド」 157 ページ	接続文字列に示されるユーザのパスワードを、指定される新しいパスワードに変更します。
「Close メソッド」 158 ページ	データベース接続を閉じます。
「CountUploadRows メソッド」 158 ページ	**UL 拡張：**次回の同期でアップロードする必要のあるロー数を返します。
「CreateCommand メソッド」 160 ページ	この接続と現在のトランザクションに関連付けられる ULCommand オブジェクトを作成して初期化します。ULCommand のプロパティを使用して、その動作を制御できます。
「CreateNotificationQueue メソッド」 161 ページ	イベント・キューを作成します。
「DeclareEvent メソッド」 162 ページ	指定されたイベントを宣言します。
「DestroyNotificationQueue メソッド」 163 ページ	イベント・キューを破棄します。
EnlistTransaction (DbConnection から継承)	指定されたトランザクションにエンリストします。
「ExecuteNextSQLPassthroughScript メソッド」 164 ページ	保留中の次の SQL パススルー・スクリプトを実行します。
「ExecuteSQLPassthroughScripts メソッド」 164 ページ	保留中のすべての SQL パススルー・スクリプトを実行します。

メンバ名	説明
「ExecuteTable メソッド」 165 ページ	**UL 拡張：**直接の操作用に、ULTable にデータベース・テーブルを取り出します。テーブルのプライマリ・キーを使用して、テーブルが開かれます (ソートされます)。
「GetLastDownloadTime メソッド」 169 ページ	**UL 拡張：**指定されたパブリケーションの最後のダウンロードの時刻を返します。
「GetNewUUID メソッド」 170 ページ	**UL 拡張：**新しい UUID (System.Guid) を生成します。
「GetNotification メソッド」 171 ページ	通知またはタイムアウトをブロックします。イベント名または NULL を返します。
「GetNotificationParameter メソッド」 172 ページ	GetNotification() によって読み込まれたイベントのパラメータ値を取得します。
「GetSQLPassthroughScriptCount メソッド」 173 ページ	保留中の SQL パススルー・スクリプトの数を取得します。
「GetSchema メソッド」 173 ページ	この DbConnection のデータ・ソースに関するスキーマ情報を返します。
「GrantConnectTo メソッド」 176 ページ	**UL 拡張：**パスワードを指定して、特定のユーザ ID に Ultra Light データベースへのアクセスを許可します。
「Open メソッド」 177 ページ	以前に指定された接続文字列を使用してデータベースへの接続を開きます。
「RegisterForEvent メソッド」 178 ページ	オブジェクトからイベントを取得するためのキューを登録します。
「ResetLastDownloadTime メソッド」 179 ページ	**UL 拡張：**最後のダウンロードの時刻をリセットします。
「RevokeConnectFrom メソッド」 179 ページ	**UL 拡張：**指定されたユーザ ID から Ultra Light データベースへのアクセス権を取り消します。
「RollbackPartialDownload メソッド」 180 ページ	**UL 拡張：**部分的なダウンロードからの、データベースへの未処理の変更をロールバックします。
「SendNotification メソッド」 180 ページ	一致するキューに通知を送信します。一致するキューの数を返します。
「StartSynchronizationDelete メソッド」 181 ページ	**UL 拡張：**同期用に、この接続によって行われる以後のすべての削除にマークを付けます。

メンバ名	説明
「StopSynchronizationDelete メソッド」 182 ページ	**UL 拡張：**削除操作が同期されないようにします。
「Synchronize メソッド」 182 ページ	**UL 拡張：**現在の ULConnection.SyncParms を使用してデータベースの同期をとります。
「TriggerEvent メソッド」 184 ページ	イベントをトリガします。送信済みの通知の数を返します。
「ValidateDatabase メソッド」 184 ページ	現在のデータベースの検証を実行します。

パブリック・イベント

メンバ名	説明
「InfoMessage イベント」 185 ページ	Ultra Light.NET が、この接続の警告または情報メッセージを送信するときに発生します。
「StateChange イベント」 187 ページ	この接続のステータスが変更されると発生します。

参照

- 「ULConnection クラス」 139 ページ
- 「RuntimeType プロパティ」 256 ページ
- 「Open メソッド」 177 ページ
- 「Schema プロパティ」 151 ページ
- IDbConnection
- IDisposable

ULConnection コンストラクタ

「ULConnection クラス」 139 ページの新しいインスタンスを初期化します。

ULConnection() コンストラクタ

ULConnection オブジェクトを初期化します。接続を開いてから、データベース操作を実行してください。

構文

Visual Basic
Public Sub **New()**

C#
public **ULConnection();**

備考

Ultra Light.NET の Ultra Light エンジンのランタイムを使用するには、ULDatabaseManager.RuntimeType を適切な値に設定してから、他の Ultra Light.NET API を使用します。

ULConnection オブジェクトは、ULConnection.ConnectionString が設定されていないと、開くことができません。

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「ULConnection コンストラクタ」 143 ページ
- 「Open メソッド」 177 ページ
- 「ConnectionString プロパティ」 146 ページ

ULConnection(String) コンストラクタ

ULConnection オブジェクトを、指定された接続文字列で初期化します。接続を開いてから、データベース操作を実行してください。

構文

Visual Basic
Public Sub **New(_**
  ByVal *connectionString* As String _
)

C#
public **ULConnection(**
  string *connectionString*
);

パラメータ

- **connectionString**　Ultra Light.NET の接続文字列。接続文字列は、キーワード値の組がセミコロンで区切られたリストです。

  パラメータのリストについては、「ConnectionString プロパティ」 146 ページを参照してください。

備考

Ultra Light.NET の Ultra Light エンジンのランタイムを使用するには、ULDatabaseManager.RuntimeType を適切な値に設定してから、他の Ultra Light.NET API を使用します。

接続文字列は、ULConnectionParms オブジェクトを使用して指定できます。

例

次のコードでは、Windows Mobile デバイス上のデータベース ¥UltraLite¥MyDatabase.udb への接続を作成して開きます。

```
' Visual Basic
Dim openParms As ULConnectionParms = New ULConnectionParms
openParms.DatabaseOnCE = "¥UltraLite¥MyDatabase.udb"
Dim conn As ULConnection = _
  New ULConnection( openParms.ToString() )
conn.Open()

// C#
ULConnectionParms openParms = new ULConnectionParms();
openParms.DatabaseOnCE = @"¥UltraLite¥MyDatabase.udb";
ULConnection conn = new ULConnection( openParms.ToString() );
conn.Open();
```

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「ULConnection コンストラクタ」 143 ページ
- 「Open メソッド」 177 ページ
- 「ULConnection() コンストラクタ」 143 ページ
- 「RuntimeType プロパティ」 256 ページ
- 「ULConnectionParms クラス」 189 ページ
- 「ConnectionString プロパティ」 146 ページ

INVALID_DATABASE_ID フィールド

UL 拡張：ULConnection.DatabaseID が設定されていないことを示すデータベース ID の定数です。このフィールドは定数で、読み込み専用です。

構文

Visual Basic
Public Shared **INVALID_DATABASE_ID** As Long

C#
public const long **INVALID_DATABASE_ID**;

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「DatabaseID プロパティ」 148 ページ

ConnectionString プロパティ

Ultra Light.NET データベースへの接続を開くためのパラメータを指定します。接続文字列は、ULConnectionParms オブジェクトを使用して指定できます。

構文

Visual Basic
Public Overrides Property **ConnectionString** As String

C#
public override string **ConnectionString** { get; set; }

プロパティ値

キーワードと値の組み合わせがセミコロンで区切られたリスト形式の、この接続を開くためのパラメータ。デフォルトは、空の文字列 (無効な接続文字列) です。

備考

UL 拡張：Ultra Light.NET によって使用されるパラメータは Ultra Light データベースに固有であるため、その接続文字列は SQL Anywhere の接続文字列との互換性がありません。パラメータのリストについては、「Ultra Light 接続パラメータ」『Ultra Light データベース管理とリファレンス』を参照してください。

パラメータ値は、単一引用符 (') または二重引用符 (") で囲むことができますが、引用符で囲まれた部分に同じ種類の引用符が含まれないことが条件になります。値にセミコロン (;)が含まれるか、値が引用符で始まるか、値の前後にスペースが必要な場合は、値を引用符で囲む必要があります。

パラメータ値を引用符で囲まない場合は、値にセミコロン (;) が含まれていないこと、また値が単一引用符 (') と二重引用符 (") のいずれかで始まることを確認してください。値の前後のスペースは無視されます。

デフォルトでは、UID=DBA と PWD=sql で接続が確立されます。データベースをより強力に保護するには、ユーザ名 DBA のパスワードを変更するか、新しいユーザを作成して (GrantConnectTo を使用)、DBA ユーザを削除します (RevokeConnectFrom を使用)。

例

次のコードでは、Windows Mobile デバイス上の既存のデータベース *¥UltraLite¥MyDatabase.udb* への接続を作成して開きます。

```
' Visual Basic
Dim openParms As ULConnectionParms = New ULConnectionParms
openParms.DatabaseOnCE = "¥UltraLite¥MyDatabase.udb"
Dim conn As ULConnection = New ULConnection
conn.ConnectionString = openParms.ToString()
conn.Open()

// C#
ULConnectionParms openParms = new ULConnectionParms();
openParms.DatabaseOnCE = @"¥UltraLite¥MyDatabase.udb";
ULConnection conn = new ULConnection();
```



```
conn.ConnectionString = openParms.ToString();
conn.Open();
```

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「Open メソッド」 177 ページ
- 「ULConnectionParms クラス」 189 ページ
- 「GrantConnectTo メソッド」 176 ページ

ConnectionTimeout プロパティ

この機能は Ultra Light.NET ではサポートされていません。

構文

Visual Basic
Public Overrides Readonly Property **ConnectionTimeout** As Integer

C#
public override int **ConnectionTimeout** { get;}

プロパティ値

値は常に 0 です。

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ

DataSource プロパティ

この機能は Ultra Light.NET ではサポートされていません。

構文

Visual Basic
Public Overrides Readonly Property **DataSource** As String

C#
public override string **DataSource** { get;}

プロパティ値

値は常に空の文字列です。

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ

Database プロパティ

接続を開くデータベースの名前を返します。

構文

Visual Basic
Public Overrides Readonly Property **Database** As String

C#
public override string **Database** { get;}

プロパティ値

データベースの名前が含まれる文字列。

備考

Windows Mobile デバイスでは、ULConnection は dbn、ce_file の順に接続文字列を参照します。

デスクトップ・マシンでは、ULConnection は dbn、nt_file の順に接続文字列を参照します。

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ

DatabaseID プロパティ

UL 拡張：グローバル・オートインクリメント・カラムに使用するデータベース ID の値を指定します。

構文

Visual Basic
Public Property **DatabaseID** As Long

C#
public long **DatabaseID** { get; set; }

プロパティ値

現在のデータベースのデータベース ID の値。

備考

データベース ID の値は、[0,System.UInt32.MaxValue] の範囲内であることが必要です。ULConnection.INVALID_DATABASE_ID の値は、現在のデータベースにデータベース ID が設定されていないことを示すのに使用されます。

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「GetDatabaseProperty メソッド」 269 ページ
- 「SetDatabaseOption メソッド」 273 ページ
- UInt32.MaxValue
- 「INVALID_DATABASE_ID フィールド」 145 ページ

DatabaseManager プロパティ

UL 拡張：シングルトンの ULDatabaseManager オブジェクトへのアクセスに使用します。

構文

Visual Basic
Public Shared Readonly Property **DatabaseManager** As ULDatabaseManager

C#
public const ULDatabaseManager **DatabaseManager** { get;}

プロパティ値

シングルトンの ULDatabaseManager オブジェクトへの参照。

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「ULDatabaseManager クラス」 255 ページ

GlobalAutoIncrementUsage プロパティ

UL 拡張：利用可能なグローバル・オートインクリメントの値の使用済み比率 (%) を返します。

構文

Visual Basic
Public Readonly Property **GlobalAutoIncrementUsage** As Short

C#
public short **GlobalAutoIncrementUsage** { get;}

プロパティ値

使用可能なグローバル・オートインクリメントの値の使用済み比率 (%)。これは、0 ～ 100 の範囲内の整数です。

備考

比率が 100% に近づいたら、ULConnection.DatabaseID を使用して、使用中のアプリケーションに新しいグローバル・データベース ID を設定してください。

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「ULDatabaseManager クラス」 255 ページ
- 「DatabaseID プロパティ」 148 ページ

LastIdentity プロパティ

UL 拡張：直前に使用した identity の値を返します。

構文

Visual Basic
Public Readonly Property **LastIdentity** As UInt64

C#
public ulong **LastIdentity** { get;}

プロパティ値

直前に使用した identity の値 (符号なし long)。

備考

直前に使用した identity の値です。このプロパティは、次の SQL Anywhere 文と同義です。

```
SELECT @@identity
```

LastIdentity は、グローバル・オートインクリメント・カラムで使うと特に便利です。

このプロパティでは最後に割り当てられたデフォルト値がわかるだけなので、間違った結果を取らないために INSERT 文を実行した直後にこの値を取り出してください。

ときには、1 つの INSERT 文にグローバル・オートインクリメント型のカラムが複数含まれていることがあります。この場合、LastIdentity は生成されたデフォルト値の 1 つですが、その値の生成元のカラムを判別する信頼できる方法はありません。このため、このような状況を回避するようなデータベースの設計と INSERT 文の記述を行ってください。

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ

Schema プロパティ

UL 拡張：この接続に関連付けられている現在のデータベースのスキーマへのアクセスに使用します。

構文

Visual Basic
Public Readonly Property **Schema** As ULDatabaseSchema

C#
public ULDatabaseSchema **Schema** { get;}

プロパティ値

この接続が開かれているデータベースのスキーマを表す ULDatabaseSchema オブジェクトへの参照。

備考

このプロパティが有効なのは、接続が開かれている間だけです。

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「ULDatabaseSchema クラス」 265 ページ

ServerVersion プロパティ

この機能は Ultra Light.NET ではサポートされていません。

構文

Visual Basic
Public Overrides Readonly Property **ServerVersion** As String

C#
public override string **ServerVersion** { get;}

プロパティ値

値は常に空の文字列です。

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ

State プロパティ

接続の現在のステータスを返します。

構文

Visual Basic
Public Overrides Readonly Property **State** As ConnectionState

C#
public override ConnectionState **State** { get;}

プロパティ値

接続が開いている場合は System.Data.ConnectionState.Open、閉じている場合は System.Data.ConnectionState.Closed。

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「StateChange イベント」 187 ページ
- ConnectionState.Open
- ConnectionState.Closed

SyncParms プロパティ

UL 拡張： この接続の同期設定を指定します。

構文

Visual Basic
Public Readonly Property **SyncParms** As ULSyncParms

C#
public ULSyncParms **SyncParms** { get;}

プロパティ値

この接続によって同期に使用されるパラメータを表す ULSyncParms オブジェクトへの参照。パラメータを修正すると、この接続で行われる次の同期に反映されます。

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「Synchronize() メソッド」 182 ページ
- 「SyncResult プロパティ」 153 ページ
- 「ULSyncParms クラス」 514 ページ

SyncResult プロパティ

UL 拡張：この接続の最後の同期結果を返します。

構文

Visual Basic
Public Readonly Property **SyncResult** As ULSyncResult

C#
public ULSyncResult **SyncResult** { get;}

プロパティ値

この接続の最後の同期結果を表す ULSyncResult オブジェクトへの参照。

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「Synchronize() メソッド」 182 ページ
- 「SyncParms プロパティ」 152 ページ
- 「SyncResult プロパティ」 153 ページ

BeginTransaction メソッド

データベース・トランザクションを起動します。

BeginTransaction() メソッド

トランザクション・オブジェクトを返します。トランザクション・オブジェクトに関連付けられているコマンドは、単一のトランザクションとして実行されます。トランザクションは、ULTransaction.Commit または ULTransaction.Rollback で終了します。

構文

Visual Basic
Public Function **BeginTransaction()** As ULTransaction

C#
public ULTransaction **BeginTransaction();**

戻り値

新しいトランザクションを表す ULTransaction オブジェクト。

備考

トランザクションは IsolationLevel.ReadCommitted で作成されます。

コマンドをトランザクション・オブジェクトに関連付けるには、ULCommand.Transaction プロパティを使用します。現在のトランザクションは、ULConnection.CreateCommand によって作成されたコマンドに自動的に関連付けられます。

デフォルトでは、接続はトランザクションを使用しません。また、すべてのコマンドは、実行されるときに自動的にコミットされます。現在のトランザクションがコミットまたはロールバックされると、次に BeginTransaction を呼び出すまで、接続はオートコミット・モードおよび以前の独立性レベルに戻ります。

Ultra Light での独立性レベルの定義は、ADO.NET のマニュアルの IsolationLevel の説明とは若干異なります。詳細については、「Ultra Light の独立性レベル」 『Ultra Light データベース管理とリファレンス』を参照してください。

これは、System.Data.IDbConnection.BeginTransaction() と System.Data.Common.DbConnection.BeginTransaction() が厳密に型指定されたものです。

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「BeginTransaction メソッド」 153 ページ
- 「BeginTransaction(IsolationLevel) メソッド」 154 ページ
- 「Commit メソッド」 588 ページ
- 「Rollback メソッド」 589 ページ
- 「Transaction プロパティ」 104 ページ
- 「CreateCommand メソッド」 160 ページ
- IDbConnection.BeginTransaction
- DbConnection.BeginTransaction

BeginTransaction(IsolationLevel) メソッド

指定された独立性レベルのトランザクション・オブジェクトを返します。トランザクション・オブジェクトに関連付けられているコマンドは、単一のトランザクションとして実行されます。トランザクションは、ULTransaction.Commit または ULTransaction.Rollback で終了します。

構文

Visual Basic

```
Public Function BeginTransaction( _
   ByVal isolationLevel As IsolationLevel _
) As ULTransaction
```

C#

```
public ULTransaction BeginTransaction(
   IsolationLevel isolationLevel
);
```


パラメータ

- **isolationLevel**　要求されたトランザクションの独立性レベル。Ultra Light.NET では、System.Data.IsolationLevel.ReadUncommitted および ReadCommitted のみがサポートされています。

戻り値

新しいトランザクションを表す ULTransaction オブジェクト。

備考

コマンドをトランザクション・オブジェクトに関連付けるには、ULCommand.Transaction プロパティを使用します。現在のトランザクションは、ULConnection.CreateCommand によって作成されたコマンドに自動的に関連付けられます。

デフォルトでは、接続はトランザクションを使用しません。また、すべてのコマンドは、実行されるときに自動的にコミットされます。現在のトランザクションがコミットまたはロールバックされると、次に BeginTransaction を呼び出すまで、接続はオートコミット・モードおよび以前の独立性レベルに戻ります。

Ultra Light での独立性レベルの定義は、ADO.NET のマニュアルの IsolationLevel の説明とは若干異なります。詳細については、「Ultra Light の独立性レベル」 『Ultra Light データベース管理とリファレンス』を参照してください。

これは、System.Data.IDbConnection.BeginTransaction(System.Data.IsolationLevel) と System.Data.Common.DbConnection.BeginTransaction(System.Data.IsolationLevel) が厳密に型指定されたものです。

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「BeginTransaction メソッド」 153 ページ
- 「ULTransaction クラス」 586 ページ
- 「BeginTransaction() メソッド」 153 ページ
- 「Commit メソッド」 588 ページ
- 「Rollback メソッド」 589 ページ
- 「Transaction プロパティ」 104 ページ
- 「CreateCommand メソッド」 160 ページ
- IDbConnection.BeginTransaction
- DbConnection.BeginTransaction
- IsolationLevel

CancelGetNotification メソッド

指定された名前に一致する、すべてのキューに登録されている保留中の取得通知コールをキャンセルします。

構文

Visual Basic
Public Function **CancelGetNotification(_**
  ByVal *queueName* As String _
) As Integer

C#
public int **CancelGetNotification(**
  string *queueName*
);

パラメータ

- **queueName**　キュー名と一致する式。

備考

指定された名前に一致する、すべてのキューに登録されている保留中の取得通知コールをキャンセルします。

影響を受けるキューの数を返します (ブロックされた読み込み数とは異なります)。

参照

- 「CreateNotificationQueue メソッド」 161 ページ
- 「DeclareEvent メソッド」 162 ページ
- 「DestroyNotificationQueue メソッド」 163 ページ
- 「DeclareEvent メソッド」 162 ページ
- 「GetNotification メソッド」 171 ページ
- 「RegisterForEvent メソッド」 178 ページ
- 「SendNotification メソッド」 180 ページ
- 「TriggerEvent メソッド」 184 ページ
- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「GetNotification メソッド」 171 ページ
- 「ULException クラス」 322 ページ

ChangeDatabase メソッド

開いている ULConnection の現在のデータベースを変更します。

構文

Visual Basic
Public Overrides Sub **ChangeDatabase(_**
  ByVal *connectionString* As String _
)

C#
public override void **ChangeDatabase(**
  string *connectionString*
);

パラメータ

- **connectionString**　新しいデータベースへの接続を開く完全な接続文字列。

備考

パラメータのエラーがあった場合でも、現在のデータベースへの接続は閉じられます。

UL 拡張：*connectionString* は、dbn または dbf ではなく、完全な接続文字列です。

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「ConnectionString プロパティ」 146 ページ

ChangeEncryptionKey メソッド

UL 拡張：データベースの暗号化キーを、指定された新しいキーに変更します。

構文

Visual Basic

```
Public Sub ChangeEncryptionKey( _
  ByVal newKey As String _
)
```

C#

```
public void ChangeEncryptionKey(
  string  newKey
);
```

パラメータ

- **newkey**　データベースの新しい暗号化キー。

備考

暗号化キーが失われた場合は、データベースを開くことができません。

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「EncryptionKey プロパティ」 198 ページ

ChangePassword メソッド

接続文字列に示されるユーザのパスワードを、指定される新しいパスワードに変更します。

構文

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

C#
```
public static void ChangePassword(
  string  connectionString,
  string  newPassword
);
```

パラメータ

- **connectionString**　目的のデータベースに接続できるだけの情報が含まれる接続文字列。接続文字列には、ユーザ ID と現在のパスワードが含まれる可能性があります。
- **newPassword**　設定する新しいパスワード。

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ

Close メソッド

データベース接続を閉じます。

構文

Visual Basic
```
Public Overrides Sub Close()
```

C#
```
public override void Close();
```

備考

Close メソッドは、保留中のトランザクションをロールバックし、その接続を閉じます。アプリケーションは、Close を複数回呼び出すことができます。

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ

CountUploadRows メソッド

UL 拡張：次回の同期でアップロードする必要のあるロー数を返します。

CountUploadRows(String, UInt32) メソッド

UL 拡張：次回の同期でアップロードする必要のあるロー数を返します。

構文

Visual Basic

```
Public Function CountUploadRows( _
   ByVal pubs As String, _
   ByVal threshold As UInt32 _
) As UInt32
```

C#

```
public uint CountUploadRows(
   string  pubs,
   uint threshold
);
```

パラメータ

- **pubs**　ローをチェックするパブリケーションのカンマ区切りのリスト。
- **threshold**　カウントするローの最大数。CountUploadRows の所要時間を制限します。値 0 は制限が最大であることを示します。値 1 は、同期の必要なローがあるかどうかを判別する場合に使用します。

戻り値

指定されたパブリケーションからアップロードする必要のあるロー数。

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「CountUploadRows メソッド」 158 ページ
- 「CountUploadRows(String, Int64) メソッド」 159 ページ

CountUploadRows(String, Int64) メソッド

UL 拡張：次回の同期でアップロードする必要のあるロー数を返します。

構文

Visual Basic

```
Public Function CountUploadRows( _
   ByVal pubs As String, _
   ByVal threshold As Long _
) As Long
```

C#

```
public long CountUploadRows(
   string  pubs,
   long threshold
);
```

パラメータ

- **pubs**　ローをチェックするパブリケーションのカンマ区切りのリスト。
- **threshold**　カウントするローの最大数。CountUploadRows の所要時間を制限します。値 0 は制限が最大であることを示します。値 1 は、同期の必要なローがあるかどうかを判別する場合に使用します。スレッショルド値は、[0,0x0fffffff] の範囲内であることが必要です。

戻り値

指定されたパブリケーションからアップロードする必要のあるロー数。

備考

このメソッドは、System.UInt32 型をネイティブでサポートしていない言語を対象として用意されています。この型がアプリケーションでサポートされている場合は、このメソッドの別のフォームを使用します。

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「CountUploadRows メソッド」 158 ページ
- 「CountUploadRows(String, UInt32) メソッド」 159 ページ

CreateCommand メソッド

この接続と現在のトランザクションに関連付けられる ULCommand オブジェクトを作成して初期化します。ULCommand のプロパティを使用して、その動作を制御できます。

構文

Visual Basic
Public Function **CreateCommand()** As ULCommand

C#
public ULCommand **CreateCommand();**

戻り値

新しい ULCommand オブジェクト。

備考

ULCommand.CommandText を設定してから、コマンドを実行する必要があります。

これは、System.Data.IDbConnection.CreateCommand と System.Data.Common.DbConnection.CreateCommand() が厳密に型指定されたものです。

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「ULCommand クラス」 91 ページ
- 「CommandText プロパティ」 99 ページ
- IDbConnection.CreateCommand
- DbConnection.CreateCommand

CreateNotificationQueue メソッド

イベント・キューを作成します。

構文

Visual Basic

```
Public Sub CreateNotificationQueue( _
  ByVal queueName As String, _
  ByVal parameters As String _
)
```

C#

```
public void CreateNotificationQueue(
  string  queueName,
  string  parameters
);
```

パラメータ

- **queueName**　新しいキューの名前。
- **parameters**　作成パラメータ。現在は使われず、NULL に設定されています。

備考

この接続のイベント通知キューを作成します。キュー名は、接続ごとにスコープされるため、別々の接続で同じ名前を持つキューを作成できます。イベント通知が送信されると、データベース内で一致する名前を持つすべてのキューが、個別のインスタンスの通知を受け取ります。名前では大文字と小文字が区別されません。RegisterForEvent() を呼び出したときに、キューが指定されていない場合は、接続ごとに要求に応じてデフォルトのキューが作成されます。

参照

- 「CancelGetNotification メソッド」 155 ページ
- 「DeclareEvent メソッド」 162 ページ
- 「DestroyNotificationQueue メソッド」 163 ページ
- 「DeclareEvent メソッド」 162 ページ
- 「GetNotification メソッド」 171 ページ
- 「RegisterForEvent メソッド」 178 ページ
- 「SendNotification メソッド」 180 ページ
- 「TriggerEvent メソッド」 184 ページ
- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「DestroyNotificationQueue メソッド」 163 ページ
- 「ULException クラス」 322 ページ

DeclareEvent メソッド

指定されたイベントを宣言します。

構文

Visual Basic

```
Public Sub DeclareEvent( _
  ByVal eventName As String _
)
```

C#

```
public void DeclareEvent(
  string  eventName
);
```

パラメータ

- **eventName**　イベントの名前。

備考

登録およびトリガされるイベントを宣言します。Ultra Light では、データベースまたは環境での操作によってトリガされるシステム・イベントの一部が事前に定義されています。イベント名は、ユニークにする必要があります。名前では大文字と小文字が区別されません。名前がすでに使用されているか無効な場合は、エラーをスローします。

参照

- 「CancelGetNotification メソッド」 155 ページ
- 「CreateNotificationQueue メソッド」 161 ページ
- 「DestroyNotificationQueue メソッド」 163 ページ
- 「DeclareEvent メソッド」 162 ページ
- 「GetNotification メソッド」 171 ページ
- 「RegisterForEvent メソッド」 178 ページ
- 「SendNotification メソッド」 180 ページ
- 「TriggerEvent メソッド」 184 ページ
- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「CreateNotificationQueue メソッド」 161 ページ
- 「ULException クラス」 322 ページ

DestroyNotificationQueue メソッド

イベント・キューを破棄します。

構文

Visual Basic

```
Public Sub DestroyNotificationQueue( _
  ByVal queueName As String _
)
```

C#

```
public void DestroyNotificationQueue(
  string  queueName
);
```

パラメータ

- **queueName**　キューの名前。

備考

指定されたイベント通知キューを破棄します。キュー内に未読の通知が残っている場合は、警告が通知されます。未読の通知は破棄されます。接続のデフォルトのイベント・キューが作成されている場合、接続が閉じると破棄されます。

参照

- 「CancelGetNotification メソッド」 155 ページ
- 「CreateNotificationQueue メソッド」 161 ページ
- 「DeclareEvent メソッド」 162 ページ
- 「GetNotification メソッド」 171 ページ
- 「RegisterForEvent メソッド」 178 ページ
- 「SendNotification メソッド」 180 ページ
- 「TriggerEvent メソッド」 184 ページ
- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「CreateNotificationQueue メソッド」 161 ページ
- 「ULException クラス」 322 ページ

ExecuteNextSQLPassthroughScript メソッド

保留中の次の SQL パススルー・スクリプトを実行します。

構文

Visual Basic
Public Sub **ExecuteNextSQLPassthroughScript()**

C#
public void **ExecuteNextSQLPassthroughScript();**

備考

同期を実行すると、複数の SQL パススルー・スクリプトがダウンロードされることがあります。

参照

- 「ExecuteSQLPassthroughScripts メソッド」 164 ページ
- 「GetSQLPassthroughScriptCount メソッド」 173 ページ
- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「GetSQLPassthroughScriptCount メソッド」 173 ページ
- 「ExecuteSQLPassthroughScripts メソッド」 164 ページ
- 「ULException クラス」 322 ページ

ExecuteSQLPassthroughScripts メソッド

保留中のすべての SQL パススルー・スクリプトを実行します。

構文

Visual Basic
Public Sub **ExecuteSQLPassthroughScripts()**

C#
public void **ExecuteSQLPassthroughScripts();**

備考

同期を実行すると、複数の SQL パススルー・スクリプトがダウンロードされることがあります。このメソッドは、保留中のすべてのスクリプトの実行に使用されます。スクリプトの進行状況は、ULDatabaseManager.SetGlobalListener() メソッドで追跡できます。

参照

- 「ExecuteNextSQLPassthroughScript メソッド」 164 ページ
- 「GetSQLPassthroughScriptCount メソッド」 173 ページ
- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「GetSQLPassthroughScriptCount メソッド」 173 ページ
- 「ExecuteNextSQLPassthroughScript メソッド」 164 ページ
- 「ULException クラス」 322 ページ
- 「SetGlobalListener メソッド」 261 ページ

ExecuteTable メソッド

UL 拡張：直接の操作用に、ULTable にデータベース・テーブルを取り出します。テーブルのプライマリ・キーを使用して、テーブルが開かれます (ソートされます)。

ExecuteTable(String) メソッド

UL 拡張：直接の操作用に、ULTable にデータベース・テーブルを取り出します。テーブルのプライマリ・キーを使用して、テーブルが開かれます (ソートされます)。

構文

Visual Basic
Public Function **ExecuteTable(_**
  ByVal *tableName* As String _
) As ULTable

C#
public ULTable **ExecuteTable(**
  string *tableName*
);

パラメータ

- **tableName**　開くテーブルの名前。

戻り値

ULTable オブジェクトとして返されるテーブル。

備考

このメソッドは、ULCommand インスタンスを必要としない ULCommand.ExecuteTable() メソッドのショートカットです。また、以前のバージョンの Ultra Light.NET から移行できるようにするために用意されています (iAnywhere.UltraLite.Connection.GetTable() と iAnywhere.UltraLite.Table.Open() が置き換えられます)。

例

次のコードでは、テーブルのプライマリ・キーを使用して MyTable というテーブルが開かれます。また、conn という開いた ULConnection インスタンスが想定されています。

```
' Visual Basic
Dim t As ULTable = conn.ExecuteTable("MyTable")
' The line above is equivalent to
' Dim cmd As ULCommand = conn.CreateCommand()
' cmd.CommandText = "MyTable"
' cmd.CommandType = CommandType.TableDirect
' Dim t As ULTable = cmd.ExecuteTable()
' cmd.Dispose()


// C#
ULTable t = conn.ExecuteTable("MyTable");
// The line above is equivalent to
// ULTable t;
// using(ULCommand cmd = conn.CreateCommand())
// {
//     cmd.CommandText = "MyTable";
//     cmd.CommandType = CommandType.TableDirect;
//     t = cmd.ExecuteTable();
// }
```

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「ExecuteTable メソッド」 165 ページ
- 「ExecuteTable(String, String) メソッド」 166 ページ
- 「ExecuteTable(String, String, CommandBehavior) メソッド」 168 ページ
- 「ULTable クラス」 548 ページ
- 「ExecuteTable() メソッド」 125 ページ
- 「ULCommand クラス」 91 ページ

ExecuteTable(String, String) メソッド

UL 拡張：直接の操作用に、ULTable にデータベース・テーブルを取り出します。テーブルは、指定されたインデックスを使用して開かれます (ソートされます)。

構文

Visual Basic

```
Public Function ExecuteTable( _
  ByVal tableName As String, _
```



```
  ByVal indexName As String _
) As ULTable

C#
public ULTable ExecuteTable(
  string  tableName,
  string  indexName
);
```

パラメータ

- **tableName**　開くテーブルの名前。
- **indexName**　テーブルを開く (ソートする) インデックスの名前。

戻り値

ULTable オブジェクトとして返されるテーブル。

備考

このメソッドは、ULCommand インスタンスを必要としない ULCommand.ExecuteTable() メソッドのショートカットです。また、以前のバージョンの Ultra Light.NET から移行できるようにするために用意されています (iAnywhere.UltraLite.Connection.GetTable() と iAnywhere.UltraLite.Table.Open() が置き換えられます)。

例

次のコードでは、MyIndex というインデックスを使用して MyTable というテーブルが開かれます。また、conn という開いた ULConnection インスタンスが想定されています。

```
' Visual Basic
Dim t As ULTable = conn.ExecuteTable("MyTable", "MyIndex")
' The line above is equivalent to
' Dim cmd As ULCommand = conn.CreateCommand()
' cmd.CommandText = "MyTable"
' cmd.IndexName = "MyIndex"
' cmd.CommandType = CommandType.TableDirect
' Dim t As ULTable = cmd.ExecuteTable()
' cmd.Dispose()


// C#
ULTable t = conn.ExecuteTable("MyTable", "MyIndex");
// The line above is equivalent to
// ULTable t;
// using(ULCommand cmd = conn.CreateCommand())
// {
//     cmd.CommandText = "MyTable";
//     cmd.IndexName = "MyIndex";
//     cmd.CommandType = CommandType.TableDirect;
//     t = cmd.ExecuteTable();
// }
```

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「ExecuteTable メソッド」 165 ページ
- 「ExecuteTable(String) メソッド」 165 ページ
- 「ExecuteTable(String, String, CommandBehavior) メソッド」 168 ページ
- 「ULTable クラス」 548 ページ
- 「ExecuteTable() メソッド」 125 ページ
- 「ULCommand クラス」 91 ページ

ExecuteTable(String, String, CommandBehavior) メソッド

UL 拡張：直接の操作用に、コマンド動作を指定してデータベース・テーブルを取り出します。テーブルは、指定されたインデックスを使用して開かれます (ソートされます)。

構文

Visual Basic

```
Public Function ExecuteTable( _
   ByVal tableName As String, _
   ByVal indexName As String, _
   ByVal cmdBehavior As CommandBehavior _
) As ULTable
```

C#

```
public ULTable ExecuteTable(
   string  tableName,
   string  indexName,
   CommandBehavior cmdBehavior
);
```

パラメータ

- **tableName**　開くテーブルの名前。
- **indexName**　テーブルを開く (ソートする) インデックスの名前。
- **cmdBehavior**　クエリの結果の記述と、接続への影響の記述の System.Data.CommandBehavior フラグのビット単位の組み合わせ。Ultra Light.NET で有効なのは、System.Data.CommandBehavior.Default フラグ、System.Data.CommandBehavior.CloseConnection フラグ、System.Data.CommandBehavior.SchemaOnly フラグだけです。

戻り値

ULTable オブジェクトとして返されるテーブル。

備考

このメソッドは、ULCommand インスタンスを必要としない ULCommand.ExecuteTable(System.Data.CommandBehavior) のショートカットです。また、以前のバージョンの Ultra Light.NET から移行できるようにするために用意されています

(iAnywhere.UltraLite.Connection.GetTable() と iAnywhere.UltraLite.Table.Open() が置き換えられます)。

例

次のコードでは、MyIndex というインデックスを使用して MyTable というテーブルが開かれます。また、conn という開いた ULConnection インスタンスが想定されています。

```
' Visual Basic
Dim t As ULTable = conn.ExecuteTable( _
    "MyTable", "MyIndex", CommandBehavior.Default _
  )
' The line above is equivalent to
' Dim cmd As ULCommand = conn.CreateCommand()
' cmd.CommandText = "MyTable"
' cmd.IndexName = "MyIndex"
' cmd.CommandType = CommandType.TableDirect
' Dim t As ULTable = cmd.ExecuteTable(CommandBehavior.Default)
' cmd.Dispose()


// C#
ULTable t = conn.ExecuteTable(
    "MyTable", "MyIndex", CommandBehavior.Default
  );
// The line above is equivalent to
// ULTable t;
// using(ULCommand cmd = conn.CreateCommand())
// {
//     cmd.CommandText = "MyTable";
//     cmd.IndexName = "MyIndex";
//     cmd.CommandType = CommandType.TableDirect;
//     t = cmd.ExecuteTable(CommandBehavior.Default);
// }
```

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「ExecuteTable メソッド」 165 ページ
- 「ExecuteTable(String) メソッド」 165 ページ
- 「ExecuteTable(String, String) メソッド」 166 ページ
- 「ULTable クラス」 548 ページ
- 「ExecuteTable(CommandBehavior) メソッド」 126 ページ
- 「ULCommand クラス」 91 ページ
- CommandBehavior
- CommandBehavior.Default
- CommandBehavior.CloseConnection
- CommandBehavior.SchemaOnly

GetLastDownloadTime メソッド

UL 拡張：指定されたパブリケーションの最後のダウンロードの時刻を返します。

構文

Visual Basic
Public Function **GetLastDownloadTime(_**
  ByVal *publication* As String _
) As Date

C#
public DateTime **GetLastDownloadTime(**
  string *publication*
);

パラメータ

- **publication** チェックするパブリケーション。詳細については、「ULPublicationSchema クラス」 412 ページを参照してください。

戻り値

最後のダウンロードのタイムスタンプ。

備考

publication パラメータは、チェックするパブリケーションの名前です。特殊な定数 ULPublicationSchema.SYNC_ALL_DB を使用した場合、データベース全体を最後にダウンロードした時刻を返します。

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「ResetLastDownloadTime メソッド」 179 ページ
- 「SYNC_ALL_DB フィールド」 413 ページ

GetNewUUID メソッド

UL 拡張：新しい UUID (System.Guid) を生成します。

構文

Visual Basic
Public Function **GetNewUUID()** As Guid

C#
public Guid **GetNewUUID();**

戻り値

System.Guid として返される新しい UUID。

備考

このメソッドは、.NET Compact Framework に含まれていないため、ここで提供されています。

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- Guid

GetNotification メソッド

通知またはタイムアウトをブロックします。イベント名または NULL を返します。

構文

Visual Basic

```
Public Function GetNotification( _
   ByVal queueName As String, _
   ByVal wait_ms As Integer _
) As String
```

C#

```
public string  GetNotification(
   string  queueName,
   int wait_ms
);
```

パラメータ

- **queueName**　待機するキューの名前。
- **wait_ms**　待機時間 (ミリ秒)。待機時間を無限に設定するには、System.Threading.Timeout.Infinite (-1) を使用します。

備考

イベント通知を読み込みます。この呼び出しは、通知が受信されるまで、または指定された待機時間が経過するまでブロックします。無限に待機する場合は、¥p wait_ms に System.Threading.Timeout.Infinite を渡します。待機をキャンセルするには、指定したキューに別の通知を送信するか、CancelGetNotification() を使用します。通知を読み込んだら、ReadNotificationParameter() を使用して追加のパラメータを取得します。

待機時間の期限が切れるか、待機がキャンセルされると、NULL を返し、それ以外の場合はイベント名を返します。

参照

- 「CancelGetNotification メソッド」 155 ページ
- 「CreateNotificationQueue メソッド」 161 ページ
- 「DeclareEvent メソッド」 162 ページ
- 「DestroyNotificationQueue メソッド」 163 ページ
- 「DeclareEvent メソッド」 162 ページ
- 「RegisterForEvent メソッド」 178 ページ
- 「SendNotification メソッド」 180 ページ
- 「TriggerEvent メソッド」 184 ページ
- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「SendNotification メソッド」 180 ページ
- 「GetNotificationParameter メソッド」 172 ページ
- 「CancelGetNotification メソッド」 155 ページ
- 「ULException クラス」 322 ページ

GetNotificationParameter メソッド

GetNotification() によって読み込まれたイベントのパラメータ値を取得します。

構文

Visual Basic

```
Public Function GetNotificationParameter( _
   ByVal queueName As String, _
   ByVal parameterName As String _
) As String
```

C#

```
public string  GetNotificationParameter(
   string  queueName,
   string  parameterName
);
```

パラメータ

- **queueName**　待機するキューの名前。
- **parameterName**　値を返すパラメータの名前。

備考

ULGetNotification() によって読み込まれたイベント通知のパラメータを取得します。指定されたキューでの最後に読み込まれた通知のパラメータのみ取得できます。

パラメータが見つかるとパラメータ値を返し、それ以外の場合は NULL を返します。

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「GetNotification メソッド」 171 ページ
- 「ULException クラス」 322 ページ

GetSQLPassthroughScriptCount メソッド

保留中の SQL パススルー・スクリプトの数を取得します。

構文

Visual Basic
Public Function **GetSQLPassthroughScriptCount()** As Long

C#
public long **GetSQLPassthroughScriptCount();**

備考

同期を実行すると、複数の SQL パススルー・スクリプトがダウンロードされることがあります。

参照

- 「ExecuteNextSQLPassthroughScript メソッド」 164 ページ
- 「ExecuteSQLPassthroughScripts メソッド」 164 ページ
- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「ExecuteNextSQLPassthroughScript メソッド」 164 ページ
- 「ExecuteSQLPassthroughScripts メソッド」 164 ページ
- 「ULException クラス」 322 ページ

GetSchema メソッド

この DbConnection のデータ・ソースに関するスキーマ情報を返します。

GetSchema() メソッド

サポートされているスキーマ・コレクションのリストを返します。

構文

Visual Basic
Public Overrides Function **GetSchema()** As DataTable

C#
public override DataTable **GetSchema();**

備考

使用できるメタデータの詳細については、「GetSchema(String, String[]) メソッド」 174 ページを参照してください。

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「GetSchema メソッド」 173 ページ

GetSchema(String) メソッド

この ULConnection について指定されたメタデータ・コレクションに関する情報を返します。

構文

Visual Basic

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

C#

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

パラメータ

- **collection**　メタデータ・コレクションの名前。指定しない場合、MetaDataCollections が使用されます。

備考

使用できるメタデータの詳細については、「GetSchema(String, String[]) メソッド」 174 ページを参照してください。

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「GetSchema メソッド」 173 ページ
- 「ULConnection クラス」 139 ページ

GetSchema(String, String[]) メソッド

この ULConnection のデータ・ソースのスキーマ情報を返します。このとき、文字列が指定されている場合はスキーマ名として使用し、文字列配列が指定されている場合は制限値として使用します。

構文

Visual Basic

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

C#

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

パラメータ

- **collection**　メタデータ・コレクションの名前。指定しない場合、MetaDataCollections が使用されます。
- **restrictions**　要求されるスキーマの制限値のセット。

戻り値

スキーマ情報が格納されている DataTable。

備考

このメソッドを使用すると、データベースに各種のメタデータを問い合わせることができます。メタデータの各型にはコレクション名が指定されており、そのデータを受け取るにはコレクション名を渡す必要があります。デフォルトのコレクション名は MetaDataCollections です。

引数を指定せずに、または **MetaDataCollections** というスキーマ・コレクション名を指定して GetSchema メソッドを呼び出すことによって、.NET データ・プロバイダに問い合わせをして、サポートされているスキーマ・コレクションのリストを判断できます。これによって、サポートされているスキーマ・コレクション (CollectionName)、それぞれがサポートする制限の数 (NumberOfRestrictions)、使用する識別子部分の数のリストから成る DataTable が返されます。

コレクション	メタデータ
Columns	データベース内のすべてのカラムに関する情報を返します。
DataSourceInformation	データベース・プロバイダに関する情報を返します。
DataType	サポートされているデータ型のリストを返します。
ForeignKeys	データベース内のすべての外部キーに関する情報を返します。
IndexColumns	データベース内のすべてのインデックス・カラムに関する情報を返します。
Indexes	データベース内のすべてのインデックスに関する情報を返します。
MetaDataCollections	すべてのコレクション名のリストを返します。

コレクション	メタデータ
Publications	データベース内のすべてのパブリケーションに関する情報を返します。
ReservedWords	Ultra Light が使用する予約語のリストを返します。
Restrictions	GetSchema で使用される制限に関する情報を返します。
Tables	データベース内のすべてのテーブルに関する情報を返します。

これらのコレクション名は、ULMetaDataCollectionNames クラスの読み込み専用プロパティとしても使用できます。

返される結果は、GetSchema への呼び出しの中で制限の配列を指定することによってフィルタできます。

各コレクションで有効な制限は、次の文を呼び出すことで問い合わせることができます。

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

コレクションが 4 つの制限を必要とする場合、制限パラメータは、NULL、または 4 つの値から成る文字列です。

特定の制限をフィルタするには、フィルタする文字列を配列内の所定の位置に指定し、使用しない位置には NULL を指定します。たとえば、Tables コレクションには、Table、TableType、SyncType という 3 つの制限があります。

Table コレクションをフィルタするには、次の手順に従います。

GetSchema("Tables", new string[ ] { "my_table", NULL, NULL })：my_table という名前のすべてのテーブルに関する情報を返します。

GetSchema("Tables", new string[ ] { NULL, "User", NULL })：すべてのユーザ・テーブルに関する情報を返します。

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「GetSchema メソッド」 173 ページ
- 「ULConnection クラス」 139 ページ
- 「ULMetaDataCollectionNames クラス」 366 ページ

GrantConnectTo メソッド

UL 拡張：パスワードを指定して、特定のユーザ ID に Ultra Light データベースへのアクセスを許可します。

構文

Visual Basic
```
Public Sub GrantConnectTo( _
  ByVal uid As String, _
  ByVal pwd As String _
)
```

C#
```
public void GrantConnectTo(
  string  uid,
  string  pwd
);
```

パラメータ

- **uid**　データベースへのアクセス権を受け取るユーザ ID。ユーザ ID の最大長は 16 文字です。
- **pwd**　ユーザ ID に関連付けられるパスワード。最大長は 16 文字です。

備考

既存のユーザ ID が指定されていれば、この関数を使用してそのユーザのパスワードを更新します。Ultra Light では、最大で 4 人のユーザがサポートされます。このメソッドが有効になるのは、接続が開かれたときにユーザ認証が有効になっていた場合だけです。

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「UserID プロパティ」 199 ページ
- 「Password プロパティ」 199 ページ
- 「ConnectionString プロパティ」 146 ページ

Open メソッド

以前に指定された接続文字列を使用してデータベースへの接続を開きます。

構文

Visual Basic
```
Public Overrides Sub Open()
```

C#
```
public override void Open();
```

備考

接続は、使用し終わったら明示的に閉じるか、破棄する必要があります。

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「ConnectionString プロパティ」 146 ページ
- 「State プロパティ」 152 ページ

RegisterForEvent メソッド

オブジェクトからイベントを取得するためのキューを登録します。

構文

Visual Basic

```
Public Sub RegisterForEvent( _
   ByVal eventName As String, _
   ByVal objectName As String, _
   ByVal queueName As String, _
   ByVal registerNotUnReg As Boolean _
)
```

C#

```
public void RegisterForEvent(
   string  eventName,
   string  objectName,
   string  queueName,
   bool registerNotUnReg
);
```

パラメータ

- **eventName**　イベントの名前。
- **objectName**　イベントを適用するオブジェクトの名前。たとえば、テーブル名です。
- **queueName**　使用するイベント・キューの名前。
- **registerNotUnReg**　登録する場合は true、登録解除する場合は false。

備考

イベントの通知を受け取るためのキューを登録します。キュー名が指定されていない場合は、デフォルトの接続キューが暗黙で指定され、必要に応じて作成されます。特定のシステム・イベントでは、そのイベントが適用されるオブジェクト名を指定できます。たとえば、TableModified イベントではテーブル名を指定できます。SendNotification() とは異なり、登録された特定のキューのみイベントの通知を受信します。別の接続の同じ名前を持つ他のキューは、明示的に登録されている場合を除き、イベントの通知を受信しません。キューまたはイベントが存在しない場合は、エラーをスローします。

参照

- 「CancelGetNotification メソッド」 155 ページ
- 「CreateNotificationQueue メソッド」 161 ページ
- 「DeclareEvent メソッド」 162 ページ
- 「DestroyNotificationQueue メソッド」 163 ページ
- 「DeclareEvent メソッド」 162 ページ
- 「GetNotification メソッド」 171 ページ
- 「SendNotification メソッド」 180 ページ
- 「TriggerEvent メソッド」 184 ページ
- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「DeclareEvent メソッド」 162 ページ
- 「CreateNotificationQueue メソッド」 161 ページ
- 「ULException クラス」 322 ページ

ResetLastDownloadTime メソッド

UL 拡張：最後のダウンロードの時刻をリセットします。

構文

Visual Basic

```
Public Sub ResetLastDownloadTime( _
  ByVal pubs As String _
)
```

C#

```
public void ResetLastDownloadTime(
  string  pubs
);
```

パラメータ

- **pubs**　リセットするパブリケーションのセット。詳細については、「ULPublicationSchema クラス」 412 ページを参照してください。

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「GetLastDownloadTime メソッド」 169 ページ

RevokeConnectFrom メソッド

UL 拡張：指定されたユーザ ID から Ultra Light データベースへのアクセス権を取り消します。

構文

Visual Basic
```
Public Sub RevokeConnectFrom( _
  ByVal uid As String _
)
```

C#
```
public void RevokeConnectFrom(
  string  uid
);
```

パラメータ

- **uid**　データベースへのアクセス権を取り消すユーザ ID。

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「GrantConnectTo メソッド」 176 ページ

RollbackPartialDownload メソッド

UL 拡張：部分的なダウンロードからの、データベースへの未処理の変更をロールバックします。

構文

Visual Basic
```
Public Sub RollbackPartialDownload()
```

C#
```
public void RollbackPartialDownload();
```

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「KeepPartialDownload プロパティ」 518 ページ
- 「ResumePartialDownload プロパティ」 522 ページ

SendNotification メソッド

一致するキューに通知を送信します。一致するキューの数を返します。

構文

Visual Basic
```
Public Function SendNotification( _
  ByVal queueName As String, _
  ByVal eventName As String, _
```

```
  ByVal parameters As String _
) As Integer

C#
public int SendNotification(
  string  queueName,
  string  eventName,
  string  parameters
);
```

パラメータ

- **queueName**　使用するイベント・キューの名前。
- **eventName**　イベントの名前。
- **parameters**　受け渡すパラメータ。

備考

指定された名前に一致するすべてのキュー (現在の接続におけるキューを含む) に通知を送信します。この呼び出しはブロックしません。特別なキュー名の "*" を使用すると、すべてのキューに送信します。

送信済み通知の数 (一致するキューの数) を返します。

参照

- 「CancelGetNotification メソッド」 155 ページ
- 「CreateNotificationQueue メソッド」 161 ページ
- 「DeclareEvent メソッド」 162 ページ
- 「DestroyNotificationQueue メソッド」 163 ページ
- 「DeclareEvent メソッド」 162 ページ
- 「GetNotification メソッド」 171 ページ
- 「RegisterForEvent メソッド」 178 ページ
- 「TriggerEvent メソッド」 184 ページ
- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「DeclareEvent メソッド」 162 ページ
- 「RegisterForEvent メソッド」 178 ページ
- 「ULException クラス」 322 ページ

StartSynchronizationDelete メソッド

UL 拡張：同期用に、この接続によって行われる以後のすべての削除にマークを付けます。

構文

```
Visual Basic
Public Sub StartSynchronizationDelete()

C#
public void StartSynchronizationDelete();
```

備考

この関数が呼び出されると、すべての削除操作が再度同期され、Ultra Light データベースから削除されたローが統合データベースからも削除されます。

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「StopSynchronizationDelete メソッド」 182 ページ
- 「Truncate メソッド」 569 ページ

StopSynchronizationDelete メソッド

UL 拡張：削除操作が同期されないようにします。

構文

Visual Basic
Public Sub **StopSynchronizationDelete()**

C#
public void **StopSynchronizationDelete();**

備考

領域を節約するために Ultra Light データベースの古い情報を削除して、統合データベースにはそれを残しておく場合に、このメソッドを使用すると便利です。

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「StartSynchronizationDelete メソッド」 181 ページ

Synchronize メソッド

UL 拡張：現在の ULConnection.SyncParms を使用してデータベースの同期をとります。

Synchronize() メソッド

UL 拡張：現在の ULConnection.SyncParms を使用してデータベースの同期をとります。

構文

Visual Basic
Public Sub **Synchronize()**

C#
public void **Synchronize();**

備考

結果の詳細なステータスは、この接続の ULConnection.SyncResult でレポートされます。

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「Synchronize メソッド」 182 ページ
- 「Synchronize(ULSyncProgressListener) メソッド」 183 ページ
- 「SyncParms プロパティ」 152 ページ
- 「SyncResult プロパティ」 153 ページ

Synchronize(ULSyncProgressListener) メソッド

UL 拡張：指定されたリスナに送信されるプログレス・イベントとともに、現在の ULConnection.SyncParms を使用してデータベースの同期をとります。

構文

Visual Basic

```
Public Sub Synchronize( _
  ByVal listener As ULSyncProgressListener _
)
```

C#

```
public void Synchronize(
  ULSyncProgressListener listener
);
```

パラメータ

- **listener**　同期プログレス・イベントを受信するオブジェクト。

備考

同期中のエラーは、ULSyncProgressState.STATE_ERROR イベントとして送信され、ULExceptions としてスローされます。

結果の詳細なステータスは、この接続の ULConnection.SyncResult でレポートされます。

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「Synchronize メソッド」 182 ページ
- 「ULSyncProgressListener インタフェース」 538 ページ
- 「Synchronize() メソッド」 182 ページ
- 「SyncParms プロパティ」 152 ページ
- 「ULSyncProgressState 列挙体」 540 ページ
- 「ULException クラス」 322 ページ
- 「SyncResult プロパティ」 153 ページ

TriggerEvent メソッド

イベントをトリガします。送信済みの通知の数を返します。

構文

Visual Basic
```
Public Function TriggerEvent( _
  ByVal eventName As String, _
  ByVal parameters As String _
) As Integer
```

C#
```
public int TriggerEvent(
  string  eventName,
  string  parameters
);
```

パラメータ

- **eventName**　トリガされるイベントの名前。
- **parameters**　受け渡すパラメータ。

備考

イベントをトリガして、登録されたすべてのキューに通知を送信します。

送信済みのイベント通知の数を返します。

参照

- 「CancelGetNotification メソッド」 155 ページ
- 「CreateNotificationQueue メソッド」 161 ページ
- 「DeclareEvent メソッド」 162 ページ
- 「DestroyNotificationQueue メソッド」 163 ページ
- 「DeclareEvent メソッド」 162 ページ
- 「GetNotification メソッド」 171 ページ
- 「RegisterForEvent メソッド」 178 ページ
- 「SendNotification メソッド」 180 ページ
- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「DeclareEvent メソッド」 162 ページ
- 「RegisterForEvent メソッド」 178 ページ
- 「ULException クラス」 322 ページ

ValidateDatabase メソッド

現在のデータベースの検証を実行します。

構文

Visual Basic
```
Public Sub ValidateDatabase( _
  ByVal how As ULDBValid, _
  ByVal tableName As String _
)
```

C#
```
public void ValidateDatabase(
  ULDBValid how,
  string  tableName
);
```

パラメータ

- **how**　データベースの検証方法。詳細については、「ULDBValid 列挙体」 321 ページを参照してください。
- **tableName**　NULL (Visual Basic の場合は Nothing) の場合はデータベース全体を検証し、それ以外の場合は指定されたテーブルだけを検証します。

例

次のコードでは、現在のデータベースを検証します。

```
' Visual Basic
conn.ValidateDatabase( iAnywhere.Data.UltraLite.ULVF_INDEX, Nothing )

// C#
conn.ValidateDatabase( iAnywhere.Data.UltraLite.ULVF_INDEX, null )
```

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「ValidateDatabase メソッド」 263 ページ
- 「ULDBValid 列挙体」 321 ページ

InfoMessage イベント

Ultra Light.NET が、この接続の警告または情報メッセージを送信するときに発生します。

構文

Visual Basic
```
Public Event InfoMessage As ULInfoMessageEventHandler
```

C#
```
public event ULInfoMessageEventHandler InfoMessage;
```

備考

Ultra Light.NET の警告または情報メッセージを処理するには、ULInfoMessageEventHandler デリゲートを作成し、このイベントにアタッチする必要があります。

例

次のコードでは、情報メッセージのイベント・ハンドラが定義されます。

```
' Visual Basic
Private Sub MyInfoMessageHandler( _
      obj As Object, args As ULInfoMessageEventArgs _
    )
  System.Console.WriteLine( _
      "InfoMesageHandler: " + args.NativeError + ", " _
      + args.Message _
    )
End Sub

// C#
private void MyInfoMessageHandler(
      object obj, ULInfoMessageEventArgs args
    )
{
  System.Console.WriteLine(
      "InfoMesageHandler: " + args.NativeError + ", "
      + args.Message
    );
}
```

次のコードでは、MyInfoMessageHandler が conn という接続に追加されます。

```
' Visual Basic
AddHandler conn.InfoMessage, AddressOf MyInfoMessageHandler

// C#
conn.InfoMessage +=
  new ULInfoMessageEventHandler(MyInfoMessageHandler);
```

イベント・データ

- **NativeError**　データベースによって返される情報メッセージまたは警告に対応する SQL コードです。
- **Message**　データベースによって返される情報メッセージまたは警告メッセージの文字列です。
- **Source**　メッセージを返す ADO.NET データ・プロバイダの名前です。

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- 「ULInfoMessageEventHandler デリゲート」 365 ページ

StateChange イベント

この接続のステータスが変更されると発生します。

構文

Visual Basic
Public Overrides Event **StateChange** As StateChangeEventHandler

C#
public event override StateChangeEventHandler **StateChange**;

備考

ステータス変更メッセージを処理するには、System.Data.StateChangeEventHandler デリゲートを作成し、このイベントにアタッチする必要があります。

例

次のコードでは、ステータス変更のイベント・ハンドラが定義されます。

```
' Visual Basic
Private Sub MyStateHandler( _
      obj As Object, args As StateChangeEventArgs _
    )
  System.Console.WriteLine( _
      "StateHandler: " + args.OriginalState + " to " _
      + args.CurrentState _
    )
End Sub

// C#
private void MyStateHandler(
      object obj, StateChangeEventArgs args
    )
{
  System.Console.WriteLine(
      "StateHandler: " + args.OriginalState + " to "
      + args.CurrentState
    );
}
```

次のコードでは、MyStateHandler が conn という接続に追加されます。

```
' Visual Basic
AddHandler conn.StateChange, AddressOf MyStateHandler

// C#
conn.StateChange += new StateChangeEventHandler(MyStateHandler);
```

イベント・データ

- **CurrentState**　接続の新しいステータスを取得します。イベントが発生すると、接続オブジェクトはすでに新しいステータスになっています。
- **OriginalState**　接続の元のステータスを取得します。

参照

- 「ULConnection クラス」 139 ページ
- 「ULConnection メンバ」 139 ページ
- StateChangeEventHandler

ULConnectionParms クラス

UL 拡張：Ultra Light データベースへの接続を開く接続文字列を作成します。頻繁に使用する接続パラメータは、ULConnectionParms オブジェクトの個々のプロパティです。

構文

Visual Basic
Public Class **ULConnectionParms**
  Inherits Component

C#
public class **ULConnectionParms**: Component

備考

ULConnectionParms オブジェクトは、接続を開くパラメータ (ULConnection.Open)、またはデータベースを削除するパラメータ (ULDatabaseManager.DropDatabase) を指定するために使用します。

前後のスペースは、すべての値で無視されます。値には、前後のスペースやセミコロン (;) を含めることはできません。また、一重引用符 (') または二重引用符 (") で始めることはできません。

接続文字列を作成するときは、データベースを識別し、オプションの接続設定を指定する必要があります。ULConnectionParms オブジェクトで適切なプロパティを設定して、接続パラメータをすべて指定したら、ULConnectionParms.ToString を使用して接続文字列を作成します。作成した文字列は、ULConnection(String) コンストラクタで新しい ULConnection を作成したり、既存の ULConnection オブジェクトの ULConnection.ConnectionString を設定したりするのに使用されます。

Identifying the database

各インスタンスには、データベースへのプラットフォーム固有のパスがあります。実行されているプラットフォームに対応する値だけが使用されます。たとえば、次のコードでは、パス ¥UltraLite¥mydb1.udb は Windows Mobile で使用され、mydb2.udb はその他のプラットフォームで使用されます。

```
' Visual Basic
Dim dbName As ULConnectionParms = new ULConnectionParms
dbName.DatabaseOnCE = "¥UltraLite¥mydb1.udb"
dbName.DatabaseOnDesktop = "somedir¥mydb2.udb"

// C#
ULConnectionParms dbName = new ULConnectionParms();
dbName.DatabaseOnCE = "¥¥UltraLite¥¥mydb1.udb";
dbName.DatabaseOnDesktop = @"somedir¥mydb2.udb";
```

Ultra Light データベース・ファイルに推奨されるファイル拡張子は .udb です。Windows Mobile デバイスでは、デフォルトのデータベースは ¥UltraLiteDB¥ulstore.udb です。その他の Windows プラットフォームでは、デフォルトのデータベースは ulstore.udb です。C# では、パス内の円記号をエスケープするか、アットマーク (@) で囲まれた文字列リテラルを使用する必要があります。

複数のデータベースを使用している場合は、各データベースのデータベース名を指定する必要があります。詳細については、「AdditionalParms プロパティ」 192 ページを参照してください。

Optional connection settings

アプリケーションでの必要性やデータベースの作成方法によっては、デフォルト以外の ULConnectionParms.UserID や ULConnectionParms.Password、データベースの ULConnectionParms.EncryptionKey、接続の ULConnectionParms.CacheSize を指定することが必要になる場合があります。複数の接続を使用するアプリケーションでは、各接続に対してユニークな ULConnectionParms.ConnectionName を指定する必要があります。

データベースは、1 人の認証済みユーザ DBA で作成されます。このユーザの最初のパスワードは sql です。デフォルトでは、ユーザ ID DBA とパスワード sql を使用して、接続が開かれます。デフォルトのユーザを無効にするには、ULConnection.RevokeConnectFrom を使用します。ユーザを追加したりユーザのパスワードを変更するには、ULConnection.GrantConnectTo を使用します。

データベースを作成したときに暗号化キーを指定した場合は、そのデータベースへの以後の接続では、すべて同じ暗号化キーを使用する必要があります。データベースの暗号化キーを変更するには、ULConnection.ChangeEncryptionKey を使用します。

詳細については、「Ultra Light 接続パラメータ」『Ultra Light データベース管理とリファレンス』を参照してください。

参照

- 「ULConnectionParms メンバ」 190 ページ
- 「Open メソッド」 177 ページ
- 「DropDatabase メソッド」 259 ページ
- 「ToString メソッド」 200 ページ
- 「ULConnection クラス」 139 ページ
- 「ULConnection(String) コンストラクタ」 144 ページ
- 「ConnectionString プロパティ」 146 ページ
- 「UserID プロパティ」 199 ページ
- 「Password プロパティ」 199 ページ
- 「EncryptionKey プロパティ」 198 ページ
- 「CacheSize プロパティ」 196 ページ
- 「ConnectionName プロパティ」 197 ページ
- 「RevokeConnectFrom メソッド」 179 ページ
- 「GrantConnectTo メソッド」 176 ページ
- 「ChangeEncryptionKey メソッド」 157 ページ

ULConnectionParms メンバ

パブリック・コンストラクタ

メンバ名	説明
「ULConnectionParms コンストラクタ」 192 ページ	ULConnectionParms インスタンスを、そのデフォルト値で初期化します。

パブリック・プロパティ

メンバ名	説明
「AdditionalParms プロパティ」 192 ページ	「名前=値」のペアをセミコロンで区切ったリストで、追加のパラメータを指定します。ここで指定されるのは、使用頻度の低いパラメータです。
「CacheSize プロパティ」 196 ページ	キャッシュのサイズを指定します。
「ConnectionName プロパティ」 197 ページ	接続の名前を指定します。接続名が必要となるのは、データベースとの接続を複数作成する場合だけです。
「DatabaseOnCE プロパティ」 197 ページ	Windows Mobile 上の Ultra Light データベースのパスとファイル名を指定します。
「DatabaseOnDesktop プロパティ」 197 ページ	Windows デスクトップ・プラットフォーム上の Ultra Light データベースのパスとファイル名を指定します。
「EncryptionKey プロパティ」 198 ページ	データベースを暗号化するためのキーを指定します。
「Password プロパティ」 199 ページ	認証済みユーザのパスワードを指定します。
「UserID プロパティ」 199 ページ	データベースで認証されるユーザを指定します。

パブリック・メソッド

メンバ名	説明
「ToString メソッド」 200 ページ	このインスタンスの文字列表現を返します。

パブリック・イベント

メンバ名	説明
「UnusedEvent イベント」 200 ページ	使われていません。

参照

- 「ULConnectionParms クラス」 189 ページ
- 「Open メソッド」 177 ページ
- 「DropDatabase メソッド」 259 ページ
- 「ToString メソッド」 200 ページ
- 「ULConnection クラス」 139 ページ
- 「ULConnection(String) コンストラクタ」 144 ページ
- 「ConnectionString プロパティ」 146 ページ
- 「UserID プロパティ」 199 ページ
- 「Password プロパティ」 199 ページ
- 「EncryptionKey プロパティ」 198 ページ
- 「CacheSize プロパティ」 196 ページ
- 「ConnectionName プロパティ」 197 ページ
- 「RevokeConnectFrom メソッド」 179 ページ
- 「GrantConnectTo メソッド」 176 ページ
- 「ChangeEncryptionKey メソッド」 157 ページ

ULConnectionParms コンストラクタ

ULConnectionParms インスタンスを、そのデフォルト値で初期化します。

構文

Visual Basic
Public Sub **New()**

C#
public **ULConnectionParms();**

参照

- 「ULConnectionParms クラス」 189 ページ
- 「ULConnectionParms メンバ」 190 ページ

AdditionalParms プロパティ

「名前=値」のペアをセミコロンで区切ったリストで、追加のパラメータを指定します。ここで指定されるのは、あまり一般的に使用されないパラメータです。

構文

Visual Basic
Public Property **AdditionalParms** As String

C#
public string **AdditionalParms** { get; set; }

プロパティ値

「キーワード=値」をセミコロンで区切ったリストによる追加のパラメータ。keyword=value リストの値は、ULConnection.ConnectionString のルールに従う必要があります。デフォルトは NULL 参照 (Visual Basic の Nothing) です。

備考

ページ・サイズや予約サイズのパラメータの値は、バイト単位で指定します。キロバイトの単位を示すにはサフィックス k または K を使用し、メガバイトの単位を示すにはサフィックス m または M を使用します。

追加のパラメータを次に示します。

キーワード	説明
dbn	接続する必要のある、ロードしたデータベースを識別します。 データベースが起動されると、データベース名が割り当てられます。データベース名は、dbn パラメータを使用して明示的に割り当てられるか、または拡張子とパスが削除されたベース・ファイル名を使用して Ultra Light によって割り当てられます。 接続が開かれると、Ultra Light は dbn が一致する実行中のデータベースをまず検索します。一致するデータベースが見つからない場合、Ultra Light は適切なデータベースのファイル名のパラメータ (DatabaseOnCE または DatabaseOnDesktop) を使用して、新しいデータベースを起動します。 このパラメータが必要なのは、同じベース・ファイル名を持つ 2 つの異なるデータベースにアプリケーション (または Ultra Light エンジン) がアクセスする場合です。 このパラメータが使用されるのは、ULConnection.Open を使用して接続を開いたときだけです。

キーワード	説明
ordered_table_scans	ORDER BY 句を指定しない SQL クエリで、デフォルトで順序付けされたテーブル・スキャンを実行するかどうかを指定します。 リリース 10.0.1 では、Ultra Light で動的 SQL を使用するとき、クエリ実行の順序は重要ではありません。Ultra Light は、プライマリ・キー・インデックスを使用するのではなく、データベース・ページからローに直接アクセスします。これにより、ローをフェッチするパフォーマンスが改善されました。この最適化を使用するには、クエリが読み込み専用であり、すべてのローをスキャンする必要があります。 ローが特定の順序に並べることを期待する場合は、SQL クエリに ORDER BY 文を指定します。ただし、一部のアプリケーションでは、プライマリ・キーの順にローを返すようなデフォルトの動作に依存する可能性があります。このような場合は、ordered_table_scans パラメータを 1 (true、yes、on) に設定し、テーブル上で反復するときに以前の動作に戻すようにすることが必要です。 ordered_table_scans を 1 (true、yes、on) に設定しても、ORDER BY 句を指定しなかったり、クエリがインデックスを利用できなかったりすると、Ultra Light はプライマリ・キーを使用してデフォルトの動作を実行します。 次のパラメータ文字列を使用すると、Ultra Light は以前のリリースと同じように動作します。 createParms.AdditionalParms = "ordered_table_scans=yes" デフォルトは 0 (false、off、no) です。 このパラメータが使用されるのは、ULConnection.Open を使用して接続を開いたときだけです。 詳細については、「Ultra Light 接続パラメータ」 『Ultra Light データベース管理とリファレンス』を参照してください。

キーワード	説明
reserve_size	Ultra Light の永続的データの保管に使用する、ファイル・システム領域を予約します。 reserve_size パラメータを使用すると、データを挿入することなく、Ultra Light データベースが必要とするファイル・システム領域を事前に割り付けることができます。ファイル・システムの領域を予約すると、パフォーマンスが多少向上し、メモリが不足するという障害を防ぐことができます。デフォルトでは、永続ストア・ファイルのサイズは、アプリケーションがデータベースを更新して、サイズを大きくする必要が生じた場合にだけ大きくなります。 reserve_size で予約されるファイル・システム領域には、未加工データだけでなく、永続ストア・ファイルのメタデータも含まれることに注意してください。データベースのデータ量から、必要なファイル・システム領域を計算する場合は、メタデータのオーバヘッドとデータの圧縮を考慮してください。テスト・データを入れてデータベースを実行し、永続ストア・ファイルのサイズを確認することをおすすめします。 reserve_size パラメータは、起動時に永続ストア・ファイルを設定された予約サイズまで大きくすることにより、領域を予約します。これは、そのファイルが以前に存在していたかどうかに関係なく行われます。このファイルはトランケートされません。 次のパラメータ文字列では、起動時に永続ストア・ファイルのサイズが最低 2 MB 確保されます。 createParms.AdditionalParms = "reserve_size=2m" このパラメータが使用されるのは、ULConnection.Open を使用して接続を開いたときだけです。
start	ロケーションを指定して、Ultra Light エンジンを開始します。 現在実行中でないエンジンに接続する場合は、StartLine (START) 接続パラメータだけを指定します。 ロケーションの指定は、Ultra Light エンジンがシステム パスに登録されていない場合にのみ必要です。 Ultra Light.NET で Ultra Light エンジンを使用する方法の詳細については、「RuntimeType プロパティ」 256 ページを参照してください。

参照

- 「ULConnectionParms クラス」 189 ページ
- 「ULConnectionParms メンバ」 190 ページ
- 「ConnectionString プロパティ」 146 ページ
- 「DatabaseOnCE プロパティ」 197 ページ
- 「DatabaseOnDesktop プロパティ」 197 ページ
- 「Open メソッド」 177 ページ

CacheSize プロパティ

キャッシュのサイズを指定します。

構文

Visual Basic
Public Property **CacheSize** As String

C#
public string **CacheSize** { get; set; }

プロパティ値

キャッシュ・サイズを指定する文字列。デフォルトは NULL 参照 (Visual Basic の Nothing) です。この場合、デフォルトの 16 ページが使用されます。

備考

キャッシュ・サイズの値は、バイト単位で指定します。キロバイトの単位を示すにはサフィックス k または K を使用し、メガバイトの単位を示すにはサフィックス m または M を使用します。

たとえば、次の例ではキャッシュ・サイズが 128 KB に設定されます。

```
connParms.CacheSize = "128k"
```

キャッシュ・サイズが未指定であるか、正しく指定されていない場合は、デフォルトのサイズが使用されます。デフォルトのキャッシュ・サイズは 16 ページです。デフォルトのページ・サイズは 4 KB なので、デフォルトのキャッシュ・サイズは 64 KB です。キャッシュの最小サイズは、プラットフォームによって異なります。

デフォルトのキャッシュ・サイズは小さめの値です。テスト結果により、パフォーマンスの向上が必要と判断した場合は、キャッシュ・サイズを大きくしてください。キャッシュ・サイズをデータベース自体のサイズよりも大きくすると、パフォーマンスは向上しません。また、キャッシュ・サイズが大きいと、その他の使用可能なアプリケーションの数が少なくなることがあります。

参照

- 「Ultra Light CACHE_SIZE 接続パラメータ」 『Ultra Light データベース管理とリファレンス』
- 「ULConnectionParms クラス」 189 ページ
- 「ULConnectionParms メンバ」 190 ページ

ConnectionName プロパティ

接続の名前を指定します。接続名が必要となるのは、データベースとの接続を複数作成する場合だけです。

構文

Visual Basic
Public Property **ConnectionName** As String

C#
public string **ConnectionName** { get; set; }

プロパティ値

接続の名前を指定する文字列。デフォルト値は NULL 参照 (Visual Basic の Nothing) です。

参照

- 「ULConnectionParms クラス」 189 ページ
- 「ULConnectionParms メンバ」 190 ページ

DatabaseOnCE プロパティ

Windows Mobile 上の Ultra Light データベースのパスとファイル名を指定します。

構文

Visual Basic
Public Property **DatabaseOnCE** As String

C#
public string **DatabaseOnCE** { get; set; }

プロパティ値

データベースへのフル・パスを指定する文字列。値が NULL 参照 (Visual Basic の Nothing) の場合、データベース ¥UltraLiteDB¥ulstore.udb が使用されます。C# では、パス内の円記号をエスケープするか、アットマーク (@) で囲まれた文字列リテラルを使用する必要があります。デフォルト値は NULL 参照 (Visual Basic の Nothing) です。

参照

- 「ULConnectionParms クラス」 189 ページ
- 「ULConnectionParms メンバ」 190 ページ

DatabaseOnDesktop プロパティ

Windows デスクトップ・プラットフォーム上の Ultra Light データベースのパスとファイル名を指定します。

構文

Visual Basic
Public Property **DatabaseOnDesktop** As String

C#
public string **DatabaseOnDesktop** { get; set; }

プロパティ値

データベースへの絶対パスまたは相対パスを指定する文字列。値が NULL 参照 (Visual Basic の Nothing) の場合、データベース ulstore.udb が使用されます。C# では、パス内の円記号をエスケープするか、アットマーク (@) で囲まれた文字列リテラルを使用する必要があります。デフォルト値は NULL 参照 (Visual Basic の Nothing) です。

参照

- 「ULConnectionParms クラス」 189 ページ
- 「ULConnectionParms メンバ」 190 ページ

EncryptionKey プロパティ

データベースを暗号化するためのキーを指定します。

構文

Visual Basic
Public Property **EncryptionKey** As String

C#
public string **EncryptionKey** { get; set; }

プロパティ値

暗号化キーを指定する文字列。デフォルト値は NULL 参照 (Visual Basic の Nothing) で、暗号化は行われません。

備考

すべての接続では、データベースが作成されたときに指定されたキーと同じキーを使用する必要があります。キーを忘れた場合はデータベースにまったくアクセスできなくなります。

すべてのパスワードに共通することですが、最善の方法は、簡単には推測できないキー値を選択することです。キーの長さは任意ですが、短いと推測されやすいため、一般的には長い方が適しています。数字、文字、特殊文字を組み合わせると、キーは推測されにくくなります。

参照

- 「ULConnectionParms クラス」 189 ページ
- 「ULConnectionParms メンバ」 190 ページ
- 「ChangeEncryptionKey メソッド」 157 ページ

Password プロパティ

認証済みユーザのパスワードを指定します。

構文

Visual Basic
Public Property **Password** As String

C#
public string **Password** { get; set; }

プロパティ値

データベースのユーザ ID を指定する文字列。デフォルトは NULL 参照 (Visual Basic の Nothing) です。

備考

パスワードでは大文字と小文字が区別されます。

データベースが作成されると、ユーザ ID DBA のパスワードは SQL に設定されます。

参照

- 「ULConnectionParms クラス」 189 ページ
- 「ULConnectionParms メンバ」 190 ページ
- 「UserID プロパティ」 199 ページ

UserID プロパティ

データベースで認証されるユーザを指定します。

構文

Visual Basic
Public Property **UserID** As String

C#
public string **UserID** { get; set; }

プロパティ値

データベースのユーザ ID を指定する文字列。デフォルト値は NULL 参照 (Visual Basic の Nothing) です。

備考

ユーザ ID の大文字と小文字は区別されません。

データベースは最初、DBA という名前の 1 人の認証済みユーザで作成されます。

ユーザ ID とパスワードを両方とも入力しないと、ユーザ名 DBA とパスワード SQL が使用されます。データベースをより強力に保護するには、ユーザ名 DBA のパスワードを変更するか、新

しいユーザを作成して (ULConnection.GrantConnectTo を使用)、DBA ユーザを削除します (ULConnection.RevokeConnectFrom を使用)。

参照

- 「ULConnectionParms クラス」 189 ページ
- 「ULConnectionParms メンバ」 190 ページ
- 「Password プロパティ」 199 ページ
- 「GrantConnectTo メソッド」 176 ページ
- 「RevokeConnectFrom メソッド」 179 ページ

ToString メソッド

このインスタンスの文字列表現を返します。

構文

Visual Basic
Public Overrides Function **ToString()** As String

C#
public override string **ToString();**

戻り値

「キーワード=値」の組み合わせがセミコロンで区切られたリスト形式の、このインスタンスの文字列表現。

参照

- 「ULConnectionParms クラス」 189 ページ
- 「ULConnectionParms メンバ」 190 ページ

UnusedEvent イベント

使われていません。

構文

Visual Basic
Public Event **UnusedEvent** As ULConnectionParms.UnusedEventHandler

C#
public event ULConnectionParms.UnusedEventHandler **UnusedEvent**;

備考

この public Event は、Visual Basic .NET プロジェクトでこのクラスを組み込む場合の Visual Studio .NET のバグを修正するために用意されています。実用的な用途はありません。

参照

- 「ULConnectionParms クラス」 189 ページ
- 「ULConnectionParms メンバ」 190 ページ

ULConnectionParms.UnusedEventHandler デリゲート

UL 拡張：使われていません。

構文

Visual Basic

```
Public Delegate Sub ULConnectionParms.UnusedEventHandler( _
  ByVal sender As Object, _
  ByVal args As EventArgs _
)
```

C#

```
public delegate void ULConnectionParms.UnusedEventHandler(
  object sender,
  EventArgs args
);
```

備考

この public Delegate は、Visual Basic .NET プロジェクトでこのクラスを組み込む場合の Visual Studio .NET のバグを修正するために用意されています。実用的な用途はありません。

ULConnectionStringBuilder クラス

Ultra Light データベースへの接続を開く接続文字列を作成します。頻繁に使用する接続パラメータは、ULConnectionStringBuilder オブジェクトの個々のプロパティです。このクラスは継承できません。

構文

Visual Basic
Public NotInheritable Class **ULConnectionStringBuilder**
  Inherits DbConnectionStringBuilder

C#
public sealed class **ULConnectionStringBuilder**: DbConnectionstring Builder

備考

制限：ULConnectionStringBuilder クラスは、.NET Compact Framework 2.0 では使用できません。

ULConnectionStringBuilder オブジェクトは、接続を開くパラメータ (ULConnection.Open)、またはデータベースを削除するパラメータ (ULDatabaseManager.DropDatabase) を指定するために使用します。

前後のスペースは、すべての値で無視されます。値には、前後のスペースやセミコロン (;) を含めることはできません。また、一重引用符 (') または二重引用符 (") で始めることはできません。

接続文字列を作成するときは、データベースを識別し、オプションの接続設定を指定する必要があります。ULConnectionStringBuilder オブジェクトで適切なプロパティを設定して、接続パラメータをすべて指定したら、System.Data.Common.DbConnectionStringBuilder.ConnectionString を使用して接続文字列を作成します。作成した文字列は、ULConnection(String) コンストラクタで新しい ULConnection を作成したり、既存の ULConnection オブジェクトの ULConnection.ConnectionString を設定したりするのに使用されます。

Identifying the database

各インスタンスには、データベースへのプラットフォーム固有のパスがあります。実行されているプラットフォームに対応する値だけが使用されます。たとえば、次のコードでは、パス ¥UltraLite¥mydb1.udb は Windows Mobile で使用され、mydb2.udb はその他のプラットフォームで使用されます。

```
' Visual Basic
Dim dbName As ULConnectionStringBuilder = _
  new ULConnectionStringBuilder
dbName.DatabaseOnCE = "¥UltraLite¥mydb1.udb"
dbName.DatabaseOnDesktop = "somedir¥mydb2.udb"

// C#
ULConnectionStringBuilder dbName = new ULConnectionStringBuilder();
dbName.DatabaseOnCE = "¥¥UltraLite¥¥mydb1.udb";
dbName.DatabaseOnDesktop = @"somedir¥mydb2.udb";
```

Ultra Light データベース・ファイルに推奨されるファイル拡張子は .udb です。Windows Mobile デバイスでは、デフォルトのデータベースは ¥UltraLiteDB¥ulstore.udb です。その他の Windows プラットフォームでは、デフォルトのデータベースは ulstore.udb です。C# では、パス内の円記

号をエスケープするか、アットマーク (@) で囲まれた文字列リテラルを使用する必要があります。

複数のデータベースを使用している場合は、各データベースのデータベース名を指定する必要があります。詳細については、「DatabaseName プロパティ」 210 ページを参照してください。

Optional connection settings

アプリケーションでの必要性やデータベースの作成方法によっては、デフォルト以外の ULConnectionStringBuilder.UserID や ULConnectionStringBuilder.Password、データベースの ULConnectionStringBuilder.DatabaseKey、接続の ULConnectionStringBuilder.CacheSize を指定することが必要になる場合があります。複数の接続を使用するアプリケーションでは、各接続に対してユニークな ULConnectionStringBuilder.ConnectionName を指定する必要があります。

データベースは、1 人の認証済みユーザ DBA で作成されます。このユーザの最初のパスワードは sql です。デフォルトでは、ユーザ ID DBA とパスワード sql を使用して、接続が開かれます。デフォルトのユーザを無効にするには、ULConnection.RevokeConnectFrom を使用します。ユーザを追加したりユーザのパスワードを変更するには、ULConnection.GrantConnectTo を使用します。

データベースを作成したときに暗号化キーを指定した場合は、そのデータベースへの以後の接続では、すべて同じ暗号化キーを使用する必要があります。データベースの暗号化キーを変更するには、ULConnection.ChangeEncryptionKey を使用します。

詳細については、「Ultra Light 接続パラメータ」 『Ultra Light データベース管理とリファレンス』を参照してください。

参照

- 「ULConnectionStringBuilder メンバ」 205 ページ
- 「Open メソッド」 177 ページ
- 「DropDatabase メソッド」 259 ページ
- DbConnectionStringBuilder.ConnectionString
- 「ULConnection クラス」 139 ページ
- 「ULConnection(String) コンストラクタ」 144 ページ
- 「ConnectionString プロパティ」 146 ページ
- 「DatabaseName プロパティ」 210 ページ
- 「UserID プロパティ」 216 ページ
- 「Password プロパティ」 214 ページ
- 「DatabaseKey プロパティ」 209 ページ
- 「CacheSize プロパティ」 208 ページ
- 「ConnectionName プロパティ」 209 ページ
- 「RevokeConnectFrom メソッド」 179 ページ
- 「GrantConnectTo メソッド」 176 ページ
- 「ChangeEncryptionKey メソッド」 157 ページ

ULConnectionStringBuilder メンバ

パブリック・コンストラクタ

メンバ名	説明
「ULConnectionStringBuilder コンストラクタ」 207 ページ	「ULConnectionStringBuilder クラス」 203 ページの新しいインスタンスを初期化します。

パブリック・プロパティ

メンバ名	説明
BrowsableConnectionString (DbConnectionStringBuilder から継承)	Visual Studio デザイナで DbConnectionStringBuilder.ConnectionString を表示するかどうかを示す値を取得または設定します。
「CacheSize プロパティ」 208 ページ	**UL 拡張：**キャッシュのサイズを指定します。
「ConnectionName プロパティ」 209 ページ	接続の名前を指定します。接続名が必要となるのは、データベースとの接続を複数作成する場合だけです。
ConnectionString (DbConnectionStringBuilder から継承)	DbConnectionStringBuilder に関連付けられた接続文字列を取得または設定します。
Count (DbConnectionStringBuilder から継承)	DbConnectionStringBuilder.ConnectionString に含まれているキーの現在の数を取得します。
「DatabaseKey プロパティ」 209 ページ	データベースを暗号化するためのキーを指定します。
「DatabaseName プロパティ」 210 ページ	データベース名、または接続する必要のあるロードしたデータベースの名前を指定します。
「DatabaseOnCE プロパティ」 211 ページ	**UL 拡張：**Windows Mobile 上の Ultra Light データベースのパスとファイル名を指定します。
「DatabaseOnDesktop プロパティ」 211 ページ	**UL 拡張：**Windows デスクトップ・プラットフォーム上の Ultra Light データベースのパスとファイル名を指定します。
IsFixedSize (DbConnectionStringBuilder から継承)	DbConnectionStringBuilder のサイズが固定かどうかを示す値を取得します。
IsReadOnly (DbConnectionStringBuilder から継承)	DbConnectionStringBuilder が読み込み専用かどうかを示す値を取得します。
「Item プロパティ」 212 ページ	指定された接続キーワードの値を指定します。

メンバ名	説明
Keys (DbConnectionStringBuilder から継承)	DbConnectionStringBuilder にキーが含まれている ICollection を取得します。
「OrderedTableScans プロパティ」 213 ページ	ORDER BY 句を指定しない SQL クエリで、デフォルトで順序付けされたテーブル・スキャンを実行するかどうかを指定します。
「Password プロパティ」 214 ページ	認証済みユーザのパスワードを指定します。
「ReserveSize プロパティ」 215 ページ	**UL 拡張：** Ultra Light の永続的データの保管に使用する予約ファイル・システム領域を指定します。
「StartLine プロパティ」 215 ページ	ロケーションを指定して、Ultra Light エンジンを開始します。
「UserID プロパティ」 216 ページ	データベースで認証されるユーザを指定します。
Values (DbConnectionStringBuilder から継承)	DbConnectionStringBuilder に値が含まれている ICollection を取得します。

パブリック・メソッド

メンバ名	説明
Add (DbConnectionStringBuilder から継承)	指定されたキーと値を持つエントリを DbConnectionStringBuilder に追加します。
Clear (DbConnectionStringBuilder から継承)	DbConnectionStringBuilder インスタンスの内容をクリアします。
「ContainsKey メソッド」 217 ページ	ULConnectionStringBuilder オブジェクトに特定のキーワードが含まれているかどうかを判断します。
「EquivalentTo メソッド」 217 ページ	この ULConnectionStringBuilder オブジェクト内の接続情報と指定された DbConnectionStringBuilder オブジェクト内の接続情報を比較します。
「GetShortName メソッド」 218 ページ	指定されたキーワードの短縮バージョンを取得します。
「Remove メソッド」 219 ページ	指定されたキーが設定されたエントリを ULConnectionStringBuilder インスタンスから削除します。

メンバ名	説明
ShouldSerialize (DbConnectionStringBuilder から継承)	指定されたキーが、この DbConnectionStringBuilder インスタンスに存在するかどうかを示します。
ToString (DbConnectionStringBuilder から継承)	この DbConnectionStringBuilder に関連付けられた接続文字列を返します。
「TryGetValue メソッド」 219 ページ	入力されたキーに対応する値を、この ULConnectionStringBuilder から取り出します。

参照

- 「ULConnectionStringBuilder クラス」 203 ページ
- 「Open メソッド」 177 ページ
- 「DropDatabase メソッド」 259 ページ
- DbConnectionStringBuilder.ConnectionString
- 「ULConnection クラス」 139 ページ
- 「ULConnection(String) コンストラクタ」 144 ページ
- 「ConnectionString プロパティ」 146 ページ
- 「DatabaseName プロパティ」 210 ページ
- 「UserID プロパティ」 216 ページ
- 「Password プロパティ」 214 ページ
- 「DatabaseKey プロパティ」 209 ページ
- 「CacheSize プロパティ」 208 ページ
- 「ConnectionName プロパティ」 209 ページ
- 「RevokeConnectFrom メソッド」 179 ページ
- 「GrantConnectTo メソッド」 176 ページ
- 「ChangeEncryptionKey メソッド」 157 ページ

ULConnectionStringBuilder コンストラクタ

「ULConnectionStringBuilder クラス」 203 ページの新しいインスタンスを初期化します。

ULConnectionStringBuilder() コンストラクタ

ULConnectionStringBuilder インスタンスを、そのデフォルト値で初期化します。

構文

Visual Basic
Public Sub **New()**

C#
public **ULConnectionStringBuilder();**

参照

- 「ULConnectionStringBuilder クラス」 203 ページ
- 「ULConnectionStringBuilder メンバ」 205 ページ
- 「ULConnectionStringBuilder コンストラクタ」 207 ページ

ULConnectionStringBuilder(String) コンストラクタ

ULConnectionStringBuilder インスタンスを、指定された接続文字列で初期化します。

構文

Visual Basic

```
Public Sub New( _
  ByVal connectionString As String _
)
```

C#

```
public  ULConnectionStringBuilder(
  string  connectionString
);
```

パラメータ

- **connectionString**　Ultra Light.NET の接続文字列。接続文字列は、キーワード値の組がセミコロンで区切られたリストです。

参照

- 「ULConnectionStringBuilder クラス」 203 ページ
- 「ULConnectionStringBuilder メンバ」 205 ページ
- 「ULConnectionStringBuilder コンストラクタ」 207 ページ

CacheSize プロパティ

UL 拡張：キャッシュのサイズを指定します。

構文

Visual Basic

```
Public Property CacheSize As String
```

C#

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

プロパティ値

キャッシュ・サイズを指定する文字列。デフォルトは NULL 参照 (Visual Basic の Nothing) です。この場合、デフォルトの 16 ページが使用されます。

備考

キャッシュ・サイズの値は、バイト単位で指定します。キロバイトの単位を示すにはサフィックス k または K を使用し、メガバイトの単位を示すにはサフィックス m または M を使用します。

たとえば、次の例ではキャッシュ・サイズが 128 KB に設定されます。

```
connParms.CacheSize = "128k"
```

キャッシュ・サイズが未指定であるか、正しく指定されていない場合は、デフォルトのサイズが使用されます。デフォルトのキャッシュ・サイズは 16 ページです。デフォルトのページ・サイズは 4 KB なので、デフォルトのキャッシュ・サイズは 64 KB です。キャッシュの最小サイズは、プラットフォームによって異なります。

デフォルトのキャッシュ・サイズは小さめの値です。テスト結果により、パフォーマンスの向上が必要と判断した場合は、キャッシュ・サイズを大きくしてください。キャッシュ・サイズをデータベース自体のサイズよりも大きくすると、パフォーマンスは向上しません。また、キャッシュ・サイズが大きいと、その他の使用可能なアプリケーションの数が少なくなることがあります。

参照

- 「Ultra Light CACHE_SIZE 接続パラメータ」『Ultra Light データベース管理とリファレンス』
- 「ULConnectionStringBuilder クラス」 203 ページ
- 「ULConnectionStringBuilder メンバ」 205 ページ

ConnectionName プロパティ

接続の名前を指定します。接続名が必要となるのは、データベースとの接続を複数作成する場合だけです。

構文

Visual Basic
Public Property **ConnectionName** As String

C#
public string **ConnectionName** { get; set; }

プロパティ値

接続の名前を指定する文字列。デフォルト値は NULL 参照 (Visual Basic の Nothing) です。

参照

- 「ULConnectionStringBuilder クラス」 203 ページ
- 「ULConnectionStringBuilder メンバ」 205 ページ

DatabaseKey プロパティ

データベースを暗号化するためのキーを指定します。

構文

Visual Basic
Public Property **DatabaseKey** As String

C#
public string **DatabaseKey** { get; set; }

プロパティ値

暗号化キーを指定する文字列。デフォルト値は NULL 参照 (Visual Basic の Nothing) で、暗号化は行われません。

備考

すべての接続では、データベースが作成されたときに指定されたキーと同じキーを使用する必要があります。キーを忘れた場合はデータベースにまったくアクセスできなくなります。

すべてのパスワードに共通することですが、最善の方法は、簡単には推測できないキー値を選択することです。キーの長さは任意ですが、短いと推測されやすいため、一般的には長い方が適しています。数字、文字、特殊文字を組み合わせると、キーは推測されにくくなります。

参照

- 「ULConnectionStringBuilder クラス」 203 ページ
- 「ULConnectionStringBuilder メンバ」 205 ページ
- 「ChangeEncryptionKey メソッド」 157 ページ

DatabaseName プロパティ

データベース名、または接続する必要のあるロードしたデータベースの名前を指定します。

構文

Visual Basic
Public Property **DatabaseName** As String

C#
public string **DatabaseName** { get; set; }

プロパティ値

データベースの名前を指定する文字列。デフォルト値は NULL 参照 (Visual Basic の Nothing) です。

備考

データベースが起動されると、データベース名が割り当てられます。データベース名は、dbn パラメータを使用して明示的に割り当てられるか、または拡張子とパスが削除されたベース・ファイル名を使用して Ultra Light によって割り当てられます。

接続が開かれると、Ultra Light は dbn が一致する実行中のデータベースをまず検索します。一致するデータベースが見つからない場合、Ultra Light は適切なデータベースのファイル名のパラ

メータ (DatabaseOnCE または DatabaseOnDesktop) を使用して、新しいデータベースを起動します。

このパラメータが必要なのは、同じベース・ファイル名を持つ 2 つの異なるデータベースにアプリケーション (または Ultra Light エンジン) がアクセスする場合です。

参照

- 「ULConnectionStringBuilder クラス」 203 ページ
- 「ULConnectionStringBuilder メンバ」 205 ページ
- 「DatabaseOnCE プロパティ」 211 ページ
- 「DatabaseOnDesktop プロパティ」 211 ページ

DatabaseOnCE プロパティ

UL 拡張：Windows Mobile 上の Ultra Light データベースのパスとファイル名を指定します。

構文

Visual Basic
Public Property **DatabaseOnCE** As String

C#
public string **DatabaseOnCE** { get; set; }

プロパティ値

データベースへのフル・パスを指定する文字列。値が NULL 参照 (Visual Basic の Nothing) の場合、データベース ¥UltraLiteDB¥ulstore.udb が使用されます。C# では、パス内の円記号をエスケープするか、アットマーク (@) で囲まれた文字列リテラルを使用する必要があります。デフォルト値は NULL 参照 (Visual Basic の Nothing) です。

参照

- 「ULConnectionStringBuilder クラス」 203 ページ
- 「ULConnectionStringBuilder メンバ」 205 ページ

DatabaseOnDesktop プロパティ

UL 拡張：Windows デスクトップ・プラットフォーム上の Ultra Light データベースのパスとファイル名を指定します。

構文

Visual Basic
Public Property **DatabaseOnDesktop** As String

C#
public string **DatabaseOnDesktop** { get; set; }

プロパティ値

データベースへの絶対パスまたは相対パスを指定する文字列。値が NULL 参照 (Visual Basic の Nothing) の場合、データベース ulstore.udb が使用されます。C# では、パス内の円記号をエスケープするか、アットマーク (@) で囲まれた文字列リテラルを使用する必要があります。デフォルト値は NULL 参照 (Visual Basic の Nothing) です。

参照

- 「ULConnectionStringBuilder クラス」 203 ページ
- 「ULConnectionStringBuilder メンバ」 205 ページ

Item プロパティ

指定された接続キーワードの値を指定します。

構文

Visual Basic

```
Public Overrides Default Property Item( _
  ByVal keyword As String _
) As Object
```

C#

```
public override object this[
  string  keyword
] { get; set; }
```

パラメータ

- **keyword**　接続キーワードの名前。

プロパティ値

指定された接続キーワードの値を表すオブジェクト。

備考

接続キーワードと、ULConnectionStringBuilder の対応するプロパティを次の表に示します。

キーワード	対応するプロパティ
cache_size	「CacheSize プロパティ」 208 ページ
ce_file	「DatabaseOnCE プロパティ」 211 ページ
con	「ConnectionName プロパティ」 209 ページ
dbkey	「DatabaseKey プロパティ」 209 ページ
dbn	「DatabaseName プロパティ」 210 ページ

キーワード	対応するプロパティ
nt_file	「DatabaseOnDesktop プロパティ」 211 ページ
pwd	「Password プロパティ」 214 ページ
reserve_size	「ReserveSize プロパティ」 215 ページ
start	「StartLine プロパティ」 215 ページ
uid	「UserID プロパティ」 216 ページ

参照

- 「ULConnectionStringBuilder クラス」 203 ページ
- 「ULConnectionStringBuilder メンバ」 205 ページ
- 「CacheSize プロパティ」 208 ページ
- 「DatabaseOnCE プロパティ」 211 ページ
- 「ConnectionName プロパティ」 209 ページ
- 「DatabaseKey プロパティ」 209 ページ
- 「DatabaseName プロパティ」 210 ページ
- 「DatabaseOnDesktop プロパティ」 211 ページ
- 「Password プロパティ」 214 ページ
- 「ReserveSize プロパティ」 215 ページ
- 「StartLine プロパティ」 215 ページ
- 「UserID プロパティ」 216 ページ

OrderedTableScans プロパティ

ORDER BY 句を指定しない SQL クエリで、デフォルトで順序付けされたテーブル・スキャンを実行するかどうかを指定します。

構文

Visual Basic
Public Property **OrderedTableScans** As String

C#
public string **OrderedTableScans** { get; set; }

プロパティ値

順序付けされたテーブル・スキャンを実行するかどうかを指定する boolean 文字列。たとえば、true と false、yes と no、1 と 0 などです。デフォルト値は NULL 参照 (Visual Basic の Nothing) です。

備考

リリース 10.0.1 では、Ultra Light で動的 SQL を使用するとき、クエリ実行の順序は重要ではありません。Ultra Light は、プライマリ・キー・インデックスを使用するのではなく、データベース・ページからローに直接アクセスします。これにより、ローをフェッチするパフォーマンスが改善されました。この最適化を使用するには、クエリが読み込み専用であり、すべてのローをスキャンする必要があります。

ローが特定の順序に並べることを期待する場合は、SQL クエリに ORDER BY 文を指定します。ただし、一部のアプリケーションでは、プライマリ・キーの順にローを返すようなデフォルトの動作に依存する可能性があります。このような場合は、OrderedTableScans パラメータを 1 (true、yes、on) に設定し、テーブル上で反復するときに以前の動作に戻すようにすることが必要です。

OrderedTableScans を 1 (true、yes、on) に設定しても、ORDER BY 句を指定しなかったり、クエリがインデックスを利用できなかったりすると、Ultra Light はプライマリ・キーを使用してデフォルトの動作を実行します。

参照

- 「ULConnectionStringBuilder クラス」 203 ページ
- 「ULConnectionStringBuilder メンバ」 205 ページ

Password プロパティ

認証済みユーザのパスワードを指定します。

構文

Visual Basic
Public Property **Password** As String

C#
public string **Password** { get; set; }

プロパティ値

データベースのユーザ ID を指定する文字列。デフォルトは NULL 参照 (Visual Basic の Nothing) です。

備考

パスワードでは大文字と小文字が区別されます。

データベースが作成されると、ユーザ ID DBA のパスワードは SQL に設定されます。

参照

- 「ULConnectionStringBuilder クラス」 203 ページ
- 「ULConnectionStringBuilder メンバ」 205 ページ
- 「UserID プロパティ」 216 ページ

ReserveSize プロパティ

UL 拡張：Ultra Light の永続的データの保管に使用する予約ファイル・システム領域を指定します。

構文

Visual Basic
Public Property **ReserveSize** As String

C#
public string **ReserveSize** { get; set; }

プロパティ値

予約サイズを指定する文字列。デフォルト値は NULL 参照 (Visual Basic の Nothing) です。

備考

予約サイズ・パラメータの値は、バイト単位で指定します。キロバイトの単位を示すにはサフィックス k または K を使用し、メガバイトの単位を示すにはサフィックス m または M を使用します。

reserve_size パラメータを使用すると、データを挿入することなく、Ultra Light データベースが必要とするファイル・システム領域を事前に割り付けることができます。ファイル・システムの領域を予約すると、パフォーマンスが多少向上し、メモリが不足するという障害を防ぐことができます。デフォルトでは、永続ストア・ファイルのサイズは、アプリケーションがデータベースを更新して、サイズを大きくする必要が生じた場合にだけ大きくなります。

reserve_size で予約されるファイル・システム領域には、未加工データだけでなく、永続ストア・ファイルのメタデータも含まれることに注意してください。データベースのデータ量から、必要なファイル・システム領域を計算する場合は、メタデータのオーバヘッドとデータの圧縮を考慮してください。テスト・データを入れてデータベースを実行し、永続ストア・ファイルのサイズを確認することをおすすめします。

reserve_size パラメータは、起動時に永続ストア・ファイルを設定された予約サイズまで大きくすることにより、領域を予約します。これは、そのファイルが以前に存在していたかどうかに関係なく行われます。このファイルはトランケートされません。

次のパラメータ文字列では、起動時に永続ストア・ファイルのサイズが最低 2 MB 確保されます。

```
connParms.ReserveSize = "2m"
```

参照

- 「ULConnectionStringBuilder クラス」 203 ページ
- 「ULConnectionStringBuilder メンバ」 205 ページ

StartLine プロパティ

ロケーションを指定して、Ultra Light エンジンを開始します。

構文

Visual Basic
Public Property **StartLine** As String

C#
public string **StartLine** { get; set; }

プロパティ値

Ultra Light エンジンの実行可能ファイルのロケーションを指定する文字列。デフォルト値は NULL 参照 (Visual Basic の Nothing) です。

備考

現在実行中でないエンジンに接続する場合は、StartLine (START) 接続パラメータだけを指定します。

Ultra Light.NET で Ultra Light エンジンを使用する方法の詳細については、「RuntimeType プロパティ」 256 ページを参照してください。

参照

- 「ULConnectionStringBuilder クラス」 203 ページ
- 「ULConnectionStringBuilder メンバ」 205 ページ
- 「RuntimeType プロパティ」 256 ページ

UserID プロパティ

データベースで認証されるユーザを指定します。

構文

Visual Basic
Public Property **UserID** As String

C#
public string **UserID** { get; set; }

プロパティ値

データベースのユーザ ID を指定する文字列。デフォルト値は NULL 参照 (Visual Basic の Nothing) です。

備考

ユーザ ID の大文字と小文字は区別されません。

データベースは最初、DBA という名前の 1 人の認証済みユーザで作成されます。

ユーザ ID とパスワードを両方とも入力しないと、ユーザ名 DBA とパスワード SQL が使用されます。データベースをより強力に保護するには、ユーザ名 DBA のパスワードを変更するか、新しいユーザを作成して (ULConnection.GrantConnectTo を使用)、DBA ユーザを削除します (ULConnection.RevokeConnectFrom を使用)。

参照

- 「ULConnectionStringBuilder クラス」 203 ページ
- 「ULConnectionStringBuilder メンバ」 205 ページ
- 「Password プロパティ」 214 ページ
- 「GrantConnectTo メソッド」 176 ページ
- 「RevokeConnectFrom メソッド」 179 ページ

ContainsKey メソッド

ULConnectionStringBuilder オブジェクトに特定のキーワードが含まれているかどうかを判断します。

構文

Visual Basic

```
Public Overrides Function ContainsKey( _
  ByVal keyword As String _
) As Boolean
```

C#

```
public override bool ContainsKey(
  string  keyword
);
```

パラメータ

- **keyword**　接続キーワードの名前。

戻り値

指定されたキーワードの値がこの接続文字列ビルダに含まれる場合は True、それ以外の場合は false を返します。

参照

- 「ULConnectionStringBuilder クラス」 203 ページ
- 「ULConnectionStringBuilder メンバ」 205 ページ

EquivalentTo メソッド

この ULConnectionStringBuilder オブジェクト内の接続情報と指定された DbConnectionStringBuilder オブジェクト内の接続情報を比較します。

構文

Visual Basic

```
Public Overrides Function EquivalentTo( _
  ByVal connectionStringBuilder As DbConnectionStringBuilder _
) As Boolean
```

C#
```
public override bool EquivalentTo(
  DbConnectionstring Builder connectionStringBuilder
);
```

パラメータ

- **connectionStringBuilder**　この ULConnectionStringBuilder オブジェクトと比較する別の DbConnectionStringBuilder オブジェクト。

戻り値

このオブジェクトが指定された DbConnectionStringBuilder オブジェクトと同じ場合は True、それ以外の場合は false を返します。

参照

- 「ULConnectionStringBuilder クラス」 203 ページ
- 「ULConnectionStringBuilder メンバ」 205 ページ
- DbConnectionStringBuilder

GetShortName メソッド

指定されたキーワードの短縮バージョンを取得します。

構文

Visual Basic
```
Public Shared Function GetShortName( _
  ByVal keyword As String _
) As String
```

C#
```
public static string  GetShortName(
  string  keyword
);
```

パラメータ

- **keyword**　取り出す項目のキー。

戻り値

キーワードが認識された場合は、指定されたキーワードの短縮バージョン。その他の場合は NULL。

参照

- 「ULConnectionStringBuilder クラス」 203 ページ
- 「ULConnectionStringBuilder メンバ」 205 ページ

Remove メソッド

指定されたキーが設定されたエントリを ULConnectionStringBuilder インスタンスから削除します。

構文

Visual Basic

```
Public Overrides Function Remove( _
  ByVal keyword As String _
) As Boolean
```

C#

```
public override bool Remove(
  string  keyword
);
```

パラメータ

- **keyword**　接続キーワードの名前。

戻り値

接続文字列内のキーが削除された場合は true、キーが存在しなかった場合は false。

参照

- 「ULConnectionStringBuilder クラス」 203 ページ
- 「ULConnectionStringBuilder メンバ」 205 ページ

TryGetValue メソッド

入力されたキーに対応する値を、この ULConnectionStringBuilder から取り出します。

構文

Visual Basic

```
Public Overrides Function TryGetValue( _
  ByVal keyword As String, _
  ByVal value As Object _
) As Boolean
```

C#

```
public override bool TryGetValue(
  string  keyword,
  object value
);
```

パラメータ

- **keyword**　取り出す項目のキー。
- **value**　キーに対応する値。

戻り値

キーワードが接続文字列内にある場合は true、それ以外は false。

備考

TryGetValue メソッドを使用すると、開発者は、安全に ULConnectionStringBuilder から値を取得できます。最初に ContainsKey メソッドを呼び出す必要はありません。TryGetValue は、存在しないキーを指定してこのメソッドを呼び出しても例外を発行しないため、値を取得する前にキーを検索する必要はありません。存在しないキーを指定して TryGetValue を呼び出すと、値パラメータに NULL 値 (Visual Basic の Nothing) が設定されます。

参照

- 「ULConnectionStringBuilder クラス」 203 ページ
- 「ULConnectionStringBuilder メンバ」 205 ページ

ULCreateParms クラス

UL 拡張：Ultra Light データベースを作成するときの作成時オプションの文字列を作成します。

構文

Visual Basic
Public Class **ULCreateParms**

C#
public class **ULCreateParms**

備考

ULCreateParms オブジェクトは、データベースを作成するときのパラメータを使用するために使用します (ULDatabaseManager.CreateDatabase(string,byte[],string))。

前後のスペースは、すべての値で無視されます。値には、前後のスペースやセミコロン (;) を含めることはできません。また、一重引用符 (') または二重引用符 (") で始めることはできません。

ULCreateParms オブジェクトで適切なプロパティを設定して、作成パラメータをすべて指定したら、ULCreateParms.ToString を使用して作成パラメータ文字列を作成します。結果として返される文字列は ULDatabaseManager.CreateDatabase(string,byte[],string) メソッドの createParms パラメータとして使用できます。

詳細については、「Ultra Light 接続パラメータ」 『Ultra Light データベース管理とリファレンス』を参照してください。

例

次のコードでは、Windows Mobile デバイス上にデータベース ¥UltraLite¥MyDatabase.udb を作成します。このデータベースは、大文字と小文字を区別し、UTF8 文字セットを使用して作成されます。

```
' Visual Basic
Dim createParms As ULCreateParms = New ULCreateParms
createParms.CaseSensitive = True
createParms.UTF8Encoding = True
Dim openParms As ULConnectionParms = New ULConnectionParms
openParms.DatabaseOnCE = "¥UltraLite¥MyDatabase.udb"
' Assumes file coll_1250LATIN2.vb is
' also compiled in the current project
ULConnection.DatabaseManager.CreateDatabase( _
    openParms.ToString(), _
    iAnywhere.UltraLite.Collations.Collation_1250LATIN2.Data, _
    createParms.ToString() _
  )
Dim conn As ULConnection = _
  New ULConnection( openParms.ToString() )
conn.Open()


// C#
ULCreateParms createParms = new ULCreateParms();
createParms.CaseSensitive = true;
createParms.UTF8Encoding = true;
ULConnectionParms openParms = new ULConnectionParms();
```

```
openParms.DatabaseOnCE = @"¥UltraLite¥MyDatabase.udb";
// Assumes file coll_1250LATIN2.vb is
// also compiled in the current project
ULConnection.DatabaseManager.CreateDatabase(
    openParms.ToString(),
    iAnywhere.UltraLite.Collations.Collation_1250LATIN2.Data,
    createParms.ToString()
  );
ULConnection conn = new ULConnection( openParms.ToString() );
conn.Open();
```

参照

- 「ULCreateParms メンバ」 222 ページ
- 「GetDatabaseProperty メソッド」 269 ページ
- 「CreateDatabase メソッド」 257 ページ
- 「ToString メソッド」 231 ページ

ULCreateParms メンバ

パブリック・コンストラクタ

メンバ名	説明
「ULCreateParms コンストラクタ」 224 ページ	ULCreateParms インスタンスを、そのデフォルト値で初期化します。

パブリック・プロパティ

メンバ名	説明
「CaseSensitive プロパティ」 224 ページ	文字列を比較するときに、新しいデータベースで大文字と小文字を区別するかどうかを指定します。
「ChecksumLevel プロパティ」 224 ページ	新しいデータベースで有効にするデータベース・ページのチェックサムのレベルを指定します。
「DateFormat プロパティ」 225 ページ	新しいデータベースで文字列変換に使用される日付フォーマットを指定します。
「DateOrder プロパティ」 225 ページ	新しいデータベースで文字列変換に使用される日付順を指定します。
「FIPS プロパティ」 226 ページ	新しいデータベースで使用する暗号化 (AES_FIPS または AES) を指定します。
「MaxHashSize プロパティ」 226 ページ	新しいデータベースでインデックスのハッシュに使用するデフォルトの最大バイト数を指定します。

メンバ名	説明
「NearestCentury プロパティ」 227 ページ	新しいデータベースで文字列変換に使用される最も近い世紀を指定します。
「Obfuscate プロパティ」 227 ページ	新しいデータベースで難読化 (単純暗号化) を使用するかどうかを指定します。
「PageSize プロパティ」 228 ページ	データベースのページ・サイズ (バイトまたはキロバイト単位) を指定します。
「Precision プロパティ」 228 ページ	データベースで文字列変換に使用される浮動小数点の精度を指定します。
「Scale プロパティ」 229 ページ	新しいデータベースによる文字列変換中に、計算結果が最大 PRECISION にトランケートされるときの小数点以下の最大桁数を指定します。
「TimeFormat プロパティ」 229 ページ	データベースで文字列変換に使用される時刻フォーマットを指定します。
「TimestampFormat プロパティ」 230 ページ	データベースで文字列変換に使用されるタイムスタンプのフォーマットを指定します。
「TimestampIncrement プロパティ」 230 ページ	2 つのユニークなタイムスタンプ値間のマイクロ秒 (1,000,000 分の 1 秒) 単位の最小差を指定します。
「UTF8Encoding プロパティ」 231 ページ	新しいデータベースで使用する文字セット (UTF8 文字セットまたは照合に関連付けられた文字セット) を指定します。

パブリック・メソッド

メンバ名	説明
「ToString メソッド」 231 ページ	このインスタンスの文字列表現を返します。

参照

- 「ULCreateParms クラス」 221 ページ
- 「GetDatabaseProperty メソッド」 269 ページ
- 「CreateDatabase メソッド」 257 ページ
- 「ToString メソッド」 231 ページ

ULCreateParms コンストラクタ

ULCreateParms インスタンスを、そのデフォルト値で初期化します。

構文

Visual Basic
Public Sub **New()**

C#
public **ULCreateParms();**

参照

- 「ULCreateParms クラス」 221 ページ
- 「ULCreateParms メンバ」 222 ページ

CaseSensitive プロパティ

文字列を比較するときに、新しいデータベースで大文字と小文字を区別するかどうかを指定します。

構文

Visual Basic
Public Property **CaseSensitive** As Boolean

C#
public bool **CaseSensitive** { get; set; }

プロパティ値

データベースで大文字と小文字が区別される場合は true、区別されない場合は false。デフォルトは false です。

備考

CaseSensitive は、文字列データの比較とソートの方法にのみ影響します。テーブル名、カラム名、インデックス名、接続ユーザ ID などのデータベース識別子については、どのような場合であっても、大文字と小文字は区別されません。接続用のパスワードとデータベース暗号化キーでは、常に、大文字と小文字が区別されます。

参照

- 「ULCreateParms クラス」 221 ページ
- 「ULCreateParms メンバ」 222 ページ

ChecksumLevel プロパティ

新しいデータベースで有効にするデータベース・ページのチェックサムのレベルを指定します。

構文

Visual Basic
Public Property **ChecksumLevel** As Integer

C#
public int **ChecksumLevel** { get; set; }

プロパティ値

チェックサムのレベルを指定する整数。有効な値は 0、1、2 です。デフォルトは 0 です。

参照

- 「ULCreateParms クラス」 221 ページ
- 「ULCreateParms メンバ」 222 ページ

DateFormat プロパティ

新しいデータベースで文字列変換に使用される日付フォーマットを指定します。

構文

Visual Basic
Public Property **DateFormat** As String

C#
public string **DateFormat** { get; set; }

プロパティ値

日付フォーマットを指定する文字列。値が NULL 参照 (Visual Basic の Nothing) の場合、データベースでは "YYYY-MM-DD" が使用されます。C# では、パス内の円記号をエスケープするか、アットマーク (@) で囲まれた文字列リテラルを使用する必要があります。デフォルト値は NULL 参照 (Visual Basic の Nothing) です。

参照

- 「ULCreateParms クラス」 221 ページ
- 「ULCreateParms メンバ」 222 ページ

DateOrder プロパティ

新しいデータベースで文字列変換に使用される日付順を指定します。

構文

Visual Basic
Public Property **DateOrder** As ULDateOrder

C#
public ULDateOrder **DateOrder** { get; set; }

プロパティ値

文字列変換に使用される日付順を指定する ULDateOrder 値。デフォルト値は YMD です。

参照

- 「ULCreateParms クラス」 221 ページ
- 「ULCreateParms メンバ」 222 ページ
- 「ULDateOrder 列挙体」 316 ページ

FIPS プロパティ

新しいデータベースで使用する暗号化 (AES_FIPS または AES) を指定します。

構文

Visual Basic
Public Property **FIPS** As Boolean

C#
public bool **FIPS** { get; set; }

プロパティ値

データベースで AES_FIPS を使用して暗号化される場合は true、AES を使用して暗号化される場合は false。デフォルトは false です。

備考

新しいデータベースが作成されたときは、接続パラメータ EncryptionKey の値を指定して暗号化を有効にする必要があります。FIPS を true に設定している場合で、暗号化キーを指定しないと、ULDatabaseManager.CreateDatabase(string,byte[],string) メソッドは、暗号化キーが見つからないために失敗します。

参照

- 「ULCreateParms クラス」 221 ページ
- 「ULCreateParms メンバ」 222 ページ
- 「EncryptionKey プロパティ」 198 ページ
- 「CreateDatabase メソッド」 257 ページ

MaxHashSize プロパティ

新しいデータベースでインデックスのハッシュに使用するデフォルトの最大バイト数を指定します。

構文

Visual Basic
Public Property **MaxHashSize** As Integer

C#
public int **MaxHashSize** { get; set; }

プロパティ値

最大ハッシュ・サイズを指定する整数。値は、[0.32] の範囲内であることが必要です。デフォルトは 8 です。

参照

- 「ULCreateParms クラス」 221 ページ
- 「ULCreateParms メンバ」 222 ページ

NearestCentury プロパティ

新しいデータベースで文字列変換に使用される最も近い世紀を指定します。

構文

Visual Basic
Public Property **NearestCentury** As Integer

C#
public int **NearestCentury** { get; set; }

プロパティ値

最も近い世紀を指定する整数。値は、[0.100] の範囲内であることが必要です。デフォルトは 50 です。

参照

- 「ULCreateParms クラス」 221 ページ
- 「ULCreateParms メンバ」 222 ページ

Obfuscate プロパティ

新しいデータベースで難読化 (単純暗号化) を使用するかどうかを指定します。

構文

Visual Basic
Public Property **Obfuscate** As Boolean

C#
public bool **Obfuscate** { get; set; }

プロパティ値

データベースで難読化を使用して暗号化される場合は true、難読化されない場合は false。デフォルトは false です。

備考

FIPS 暗号化が有効になっている場合 (ULCreateParms.FIPS) は、このオプションは無視されます。難読化が有効になっている場合に、新しいデータベースの作成時に接続パラメータ EncryptionKey (DBKEY) に値を指定すると、暗号化キーは無視されます。

参照

- 「ULCreateParms クラス」 221 ページ
- 「ULCreateParms メンバ」 222 ページ
- 「FIPS プロパティ」 226 ページ

PageSize プロパティ

データベースのページ・サイズ (バイトまたはキロバイト単位) を指定します。

構文

Visual Basic
Public Property **PageSize** As Integer

C#
public int **PageSize** { get; set; }

プロパティ値

ページ・サイズ (バイト単位) を指定する整数。有効な値は、1024 (1K)、2048 (2K)、4096 (4K)、8192 (8K)、16384 (16K) です。デフォルトは 4096 です。

参照

- 「ULCreateParms クラス」 221 ページ
- 「ULCreateParms メンバ」 222 ページ

Precision プロパティ

データベースで文字列変換に使用される浮動小数点の精度を指定します。

構文

Visual Basic
Public Property **Precision** As Integer

C#
public int **Precision** { get; set; }

プロパティ値

精度を指定する整数。値は、[1,127] の範囲内であることが必要です。デフォルトは 30 です。

参照

- 「ULCreateParms クラス」 221 ページ
- 「ULCreateParms メンバ」 222 ページ
- 「Scale プロパティ」 229 ページ

Scale プロパティ

新しいデータベースによる文字列変換中に、計算結果が最大 PRECISION にトランケートされるときの小数点以下の最大桁数を指定します。

構文

Visual Basic
Public Property **Scale** As Integer

C#
public int **Scale** { get; set; }

プロパティ値

位取りを指定する整数。値は、[0.127] の範囲内であることが必要です。デフォルトは 6 です。

備考

Scale は、Precision よりも小さいか等しいことが必要です。Scale が Precision よりも大きいと、データベースの作成時にエラーが発生します。

参照

- 「ULCreateParms クラス」 221 ページ
- 「ULCreateParms メンバ」 222 ページ

TimeFormat プロパティ

データベースで文字列変換に使用される時刻フォーマットを指定します。

構文

Visual Basic
Public Property **TimeFormat** As String

C#
public string **TimeFormat** { get; set; }

プロパティ値

時刻フォーマットを指定する文字列。値が NULL 参照 (Visual Basic の Nothing) の場合、データベースでは "HH:NN:SS.SSS" が使用されます。C# では、パス内の円記号をエスケープするか、アットマーク (@) で囲まれた文字列リテラルを使用する必要があります。デフォルト値は NULL 参照 (Visual Basic の Nothing) です。

参照

- 「ULCreateParms クラス」 221 ページ
- 「ULCreateParms メンバ」 222 ページ

TimestampFormat プロパティ

データベースで文字列変換に使用されるタイムスタンプのフォーマットを指定します。

構文

Visual Basic
Public Property **TimestampFormat** As String

C#
public string **TimestampFormat** { get; set; }

プロパティ値

タイムスタンプのフォーマットを指定する文字列。値が NULL 参照 (Visual Basic の Nothing) の場合、データベースでは "YYYY-MM-DD HH:NN:SS.SSS" が使用されます。C# では、パス内の円記号をエスケープするか、アットマーク (@) で囲まれた文字列リテラルを使用する必要があります。デフォルト値は NULL 参照 (Visual Basic の Nothing) です。

参照

- 「ULCreateParms クラス」 221 ページ
- 「ULCreateParms メンバ」 222 ページ

TimestampIncrement プロパティ

2 つのユニークなタイムスタンプ値間のマイクロ秒 (1,000,000 分の 1 秒) 単位の最小差を指定します。

構文

Visual Basic
Public Property **TimestampIncrement** As Integer

C#
public int **TimestampIncrement** { get; set; }

プロパティ値

タイムスタンプのインクリメントを指定する整数。値は、[1,60000000] の範囲内であることが必要です。デフォルトは 1 です。

参照

- 「ULCreateParms クラス」 221 ページ
- 「ULCreateParms メンバ」 222 ページ

UTF8Encoding プロパティ

新しいデータベースで使用する文字セット (UTF8 文字セットまたは照合に関連付けられた文字セット) を指定します。

構文

Visual Basic
Public Property **UTF8Encoding** As Boolean

C#
public bool **UTF8Encoding** { get; set; }

プロパティ値

データベースで UTF8 文字セットが使用される場合は true、照合に関連付けられた文字セットが使用される場合は false。デフォルトは false です。

備考

照合に関連付けられた文字セット以外の文字セットに文字を格納する場合は、UTF8 文字セットを選択します。たとえば、US ソートを使用したいけれども、現地の綴りで国際住所を格納したいために UTF8Encoding を true に設定している場合は、1252LATIN1 照合を使用してデータベースを作成します。

Symbian OS デバイスで使用するデータベースの場合、UTF8Encoding を true に設定する必要があります。

Palm OS デバイスで使用するデータベースの場合、UTF8Encoding を false のままにする必要があります。

参照

- 「ULCreateParms クラス」 221 ページ
- 「ULCreateParms メンバ」 222 ページ

ToString メソッド

このインスタンスの文字列表現を返します。

構文

Visual Basic
Public Overrides Function **ToString()** As String

C#
public override string **ToString();**

戻り値

「キーワード=値」の組み合わせがセミコロンで区切られたリスト形式の、このインスタンスの文字列表現。

参照

- 「ULCreateParms クラス」 221 ページ
- 「ULCreateParms メンバ」 222 ページ

ULCursorSchema クラス

UL 拡張：Ultra Light.NET カーソルのスキーマを表します。これは抽象クラスであるため、インスタンス化はできません。

構文

Visual Basic
MustInherit Public Class **ULCursorSchema**

C#
public abstract class **ULCursorSchema**

備考

このクラスは、ULTableSchema クラスと ULResultSetSchema クラスの抽象基本クラスです。

iAnywhere.UltraLite ネームスペースから移行するユーザへの注意：カラム ID は、iAnywhere.UltraLite ネームスペース内にあるため、1 ではなく 0 から始まります。

参照

- 「ULCursorSchema メンバ」 233 ページ
- 「ULTableSchema クラス」 571 ページ
- 「ULResultSetSchema クラス」 442 ページ

ULCursorSchema メンバ

パブリック・プロパティ

メンバ名	説明
「ColumnCount プロパティ」 234 ページ	このカーソル内のカラム数を返します。
「IsOpen プロパティ」 235 ページ	カーソルのスキーマが現在開いているかどうかを確認します。
「Name プロパティ」 235 ページ	カーソルの名前を返します。

パブリック・メソッド

メンバ名	説明
「GetColumnID メソッド」 235 ページ	指定されたカラムのカラム ID を返します。
「GetColumnName メソッド」 236 ページ	指定されたカラム ID で識別されたカラムの名前を返します。

メンバ名	説明
「GetColumnPrecision メソッド」 237 ページ	カラムが数値カラム (SQL NUMERIC 型) の場合は、指定されたカラム ID によって識別されたカラムの精度を返します。
「GetColumnSQLName メソッド」 238 ページ	指定されたカラム ID で識別されたカラムの名前を返します。
「GetColumnScale メソッド」 239 ページ	カラムが数値カラム (SQL NUMERIC 型) の場合は、指定されたカラム ID によって識別されたカラムの位取りを返します。
「GetColumnSize メソッド」 239 ページ	カラムがサイズ指定されたカラム (SQL BINARY 型または CHAR 型) の場合は、指定されたカラム ID によって識別されたカラムのサイズを返します。
「GetColumnULDbType メソッド」 240 ページ	指定されたカラム ID によって識別されたカラムの Ultra Light.NET データ型を返します。
「GetSchemaTable メソッド」 241 ページ	ULDataReader のカラムのスキーマが記述された System.Data.DataTable を返します。

参照

- 「ULCursorSchema クラス」 233 ページ
- 「ULTableSchema クラス」 571 ページ
- 「ULResultSetSchema クラス」 442 ページ

ColumnCount プロパティ

このカーソル内のカラム数を返します。

構文

Visual Basic
Public Readonly Property **ColumnCount** As Short

C#
public short **ColumnCount** { get;}

プロパティ値

カーソル内のカラム数。カーソルのスキーマが閉じている場合は 0。

備考

カラム ID の範囲は、0 ～ ColumnCount-1 です。

カラムの ID とカウントは、スキーマのアップグレード中に変更されることがあります。カラムを正しく識別するには、名前でアクセスするか、キャッシュされている ID とカウントをスキーマのアップグレード後にリフレッシュします。

参照

- 「ULCursorSchema クラス」 233 ページ
- 「ULCursorSchema メンバ」 233 ページ

IsOpen プロパティ

カーソルのスキーマが現在開いているかどうかを確認します。

構文

Visual Basic
Public Readonly Property **IsOpen** As Boolean

C#
public bool **IsOpen** { get;}

プロパティ値

カーソルのスキーマが現在開いている場合は true、閉じている場合は false。

参照

- 「ULCursorSchema クラス」 233 ページ
- 「ULCursorSchema メンバ」 233 ページ

Name プロパティ

カーソルの名前を返します。

構文

Visual Basic
Public Readonly Property **Name** As String

C#
public string **Name** { get;}

プロパティ値

文字列として返されるカーソル名。

参照

- 「ULCursorSchema クラス」 233 ページ
- 「ULCursorSchema メンバ」 233 ページ

GetColumnID メソッド

指定されたカラムのカラム ID を返します。

構文

Visual Basic
```
Public Function GetColumnID( _
  ByVal name As String _
) As Short
```

C#
```
public short GetColumnID(
  string  name
);
```

パラメータ

- **name**　カラムの名前。

戻り値

指定されたカラムのカラム ID。

備考

カラム ID の範囲は、0 ～ ColumnCount-1 です。

結果セットでは、すべてのカラムに名前があるとは限らないほか、すべてのカラム名がユニークであるとは限りません。エイリアスを使用していない場合は、非計算カラムの名前には、そのカラムの元となるテーブルの名前がプレフィクスとして付けられます。たとえば、MyTable.ID は、クエリ "SELECT ID FROM MyTable" の結果セットに含まれる唯一のカラムの名前です。

カラムの ID とカウントは、スキーマのアップグレード中に変更されることがあります。カラムを正しく識別するには、名前でアクセスするか、キャッシュされている ID とカウントをスキーマのアップグレード後にリフレッシュします。

参照

- 「ULCursorSchema クラス」 233 ページ
- 「ULCursorSchema メンバ」 233 ページ
- 「ColumnCount プロパティ」 234 ページ
- 「ColumnCount プロパティ」 234 ページ

GetColumnName メソッド

指定されたカラム ID で識別されたカラムの名前を返します。

構文

Visual Basic
```
Public Function GetColumnName( _
  ByVal columnID As Integer _
) As String
```

C#
```
public string  GetColumnName(
```

```
  int columnID
);
```

パラメータ

- **columnID**　カラムの ID。値は、[0,ColumnCount-1] の範囲内であることが必要です。

戻り値

カラムの名前。カラムに名前がない場合は NULL 参照 (Visual Basic の Nothing)。SQL クエリでカラムのエイリアスが使用されている場合は、そのエイリアスが返されます。

備考

結果セットでは、すべてのカラムに名前があるとは限らないほか、すべてのカラム名がユニークであるとは限りません。エイリアスを使用していない場合は、非計算カラムの名前には、そのカラムの元となるテーブルの名前がプレフィクスとして付けられます。たとえば、MyTable.ID は、クエリ "SELECT ID FROM MyTable" の結果セットに含まれる唯一のカラムの名前です。

カラムの ID とカウントは、スキーマのアップグレード中に変更されることがあります。カラムを正しく識別するには、名前でアクセスするか、キャッシュされている ID とカウントをスキーマのアップグレード後にリフレッシュします。

参照

- 「ULCursorSchema クラス」 233 ページ
- 「ULCursorSchema メンバ」 233 ページ
- 「ColumnCount プロパティ」 234 ページ
- 「ColumnCount プロパティ」 234 ページ

GetColumnPrecision メソッド

カラムが数値カラム (SQL NUMERIC 型) の場合は、指定されたカラム ID によって識別されたカラムの精度を返します。

構文

Visual Basic

```
Public Function GetColumnPrecision( _
  ByVal columnID As Integer _
) As Integer
```

C#

```
public int GetColumnPrecision(
  int columnID
);
```

パラメータ

- **columnID**　カラムの ID。値は、[0,ColumnCount-1] の範囲内であることが必要です。

戻り値

指定された数値カラムの精度。

参照

- 「ULCursorSchema クラス」 233 ページ
- 「ULCursorSchema メンバ」 233 ページ
- 「GetColumnULDbType メソッド」 240 ページ
- 「ColumnCount プロパティ」 234 ページ

GetColumnSQLName メソッド

指定されたカラム ID で識別されたカラムの名前を返します。

構文

Visual Basic

```
Public Function GetColumnSQLName( _
  ByVal columnID As Integer _
) As String
```

C#

```
public string  GetColumnSQLName(
  int columnID
);
```

パラメータ

- **columnID**　カラムの ID。値は、[0,ColumnCount-1] の範囲内であることが必要です。

戻り値

カラムの名前。カラムに名前がない場合は NULL 参照 (Visual Basic の Nothing)。SQL クエリでカラムのエイリアスが使用されている場合は、そのエイリアスが返されます。

備考

結果セットでは、すべてのカラムに名前があるとは限らないほか、すべてのカラム名がユニークであるとは限りません。エイリアスを使用している場合は、カラムの名前がエイリアスになります。

GetColumnSQLName メソッドは、非エイリアス・カラム、非計算カラムの GetColumnName とは異なります。GetColumnSQLName は、常にカラムの名前を返します (プレフィクスとしてテーブル名は付きません)。この動作は、他の ADO.NET プロバイダの動作によく似ていますが、ユニークでない名前を生成する可能性が高くなります。

カラムの ID とカウントは、スキーマのアップグレード中に変更されることがあります。カラムを正しく識別するには、名前でアクセスするか、キャッシュされている ID とカウントをスキーマのアップグレード後にリフレッシュします。

参照

- 「ULCursorSchema クラス」 233 ページ
- 「ULCursorSchema メンバ」 233 ページ
- 「ColumnCount プロパティ」 234 ページ
- 「GetColumnName メソッド」 236 ページ
- 「ColumnCount プロパティ」 234 ページ

GetColumnScale メソッド

カラムが数値カラム (SQL NUMERIC 型) の場合は、指定されたカラム ID によって識別されたカラムの位取りを返します。

構文

Visual Basic

```
Public Function GetColumnScale( _
  ByVal columnID As Integer _
) As Integer
```

C#

```
public int GetColumnScale(
  int columnID
);
```

パラメータ

- **columnID**　カラムの ID。値は、[0,ColumnCount-1] の範囲内であることが必要です。

戻り値

指定された数値カラムの位取り。

参照

- 「ULCursorSchema クラス」 233 ページ
- 「ULCursorSchema メンバ」 233 ページ
- 「GetColumnULDbType メソッド」 240 ページ
- 「ColumnCount プロパティ」 234 ページ

GetColumnSize メソッド

カラムがサイズ指定されたカラム (SQL BINARY 型または CHAR 型) の場合は、指定されたカラム ID によって識別されたカラムのサイズを返します。

構文

Visual Basic

```
Public Function GetColumnSize( _
  ByVal columnID As Integer _
) As Integer
```

C#
public int **GetColumnSize(**
  int *columnID*
);

パラメータ

- **columnID**　カラムの ID。値は、[0,ColumnCount-1] の範囲内であることが必要です。

戻り値

指定された、サイズ指定されたカラムのサイズ。

参照

- 「ULCursorSchema クラス」 233 ページ
- 「ULCursorSchema メンバ」 233 ページ
- 「GetColumnULDbType メソッド」 240 ページ
- 「ColumnCount プロパティ」 234 ページ

GetColumnULDbType メソッド

指定されたカラム ID によって識別されたカラムの Ultra Light.NET データ型を返します。

構文

Visual Basic
Public Function **GetColumnULDbType(_**
  ByVal *columnID* As Integer _
) As ULDbType

C#
public ULDbType **GetColumnULDbType(**
  int *columnID*
);

パラメータ

- **columnID**　カラムの ID。値は、[0,ColumnCount-1] の範囲内であることが必要です。

戻り値

ULDbType 列挙整数。

参照

- 「ULCursorSchema クラス」 233 ページ
- 「ULCursorSchema メンバ」 233 ページ
- 「ColumnCount プロパティ」 234 ページ
- 「ColumnCount プロパティ」 234 ページ
- 「ULDbType 列挙体」 317 ページ

GetSchemaTable メソッド

ULDataReader のカラムのスキーマが記述された System.Data.DataTable を返します。

構文

Visual Basic
Public Function **GetSchemaTable()** As DataTable

C#
public DataTable **GetSchemaTable();**

戻り値

カラムのスキーマが記述された System.Data.DataTable。

備考

詳細については、「GetSchemaTable メソッド」 303 ページを参照してください。

参照

- 「ULCursorSchema クラス」 233 ページ
- 「ULCursorSchema メンバ」 233 ページ
- DataTable
- 「ULDataReader クラス」 276 ページ

ULDataAdapter クラス

System.Data.DataSet に入力したりデータベースを更新したりするために使用する一連のコマンドとデータベース接続を表します。このクラスは継承できません。

構文

Visual Basic
Public NotInheritable Class **ULDataAdapter**
  Inherits DbDataAdapter

C#
public sealed class **ULDataAdapter**: DbDataAdapter

備考

System.Data.DataSet は、Ultra Light データベースとは別に、データをオフラインで処理するための方法を提供します。ULDataAdapter は、System.Data.DataSet を一連の SQL 文に関連付けるためのメソッドを提供します。

Ultra Light はローカルのデータベースであり、Mobile Link には競合解決があるため、ULDataAdapter の使用は制限されています。ほとんどの場合、ULDataReader または ULTable を使用すると、より効率的にデータにアクセスできます。

継承： System.Data.Common.DbDataAdapter

実装： System.Data.IDbDataAdapter、System.Data.IDataAdapter、System.IDisposable

参照

- 「ULDataAdapter メンバ」 242 ページ
- DataSet
- 「ULDataReader クラス」 276 ページ
- 「ULTable クラス」 548 ページ
- DbDataAdapter
- IDbDataAdapter
- IDataAdapter
- IDisposable

ULDataAdapter メンバ

パブリック・コンストラクタ

メンバ名	説明
「ULDataAdapter コンストラクタ」 245 ページ	「ULDataAdapter クラス」 242 ページの新しいインスタンスを初期化します。

パブリック・プロパティ

メンバ名	説明
AcceptChangesDuringFill (DataAdapter から継承)	Fill 操作の実行時に DataTable に追加された後で DataRow で DataRow.AcceptChanges を呼び出すかどうかを示す値を取得または設定します。
AcceptChangesDuringUpdate (DataAdapter から継承)	DataAdapter.Update の実行時に DataRow.AcceptChanges を呼び出すかどうかを示す値を取得または設定します。
ContinueUpdateOnError (DataAdapter から継承)	ローの更新時にエラーが検出されたときに例外を生成するかどうかを指定する値を取得または設定します。
「DeleteCommand プロパティ」 248 ページ	System.Data.DataSet で削除されたローに該当するデータベース内のローを削除するために、DbDataAdapter.Update(System.Data.DataSet) が呼び出されたときにデータベースに対して実行される ULCommand オブジェクトを指定します。
FillLoadOption (DataAdapter から継承)	アダプタが DbDataReader から DataTable を入力する方法を決定する LoadOption を取得または設定します。
「InsertCommand プロパティ」 249 ページ	System.Data.DataSet で挿入されたローに該当するデータベース内のローを挿入するために、DbDataAdapter.Update(System.Data.DataSet) が呼び出されたときにデータベースに対して実行される ULCommand オブジェクトを指定します。
MissingMappingAction (DataAdapter から継承)	受信データと一致するテーブルまたはカラムがない場合に実行するアクションを決定します。
MissingSchemaAction (DataAdapter から継承)	既存の DataSet スキーマが受信データと一致しない場合に実行するアクションを決定します。
ReturnProviderSpecificTypes (DataAdapter から継承)	DataAdapter.Fill がプロバイダ固有の値と CLS に準拠する共通の値のどちらを返すかを示す値を取得または設定します。

メンバ名	説明
「SelectCommand プロパティ」 249 ページ	System.Data.Common.DbDataAdapter.Fill(System.Data.DataSet) または System.Data.Common.DbDataAdapter.FillSchema(System.Data.DataSet,System.Data.SchemaType) の実行時に、System.Data.DataSet にコピーするための結果セットをデータベースから取得するために使用される ULCommand を指定します。
「TableMappings プロパティ」 250 ページ	ソース・テーブルと System.Data.DataTable 間のマスタ・マッピングを提供するコレクションを返します。
UpdateBatchSize (DbDataAdapter から継承)	バッチ・プロセス・サポートを有効または無効にする値を取得または設定し、バッチで実行可能なコマンド数を指定します。
「UpdateCommand プロパティ」 251 ページ	System.Data.DataSet で更新されたローに該当するデータベース内のローを更新するために、System.Data.Common.DbDataAdapter.Update(System.Data.DataSet) が呼び出されたときにデータベースに対して実行される ULCommand オブジェクトを指定します。

パブリック・メソッド

メンバ名	説明
Fill (DbDataAdapter から継承)	DataSet または DataTable に入力します。
FillSchema (DbDataAdapter から継承)	DataTable を DataSet に追加し、スキーマがデータ・ソースのスキーマと一致するように設定します。
「GetFillParameters メソッド」 252 ページ	SELECT 文の実行時にユーザによって設定されるパラメータを返します。
ResetFillLoadOption (DataAdapter から継承)	DataAdapter.FillLoadOption をデフォルトのステータスにリセットし、DataAdapter.Fill で DataAdapter.AcceptChangesDuringFill が優先されるようにします。
ShouldSerializeAcceptChangesDuringFill (DataAdapter から継承)	DataAdapter.AcceptChangesDuringFill を保持するかどうかを決定します。
ShouldSerializeFillLoadOption (DataAdapter から継承)	DataAdapter.FillLoadOption を保持するかどうかを決定します。

メンバ名	説明
Update (DbDataAdapter から継承)	DataSet で挿入、更新、削除されたローに対して、それぞれ INSERT、UPDATE、DELETE 文を呼び出します。

パブリック・イベント

メンバ名	説明
FillError (DataAdapter から継承)	Fill 操作の実行時にエラーが発生したときに返されます。
「RowUpdated イベント」 252 ページ	データ・ソースに対してコマンドが実行された後の更新時に発生します。更新が試みられると、イベントが発生します。
「RowUpdating イベント」 253 ページ	データ・ソースに対してコマンドが実行される前の更新時に発生します。更新が試みられると、イベントが発生します。

参照

- 「ULDataAdapter クラス」 242 ページ
- DataSet
- 「ULDataReader クラス」 276 ページ
- 「ULTable クラス」 548 ページ
- DbDataAdapter
- IDbDataAdapter
- IDataAdapter
- IDisposable

ULDataAdapter コンストラクタ

「ULDataAdapter クラス」 242 ページの新しいインスタンスを初期化します。

ULDataAdapter() コンストラクタ

ULDataAdapter オブジェクトを初期化します。

構文

Visual Basic
Public Sub **New()**

C#
public **ULDataAdapter();**

参照

- 「ULDataAdapter クラス」 242 ページ
- 「ULDataAdapter メンバ」 242 ページ
- 「ULDataAdapter コンストラクタ」 245 ページ
- 「ULDataAdapter(ULCommand) コンストラクタ」 246 ページ
- 「ULDataAdapter(String, ULConnection) コンストラクタ」 246 ページ
- 「ULDataAdapter(String, String) コンストラクタ」 247 ページ

ULDataAdapter(ULCommand) コンストラクタ

ULDataAdapter オブジェクトを、指定された SELECT 文で初期化します。

構文

Visual Basic

```
Public Sub New( _
  ByVal selectCommand As ULCommand _
)
```

C#

```
public  ULDataAdapter(
  ULCommand selectCommand
);
```

パラメータ

- **selectCommand**　System.Data.DataSet に配置するためのレコードをデータ・ソースから選択するために System.Data.Common.DbDataAdapter.Fill(System.Data.DataSet) の実行時に使用される ULCommand オブジェクト。

参照

- 「ULDataAdapter クラス」 242 ページ
- 「ULDataAdapter メンバ」 242 ページ
- 「ULDataAdapter コンストラクタ」 245 ページ
- 「ULDataAdapter() コンストラクタ」 245 ページ
- 「ULDataAdapter(String, ULConnection) コンストラクタ」 246 ページ
- 「ULDataAdapter(String, String) コンストラクタ」 247 ページ
- 「ULCommand クラス」 91 ページ
- DbDataAdapter.Fill
- DataSet

ULDataAdapter(String, ULConnection) コンストラクタ

ULDataAdapter オブジェクトを、指定された SELECT 文と接続で初期化します。

構文

Visual Basic
```
Public Sub New( _
  ByVal selectCommandText As String, _
  ByVal selectConnection As ULConnection _
)
```

C#
```
public ULDataAdapter(
  string selectCommandText,
  ULConnection selectConnection
);
```

パラメータ

- **selectCommandText** ULDataAdapter の ULDataAdapter.SelectCommand によって使用される SELECT 文。
- **selectConnection** データベースへの接続を定義する ULConnection オブジェクト。

参照

- 「ULDataAdapter クラス」 242 ページ
- 「ULDataAdapter メンバ」 242 ページ
- 「ULDataAdapter コンストラクタ」 245 ページ
- 「ULDataAdapter() コンストラクタ」 245 ページ
- 「ULDataAdapter(ULCommand) コンストラクタ」 246 ページ
- 「ULDataAdapter(String, String) コンストラクタ」 247 ページ
- 「SelectCommand プロパティ」 249 ページ
- 「ULConnection クラス」 139 ページ

ULDataAdapter(String, String) コンストラクタ

ULDataAdapter オブジェクトを、指定された SELECT 文と接続文字列で初期化します。

構文

Visual Basic
```
Public Sub New( _
  ByVal selectCommandText As String, _
  ByVal selectConnectionString As String _
)
```

C#
```
public ULDataAdapter(
  string selectCommandText,
  string selectConnectionString
);
```

パラメータ

- **selectCommandText** ULDataAdapter の ULDataAdapter.SelectCommand によって使用される SELECT 文。

- **selectConnectionString**　Ultra Light.NET データベースの接続文字列。

参照

- 「ULDataAdapter クラス」 242 ページ
- 「ULDataAdapter メンバ」 242 ページ
- 「ULDataAdapter コンストラクタ」 245 ページ
- 「ULDataAdapter() コンストラクタ」 245 ページ
- 「ULDataAdapter(ULCommand) コンストラクタ」 246 ページ
- 「ULDataAdapter(String, ULConnection) コンストラクタ」 246 ページ
- 「SelectCommand プロパティ」 249 ページ

DeleteCommand プロパティ

System.Data.DataSet で削除されたローに該当するデータベース内のローを削除するために、DbDataAdapter.Update(System.Data.DataSet) が呼び出されたときにデータベースに対して実行される ULCommand オブジェクトを指定します。

構文

Visual Basic
Public Property **DeleteCommand** As ULCommand

C#
public ULCommand **DeleteCommand** { get; set; }

プロパティ値

System.Data.DataSet で削除されたローに該当するデータベース内のローを削除するために実行される ULCommand オブジェクト。

備考

DeleteCommand が既存の ULCommand オブジェクトに割り当てられる場合、ULCommand オブジェクトのクローンは作成されません。DeleteCommand は、既存の ULCommand への参照を保持します。

これは、System.Data.IDbDataAdapter.DeleteCommand と System.Data.Common.DbDataAdapter.DeleteCommand が厳密に型指定されたものです。

参照

- 「ULDataAdapter クラス」 242 ページ
- 「ULDataAdapter メンバ」 242 ページ
- 「ULCommand クラス」 91 ページ
- DbDataAdapter.Update
- DataSet
- 「ULCommand クラス」 91 ページ
- IDbDataAdapter.DeleteCommand
- DbDataAdapter.DeleteCommand

InsertCommand プロパティ

System.Data.DataSet で挿入されたローに該当するデータベース内のローを挿入するために、DbDataAdapter.Update(System.Data.DataSet) が呼び出されたときにデータベースに対して実行される ULCommand オブジェクトを指定します。

構文

Visual Basic
Public Property **InsertCommand** As ULCommand

C#
public ULCommand **InsertCommand** { get; set; }

プロパティ値

System.Data.DataSet で挿入されたローに該当するデータベース内のローを挿入するために実行される ULCommand オブジェクト。

備考

InsertCommand が既存の ULCommand オブジェクトに割り当てられる場合、ULCommand オブジェクトのクローンは作成されません。InsertCommand は、既存の ULCommand への参照を保持します。

これは、System.Data.IDbDataAdapter.InsertCommand と System.Data.Common.DbDataAdapter.InsertCommand が厳密に型指定されたものです。

参照

- 「ULDataAdapter クラス」 242 ページ
- 「ULDataAdapter メンバ」 242 ページ
- 「ULCommand クラス」 91 ページ
- DbDataAdapter.Update
- DataSet
- 「ULCommand クラス」 91 ページ
- IDbDataAdapter.InsertCommand
- DbDataAdapter.InsertCommand

SelectCommand プロパティ

System.Data.Common.DbDataAdapter.Fill(System.Data.DataSet) または System.Data.Common.DbDataAdapter.FillSchema(System.Data.DataSet,System.Data.SchemaType) の実行時に、System.Data.DataSet にコピーするための結果セットをデータベースから取得するために使用される ULCommand を指定します。

構文

Visual Basic
Public Property **SelectCommand** As ULCommand

C#
public ULCommand **SelectCommand** { get; set; }

プロパティ値

System.Data.DataSet を設定するために実行される ULCommand オブジェクト。

備考

SelectCommand が既存の ULCommand オブジェクトに割り当てられる場合、ULCommand オブジェクトのクローンは作成されません。SelectCommand は、既存の ULCommand への参照を保持します。

SelectCommand がローを返さない場合、System.Data.DataSet にテーブルが追加されず、例外も発生しません。SELECT 文は、ULDataAdapter(ULCommand)、ULDataAdapter(String,ULConnection)、ULDataAdapter(String,String) のコンストラクタにも指定できます。

これは、System.Data.IDbDataAdapter.SelectCommand と System.Data.Common.DbDataAdapter.SelectCommand が厳密に型指定されたものです。

参照

- 「ULDataAdapter クラス」 242 ページ
- 「ULDataAdapter メンバ」 242 ページ
- 「ULCommand クラス」 91 ページ
- DataSet
- DbDataAdapter.Fill
- DbDataAdapter.FillSchema
- 「ULDataAdapter(ULCommand) コンストラクタ」 246 ページ
- 「ULDataAdapter(String, ULConnection) コンストラクタ」 246 ページ
- 「ULDataAdapter(String, String) コンストラクタ」 247 ページ
- IDbDataAdapter.SelectCommand
- DbDataAdapter.SelectCommand

TableMappings プロパティ

ソース・テーブルと System.Data.DataTable 間のマスタ・マッピングを提供するコレクションを返します。

構文

Visual Basic
Public Readonly Property **TableMappings** As DataTableMappingCollection

C#
public DataTableMappingCollection **TableMappings** { get;}

プロパティ値

ソース・テーブルと System.Data.DataTables 間のマスタ・マッピングを提供する System.Data.Common.DataTableMapping オブジェクトのコレクション。デフォルト値は空のコレクションです。

備考

変更を調整する場合、ULDataAdapter は System.Data.Common.DataTableMappingCollection コレクションを使用して、データ・ソースによって使用されるカラム名を、System.Data.DataSet によって使用されるカラム名に関連付けます。

これは、System.Data.IDataAdapter.TableMappings が厳密に型指定されたものです。

参照

- 「ULDataAdapter クラス」 242 ページ
- 「ULDataAdapter メンバ」 242 ページ
- DataTable
- DataTableMapping
- DataTable
- DataTableMappingCollection
- DataSet
- IDataAdapter.TableMappings

UpdateCommand プロパティ

System.Data.DataSet で更新されたローに該当するデータベース内のローを更新するために、System.Data.Common.DbDataAdapter.Update(System.Data.DataSet) が呼び出されたときにデータベースに対して実行される ULCommand オブジェクトを指定します。

構文

Visual Basic
Public Property **UpdateCommand** As ULCommand

C#
public ULCommand **UpdateCommand** { get; set; }

プロパティ値

System.Data.DataSet で更新されたローに該当するデータベース内のローを更新するために実行される ULCommand オブジェクト。

備考

UpdateCommand が既存の ULCommand オブジェクトに割り当てられる場合、ULCommand オブジェクトのクローンは作成されません。UpdateCommand は、既存の ULCommand への参照を保持します。

このコマンドを実行するとローが返される場合、ULCommand オブジェクトの ULCommand.UpdatedRowSource の設定方法によっては、これらのローが System.Data.DataSet にマージされることがあります。

これは、System.Data.IDbDataAdapter.UpdateCommand と System.Data.Common.DbDataAdapter.DeleteCommand が厳密に型指定されたものです。

参照

- 「ULDataAdapter クラス」 242 ページ
- 「ULDataAdapter メンバ」 242 ページ
- 「ULCommand クラス」 91 ページ
- DataSet
- DbDataAdapter.Update
- 「UpdatedRowSource プロパティ」 105 ページ
- IDbDataAdapter.UpdateCommand
- DbDataAdapter.DeleteCommand

GetFillParameters メソッド

SELECT 文の実行時にユーザによって設定されるパラメータを返します。

構文

Visual Basic
Public Function **GetFillParameters()** As ULParameter

C#
public ULParameter **GetFillParameters();**

戻り値

ユーザによって設定されたパラメータが含まれる ULParameter オブジェクトの配列。

備考

これは、System.Data.Common.DbDataAdapter.GetFillParameters が厳密に型指定されたものです。

参照

- 「ULDataAdapter クラス」 242 ページ
- 「ULDataAdapter メンバ」 242 ページ
- DbDataAdapter.GetFillParameters
- 「ULParameter クラス」 375 ページ

RowUpdated イベント

データ・ソースに対してコマンドが実行された後の更新時に発生します。更新が試みられると、イベントが発生します。

構文

Visual Basic
Public Event **RowUpdated** As ULRowUpdatedEventHandler

C#
public event ULRowUpdatedEventHandler **RowUpdated**;

備考

ロー更新イベントを処理するには、ULRowUpdatedEventHandler デリゲートを作成し、このイベントにアタッチする必要があります。

イベント・データ

- **Command** DbDataAdapter.Update(System.Data.DataRow[],System.Data.Common.DataTableMapping) が呼び出されたときに実行された ULCommand を返します。
- **RecordsAffected** SQL 文の実行によって変更、挿入、または削除されたローの数を返します。SELECT 文の場合、この値は -1 です。
- **Command** DbDataAdapter.Update の呼び出し時に実行される IDbCommand を取得します。
- **Errors** RowUpdatedEventArgs.Command の実行時に .NET Framework データ・プロバイダによって生成されるエラーを取得します。
- **Row** DbDataAdapter.Update によって送信された DataRow を取得します。
- **RowCount** 更新済みレコードのバッチで処理されたローの数を取得します。
- **StatementType** 実行された SQL 文のタイプを取得します。
- **Status** RowUpdatedEventArgs.Command の UpdateStatus を取得します。
- **TableMapping** DbDataAdapter.Update によって送信された DataTableMapping を取得します。

参照

- 「ULDataAdapter クラス」 242 ページ
- 「ULDataAdapter メンバ」 242 ページ
- 「ULRowUpdatedEventHandler デリゲート」 453 ページ

RowUpdating イベント

データ・ソースに対してコマンドが実行される前の更新時に発生します。更新が試みられると、イベントが発生します。

構文

Visual Basic
Public Event **RowUpdating** As ULRowUpdatingEventHandler

C#
public event ULRowUpdatingEventHandler **RowUpdating**;

備考

ロー更新イベントを処理するには、ULRowUpdatingEventHandler デリゲートを作成し、このイベントにアタッチする必要があります。

イベント・データ

- **Command** DbDataAdapter.Update(System.Data.DataRow[],System.Data.Common.DataTableMapping) の実行時に実行される ULCommand を指定します。
- **Command** DbDataAdapter.Update 操作時に実行される IDbCommand を取得します。
- **Errors** RowUpdatedEventArgs.Command の実行時に .NET Framework データ・プロバイダによって生成されるエラーを取得します。
- **Row** 挿入、更新、または削除操作の一部としてサーバに送信される DataRow を取得します。
- **StatementType** 実行する SQL 文のタイプを取得します。
- **Status** RowUpdatedEventArgs.Command の UpdateStatus を取得または設定します。
- **TableMapping** DbDataAdapter.Update によって送信する DataTableMapping を取得します。

参照

- 「ULDataAdapter クラス」 242 ページ
- 「ULDataAdapter メンバ」 242 ページ
- 「ULRowUpdatedEventHandler デリゲート」 453 ページ

ULDatabaseManager クラス

UL 拡張 : 同期リスナと Ultra Light.NET のランタイム・タイプを管理します。ULDatabaseManager クラスを使用すると、Ultra Light.NET データベースを削除できます。このクラスは継承できません。

構文

Visual Basic
Public NotInheritable Class **ULDatabaseManager**

C#
public sealed class **ULDatabaseManager**

備考

このクラスはシングルトン・クラスであり、その唯一のインスタンスには、静的な (Visual Basic の Shared) ULConnection.DatabaseManager のみを通じてアクセスできます。

Ultra Light.NET の Ultra Light エンジンのランタイムを使用するには、ULDatabaseManager.RuntimeType を適切な値に設定してから、他の Ultra Light.NET API を使用します。

例

次の例では、Ultra Light エンジンのランタイムが選択され、接続が作成されます。

```
' Visual Basic
ULDatabaseManager.RuntimeType = ULRuntimeType.UL_ENGINE_CLIENT
Dim conn As ULConnection = new ULConnection
' The RuntimeType is now locked

// C#
ULDatabaseManager.RuntimeType = ULRuntimeType.UL_ENGINE_CLIENT;
ULConnection conn = new ULConnection();
// The RuntimeType is now locked
```

参照

- 「ULDatabaseManager メンバ」 256 ページ
- 「DatabaseManager プロパティ」 149 ページ
- 「RuntimeType プロパティ」 256 ページ

ULDatabaseManager メンバ

パブリック・プロパティ

メンバ名	説明
「RuntimeType プロパティ」 256 ページ	Ultra Light.NET ランタイム・タイプを指定します。ランタイム・タイプを選択してから、その他の Ultra Light.NET API を使用してください。

パブリック・メソッド

メンバ名	説明
「CreateDatabase メソッド」 257 ページ	新しい Ultra Light データベースを作成します。
「DropDatabase メソッド」 259 ページ	指定されたデータベースを削除します。 接続が開かれているデータベースを削除することはできません。
「SetActiveSyncListener メソッド」 260 ページ	ActiveSync 用 Mobile Link プロバイダからの ActiveSync の呼び出しを処理するリスナ・オブジェクトを指定します。
「SetGlobalListener メソッド」 261 ページ	グローバル同期および SQL パススルー・メッセージの処理に使用するリスナ・オブジェクトを指定します。
「SetServerSyncListener メソッド」 262 ページ	指定されたサーバ同期メッセージの処理に使用するリスナ・オブジェクトを指定します。
「SignalSyncIsComplete メソッド」 263 ページ	ActiveSync 用の Mobile Link プロバイダに対して、アプリケーションが同期を完了したことを通知します。
「ValidateDatabase メソッド」 263 ページ	データベースで低レベルのインデックス検証を実行します。

参照

- 「ULDatabaseManager クラス」 255 ページ
- 「DatabaseManager プロパティ」 149 ページ
- 「RuntimeType プロパティ」 256 ページ

RuntimeType プロパティ

Ultra Light.NET ランタイム・タイプを指定します。ランタイム・タイプを選択してから、その他の Ultra Light.NET API を使用してください。

構文

Visual Basic
Public Shared Property **RuntimeType** As ULRuntimeType

C#
public static ULRuntimeType **RuntimeType** { get; set; }

プロパティ値

アンマネージ Ultra Light .NET ランタイムのタイプを示す ULRuntimeType 値。

例

次の例では、Ultra Light エンジンのランタイムが選択され、接続が作成されます。

```
' Visual Basic
ULDatabaseManager.RuntimeType = ULRuntimeType.UL_ENGINE_CLIENT
Dim conn As ULConnection = new ULConnection
' The RuntimeType is now locked

// C#
ULDatabaseManager.RuntimeType = ULRuntimeType.UL_ENGINE_CLIENT;
ULConnection conn = new ULConnection();
// The RuntimeType is now locked
```

参照

- 「ULDatabaseManager クラス」 255 ページ
- 「ULDatabaseManager メンバ」 256 ページ
- 「ULRuntimeType 列挙体」 458 ページ

CreateDatabase メソッド

新しい Ultra Light データベースを作成します。

構文

Visual Basic
Public Sub **CreateDatabase(_**
  ByVal *connString* As String, _
  ByVal *collationData* As Byte(), _
  ByVal *createParms* As String _
)

C#
public void **CreateDatabase(**
  string *connString*,
  byte[] *collationData*,
  string *createParms*
);

パラメータ

- **connString**　キーワードと値の組み合わせがセミコロンで区切られたリスト形式の、データベースを指定するためのパラメータ。詳細については、「ULConnectionParms クラス」 189 ページを参照してください。
- **collationData**　データベースが文字列を格納して比較する方法を指定する照合データ。
- **createParms**　キーワードと値の組み合わせがセミコロンで区切られたリスト形式の、新しいデータベースを設定するためのパラメータ。詳細については、「ULCreateParms クラス」 221 ページを参照してください。

備考

照合を指定するには、SQL Anywhere インストール・ディレクトリの UltraLite¥Collations¥cs (C# プロジェクトの場合) または UltraLite¥Collations¥vb.net (Visual Basic プロジェクトの場合) サブディレクトリから、適切な照合データ・ソースを含める必要があります。プロジェクトに追加したら、照合の Data プロパティを使用して、CreateDatabase() メソッドに対して照合データを提供できます。

例

次のコードでは、Windows Mobile デバイス上にデータベース ¥UltraLite¥MyDatabase.udb を作成し、接続を開きます。

```
' Visual Basic
Dim openParms As ULConnectionParms = New ULConnectionParms
openParms.DatabaseOnCE = "¥UltraLite¥MyDatabase.udb"
' Assumes file UltraLite¥Collations¥vb.net¥coll_1250LATIN2.vb is
' also compiled in the current project
ULConnection.DatabaseManager.CreateDatabase( _
    openParms.ToString(), _
    iAnywhere.UltraLite.Collations.Collation_1250LATIN2.Data, _
    "" _
  )
Dim conn As ULConnection = _
  New ULConnection( openParms.ToString() )
conn.Open()


// C#
ULConnectionParms openParms = new ULConnectionParms();
openParms.DatabaseOnCE = @"¥UltraLite¥MyDatabase.udb";
// Assumes file UltraLite¥Collations¥cs¥coll_1250LATIN2.cs is
// also compiled in the current project
ULConnection.DatabaseManager.CreateDatabase(
    openParms.ToString(),
    iAnywhere.UltraLite.Collations.Collation_1250LATIN2.Data,
    ""
  );
ULConnection conn = new ULConnection( openParms.ToString() );
conn.Open();
```

参照

- 「ULDatabaseManager クラス」 255 ページ
- 「ULDatabaseManager メンバ」 256 ページ
- 「Open メソッド」 177 ページ

DropDatabase メソッド

指定されたデータベースを削除します。

接続が開かれているデータベースを削除することはできません。

構文

Visual Basic

```
Public Sub DropDatabase( _
  ByVal connString As String _
)
```

C#

```
public void DropDatabase(
  string  connString
);
```

パラメータ

- **connString**　キーワードと値の組み合わせがセミコロンで区切られたリスト形式の、データベースを指定するためのパラメータ。詳細については、「ULConnectionParms クラス」 189 ページを参照してください。

例

次のコードでは、Windows Mobile デバイス上にデータベース ¥UltraLite¥MyDatabase.udb を作成し、接続を開きます。

```
' Visual Basic
Dim connParms As ULConnectionParms = New ULConnectionParms
connParms.DatabaseOnCE = "¥UltraLite¥MyDatabase.udb"
ULConnection.DatabaseManager.DropDatabase( _
   connParms.ToString() _
 )


// C#
ULConnectionParms connParms = new ULConnectionParms();
connParms.DatabaseOnCE = @"¥UltraLite¥MyDatabase.udb";
ULConnection.DatabaseManager.DropDatabase(
   connParms.ToString()
 );
ULConnection conn = new ULConnection( openParms.ToString() );
conn.Open();
```

参照

- 「ULDatabaseManager クラス」 255 ページ
- 「ULDatabaseManager メンバ」 256 ページ
- 「Open メソッド」 177 ページ

SetActiveSyncListener メソッド

ActiveSync 用 Mobile Link プロバイダからの ActiveSync の呼び出しを処理するリスナ・オブジェクトを指定します。

構文

Visual Basic

```
Public Sub SetActiveSyncListener( _
  ByVal appClassName As String, _
  ByVal listener As ULActiveSyncListener _
)
```

C#

```
public void SetActiveSyncListener(
  string  appClassName,
  ULActiveSyncListener listener
);
```

パラメータ

- **appClassName**　アプリケーションのユニークなクラス名。これは、アプリケーションが ActiveSync で使用されるように登録されているときに使用されるクラス名です。
- **listener**　ULActiveSyncListener オブジェクト。前のリスナを削除するには、NULL (Visual Basic の Nothing) を使用します。

備考

パラメータ *appClassName* は、アプリケーションの識別に使用されるユニークな識別子です。アプリケーションは、一度に 1 つの *appClassName* のみを使用できます。リスナが、特定の *appClassName* に登録されているときは、別の *appClassName* で SetServerSyncListener または SetActiveSyncListener を呼び出すと失敗します。

ActiveSync リスナを削除するには、*listener* パラメータとして NULL 参照 (Visual Basic の Nothing) を使用して SetActiveSyncListener を呼び出します。

すべてのリスナを削除するには、すべてのパラメータに NULL 参照 (Visual Basic の Nothing) を使用して SetServerSyncListener を呼び出します。

アプリケーションは、すべてのリスナを削除してから終了する必要があります。

例

SetActiveSyncListener の例については、「ActiveSyncInvoked メソッド」 57 ページを参照してください。

参照

- 「ULDatabaseManager クラス」 255 ページ
- 「ULDatabaseManager メンバ」 256 ページ
- 「SetServerSyncListener メソッド」 262 ページ
- 「SetActiveSyncListener メソッド」 260 ページ
- 「ULActiveSyncListener インタフェース」 57 ページ
- 「ActiveSyncInvoked メソッド」 57 ページ

SetGlobalListener メソッド

グローバル同期および SQL パススルー・メッセージの処理に使用するリスナ・オブジェクトを指定します。

構文

Visual Basic

```
Public Sub SetGlobalListener( _
  ByVal syncListener As ULSyncProgressListener, _
  ByVal sqlListener As ULSqlPassthroughProgressListener _
)
```

C#

```
public void SetGlobalListener(
  ULSyncProgressListener syncListener,
  ULSqlPassthroughProgressListener sqlListener
);
```

パラメータ

- **syncListener**　SyncProgressed() を実装する ULSyncProgressListener オブジェクト。グローバル同期メッセージに対して呼び出されます。
- **sqlListener**　ScriptProgressed() を実装する ULSqlPassthroughProgressListener オブジェクト。SQL パススルー・スクリプトが実行されるたびに呼び出されます。

備考

SQL 文 SYNCHRONIZE profileName が実行されると、その進行状況のメッセージが NULL (Microsoft Visual Basic の場合は Nothing) でない場合は syncListener にルーティングされます。

データベースに接続すると、複数の SQL スクリプトが使用可能になり自動的に実行されることがあります。また、スクリプトは後続の同期にパス・ダウンされ、ULConnection.ExecuteSQLPassthroughScripts() で直接実行されることもあります。いずれの場合も、スクリプトの進行状況メッセージは、NULL (Microsoft Visual Basic の場合は Nothing) でない場合は sqlListener にルーティングされます。

いずれかのリスナを削除するには、SetGlobalListener の呼び出しで NULL 参照を渡します。11.0 では、アプリケーションは終了する前にリスナを削除する必要がなくなりました。

参照

- 「ULDatabaseManager クラス」 255 ページ
- 「ULDatabaseManager メンバ」 256 ページ
- 「ULSyncProgressListener インタフェース」 538 ページ
- 「ULSqlPassthroughProgressListener インタフェース」 477 ページ
- 「ExecuteSQLPassthroughScripts メソッド」 164 ページ

SetServerSyncListener メソッド

指定されたサーバ同期メッセージの処理に使用するリスナ・オブジェクトを指定します。

構文

Visual Basic

```
Public Sub SetServerSyncListener( _
  ByVal messageName As String, _
  ByVal appClassName As String, _
  ByVal listener As ULServerSyncListener _
)
```

C#

```
public void SetServerSyncListener(
  string  messageName,
  string  appClassName,
  ULServerSyncListener listener
);
```

パラメータ

- **messageName**　メッセージの名前。
- **appClassName**　アプリケーションのユニークなクラス名。これは、アプリケーションの識別に使用されるユニークな識別子です。
- **listener**　ULServerSyncListener オブジェクト。前のリスナを削除するには、NULL (Visual Basic の Nothing) を使用します。

備考

パラメータ *appClassName* は、アプリケーションの識別に使用されるユニークな識別子です。アプリケーションは、一度に 1 つの *appClassName* のみを使用できます。リスナが、特定の *appClassName* に登録されているときは、別の *appClassName* で SetServerSyncListener または SetActiveSyncListener を呼び出すと失敗します。

特定のメッセージのリスナを削除するには、*listener* パラメータとして NULL 参照 (Visual Basic の Nothing) を使用して SetServerSyncListener を呼び出します。

すべてのリスナを削除するには、すべてのパラメータに NULL 参照 (Visual Basic の Nothing) を使用して SetServerSyncListener を呼び出します。

アプリケーションは、すべてのリスナを削除してから終了する必要があります。

例

SetServerSyncListener の例については、「ServerSyncInvoked メソッド」 459 ページを参照してください。

参照

- 「ULDatabaseManager クラス」 255 ページ
- 「ULDatabaseManager メンバ」 256 ページ
- 「SetServerSyncListener メソッド」 262 ページ
- 「SetActiveSyncListener メソッド」 260 ページ
- 「ServerSyncInvoked メソッド」 459 ページ

SignalSyncIsComplete メソッド

ActiveSync 用の Mobile Link プロバイダに対して、アプリケーションが同期を完了したことを通知します。

構文

Visual Basic
Public Sub **SignalSyncIsComplete()**

C#
public void **SignalSyncIsComplete();**

例

SignalSyncIsComplete の例については、「ActiveSyncInvoked メソッド」 57 ページを参照してください。

参照

- 「ULDatabaseManager クラス」 255 ページ
- 「ULDatabaseManager メンバ」 256 ページ
- 「SignalSyncIsComplete メソッド」 263 ページ

ValidateDatabase メソッド

データベースで低レベルのインデックス検証を実行します。

構文

Visual Basic
```
Public Sub ValidateDatabase( _
  ByVal start_parms As String, _
  ByVal how As ULDBValid _
)
```

C#
public void **ValidateDatabase(**

```
    string  start_parms,
    ULDBValid how
);
```

パラメータ

- **start_parms**　キーワードと値の組み合わせがセミコロンで区切られたリスト形式の、データベースを指定するためのパラメータ。詳細については、「ULConnectionParms クラス」 189 ページを参照してください。
- **how**　データベースの検証方法。詳細については、「ULDBValid 列挙体」 321 ページを参照してください。

例

次のコードでは、Windows Mobile のデータベース ¥UltraLite¥MyDatabase.udb のインデックスを検証します。

```
' Visual Basic
Dim openParms As ULConnectionParms = New ULConnectionParms
openParms.DatabaseOnCE = "¥UltraLite¥MyDatabase.udb"
ULConnection.DatabaseManager.ValidateDatabase( _
    openParms.ToString(), iAnywhere.Data.UltraLite.ULVF_INDEX )
```

```
// C#
ULConnectionParms openParms = new ULConnectionParms();
openParms.DatabaseOnCE = @"¥UltraLite¥MyDatabase.udb";
ULConnection.DatabaseManager.ValidateDatabase(
    openParms.ToString(), iAnywhere.Data.UltraLite.ULVF_INDEX );
```

参照

- 「ULDatabaseManager クラス」 255 ページ
- 「ULDatabaseManager メンバ」 256 ページ
- 「ValidateDatabase メソッド」 184 ページ
- 「ULDBValid 列挙体」 321 ページ

ULDatabaseSchema クラス

UL 拡張：Ultra Light.NET データベースのスキーマを表します。このクラスは継承できません。

構文

Visual Basic
Public NotInheritable Class **ULDatabaseSchema**

C#
public sealed class **ULDatabaseSchema**

備考

このクラスにはコンストラクタがありません。ULDatabaseSchema オブジェクトは、ULConnection.Schema として接続にアタッチされ、接続が開いている間のみ有効です。

参照

- 「ULDatabaseSchema メンバ」 265 ページ
- 「ULDatabaseSchema クラス」 265 ページ
- 「Schema プロパティ」 151 ページ

ULDatabaseSchema メンバ

パブリック・プロパティ

メンバ名	説明
「CollationName プロパティ」 266 ページ	このプロパティは廃止される予定です。代わりに GetDatabaseProperty("Collation") を使用してください。データベースの照合順の名前。
「IsCaseSensitive プロパティ」 267 ページ	データベースで大文字と小文字が区別されるかどうかをチェックします。
「IsOpen プロパティ」 267 ページ	データベース・スキーマが開いているかどうかを示します。
「PublicationCount プロパティ」 268 ページ	データベース内のパブリケーションの数です。
「TableCount プロパティ」 268 ページ	データベース内のテーブルの数です。

パブリック・メソッド

メンバ名	説明
「GetDatabaseProperty メソッド」 269 ページ	指定されたデータベースのプロパティの値を返します。
「GetPublicationName メソッド」 271 ページ	指定されたパブリケーション ID で識別されたパブリケーションの名前を返します。
「GetPublicationSchema メソッド」 272 ページ	指定されたパブリケーションに対応するパブリケーション・スキーマを返します。
「GetTableName メソッド」 273 ページ	指定されたテーブル ID で識別されたテーブルの名前を返します。
「SetDatabaseOption メソッド」 273 ページ	指定されたデータベース・オプションの値を設定します。

参照

- 「ULDatabaseSchema クラス」 265 ページ
- 「ULDatabaseSchema クラス」 265 ページ
- 「Schema プロパティ」 151 ページ

CollationName プロパティ

このプロパティは廃止される予定です。代わりに GetDatabaseProperty("Collation") を使用してください。データベースの照合順の名前。

構文

Visual Basic
Public Readonly Property **CollationName** As String

C#
public string **CollationName** { get;}

プロパティ値

データベースの照合順を表す文字列。

備考

データベースの照合順は、テーブルのインデックスと結果セットのソート方法に影響します。

参照

- 「ULDatabaseSchema クラス」 265 ページ
- 「ULDatabaseSchema メンバ」 265 ページ
- 「GetDatabaseProperty メソッド」 269 ページ

IsCaseSensitive プロパティ

データベースで大文字と小文字が区別されるかどうかをチェックします。

構文

Visual Basic
Public Readonly Property **IsCaseSensitive** As Boolean

C#
public bool **IsCaseSensitive** { get;}

プロパティ値

データベースで大文字と小文字が区別される場合は true、区別されない場合は false。

備考

データベースで大文字と小文字が区別されるかどうかは、テーブルのインデックスと結果セットのソート方法に影響します。また、大文字と小文字の区別は、ULConnectionParms.UserID と ULConnectionParms.Password の検証方法に影響します。

参照

- 「ULDatabaseSchema クラス」 265 ページ
- 「ULDatabaseSchema メンバ」 265 ページ
- 「GetDatabaseProperty メソッド」 269 ページ
- 「UserID プロパティ」 199 ページ
- 「Password プロパティ」 199 ページ

IsOpen プロパティ

データベース・スキーマが開いているかどうかを示します。

構文

Visual Basic
Public Readonly Property **IsOpen** As Boolean

C#
public bool **IsOpen** { get;}

プロパティ値

データベースのスキーマが現在開いている場合は true、現在閉じている場合は false。

備考

ULDatabaseSchema オブジェクトが開いているのは、それがアタッチされている接続が開いている場合だけです。

参照

- 「ULDatabaseSchema クラス」 265 ページ
- 「ULDatabaseSchema メンバ」 265 ページ

PublicationCount プロパティ

データベース内のパブリケーションの数です。

構文

Visual Basic
Public Readonly Property **PublicationCount** As Integer

C#
public int **PublicationCount** { get;}

プロパティ値

データベース内のパブリケーション数。接続が開いていない場合は 0。

備考

パブリケーション ID の範囲は、1 ～ PublicationCount です。

注意：パブリケーションの ID とカウントは、スキーマのアップグレード中に変更されることがあります。パブリケーションを正しく識別するには、名前でアクセスするか、キャッシュされている ID とカウントをスキーマのアップグレード後にリフレッシュします。

参照

- 「ULDatabaseSchema クラス」 265 ページ
- 「ULDatabaseSchema メンバ」 265 ページ
- 「GetPublicationName メソッド」 271 ページ

TableCount プロパティ

データベース内のテーブルの数です。

構文

Visual Basic
Public Readonly Property **TableCount** As Integer

C#
public int **TableCount** { get;}

プロパティ値

データベース内のテーブル数。接続が開いていない場合は 0。

備考

テーブル ID の範囲は、1 ～ TableCount です。

注意：テーブルの ID とカウントは、スキーマのアップグレード中に変更されることがあります。テーブルを正しく識別するには、名前でアクセスするか、キャッシュされている ID とカウントをスキーマのアップグレード後にリフレッシュします。

参照

- 「ULDatabaseSchema クラス」 265 ページ
- 「ULDatabaseSchema メンバ」 265 ページ

GetDatabaseProperty メソッド

指定されたデータベースのプロパティの値を返します。

構文

Visual Basic

```
Public Function GetDatabaseProperty( _
   ByVal name As String _
) As String
```

C#

```
public string  GetDatabaseProperty(
   string  name
);
```

パラメータ

- **name**　値を取得するデータベース・プロパティの名前。プロパティ名では大文字と小文字が区別されません。

戻り値

文字列として返されるプロパティの値。

備考

識別されるプロパティは次のとおりです。

プロパティ	説明
CaseSensitive	大文字と小文字の区別のステータス。データベースで大文字と小文字が区別される場合は、ON を返します。それ以外の場合は、OFF を返します。 データベースで大文字と小文字が区別されるかどうかは、テーブルのインデックスと結果セットのソート方法に影響します。 大文字と小文字の区別は、接続の ULConnectionParms.UserID と ULConnectionParms.Password の検証方法に影響します。ユーザ ID は常に大文字と小文字が区別されません。パスワードは常に大文字と小文字が区別されます。
CharSet	データベースの文字セット。
ChecksumLevel	データベースで有効にするデータベース・ページのチェックサムのレベル。
Collation	データベースの照合順の名前。
CollationName	このプロパティは廃止される予定です。代わりに "Collation" を使用してください。
ConnCount	データベースとの接続の数。
date_format	データベースによる文字列変換に使用される日付フォーマット。 このフォーマットは、System.DateTime によって使用されるフォーマットと同じであるとは限りません。
date_order	データベースによる文字列変換に使用される日付順。
Encryption	データベースに適用されている暗号化のタイプ。None、Simple、AES、または AES_FIPS を返します。
File	データベースのファイル名。
global_database_id	グローバル・オートインクリメント・カラムに使用される global_database_id オプションの値。
isolation_level	あるトランザクションの操作が同時に実行される他のトランザクションの操作でも認識される程度を制御するのに使用される isolation_level オプションの値。 この値は、接続単位で設定します。
MaxHashSize	インデックスのハッシュに使用する、デフォルトの最大バイト数。このプロパティは、インデックス単位で指定できます。

プロパティ	説明
ml_remote_id	同期中にデータベースを識別するために使用する、ml_remote_id オプションの値。
Name	データベースの名前 (DBN)。
nearest_century	データベースによる文字列変換に使用される最も近い世紀。
PageSize	データベースのページ・サイズ (バイト)。
precision	データベースによる文字列変換に使用される浮動小数点の精度。
scale	データベースによる文字列変換中に、計算結果が最大精度にトランケートされるときの小数点以下の最大桁数。
time_format	データベースによる文字列変換に使用される時刻フォーマット。 このフォーマットは、System.TimeSpan によって使用されるフォーマットと同じであるとは限りません。
timestamp_format	データベースによる文字列変換に使用されるタイムスタンプのフォーマット。 このフォーマットは、System.DateTime によって使用されるフォーマットと同じであるとは限りません。
timestamp_increment	2 つのユニークなタイムスタンプ値間のマイクロ秒 (1,000,000 分の 1 秒) 単位の最小差。

参照

- 「ULDatabaseSchema クラス」 265 ページ
- 「ULDatabaseSchema メンバ」 265 ページ
- 「SetDatabaseOption メソッド」 273 ページ
- 「CollationName プロパティ」 266 ページ
- 「UserID プロパティ」 199 ページ
- 「Password プロパティ」 199 ページ
- TimeSpan
- DateTime

GetPublicationName メソッド

指定されたパブリケーション ID で識別されたパブリケーションの名前を返します。

構文

Visual Basic
```
Public Function GetPublicationName( _
  ByVal publD As Integer _
) As String
```

C#
```
public string  GetPublicationName(
  int publD
);
```

パラメータ

- **pubID**　パブリケーションの ID。値は、[1,PublicationCount] の範囲内であることが必要です。

戻り値

文字列として返されるパブリケーション名。

備考

注意：パブリケーションの ID とカウントは、スキーマのアップグレード中に変更されることがあります。パブリケーションを正しく識別するには、名前でアクセスするか、キャッシュされている ID とカウントをスキーマのアップグレード後にリフレッシュします。

参照

- 「ULDatabaseSchema クラス」 265 ページ
- 「ULDatabaseSchema メンバ」 265 ページ
- 「PublicationCount プロパティ」 268 ページ
- 「PublicationCount プロパティ」 268 ページ

GetPublicationSchema メソッド

指定されたパブリケーションに対応するパブリケーション・スキーマを返します。

構文

Visual Basic
```
Public Function GetPublicationSchema( _
  ByVal name As String _
) As ULPublicationSchema
```

C#
```
public ULPublicationSchema GetPublicationSchema(
  string  name
);
```

パラメータ

- **name**　パブリケーションの名前。

戻り値

指定されたパブリケーションを表す ULPublicationSchema オブジェクト。

参照

- 「ULDatabaseSchema クラス」 265 ページ
- 「ULDatabaseSchema メンバ」 265 ページ
- 「GetPublicationName メソッド」 271 ページ
- 「ULPublicationSchema クラス」 412 ページ

GetTableName メソッド

指定されたテーブル ID で識別されたテーブルの名前を返します。

構文

Visual Basic

```
Public Function GetTableName( _
  ByVal tableID As Integer _
) As String
```

C#

```
public string  GetTableName(
  int tableID
);
```

パラメータ

- **tableID**　テーブルの ID。値は、[1,TableCount] の範囲内であることが必要です。

戻り値

文字列として返されるテーブル名。

備考

テーブル ID は、スキーマのアップグレード中に変更されることがあります。テーブルを正しく識別するには、名前でアクセスするか、キャッシュされている ID をスキーマのアップグレード後にリフレッシュします。

参照

- 「ULDatabaseSchema クラス」 265 ページ
- 「ULDatabaseSchema メンバ」 265 ページ
- 「TableCount プロパティ」 268 ページ

SetDatabaseOption メソッド

指定されたデータベース・オプションの値を設定します。

構文

Visual Basic

```
Public Sub SetDatabaseOption( _
  ByVal name As String, _
  ByVal value As String _
)
```

C#

```
public void SetDatabaseOption(
  string  name,
  string  value
);
```

パラメータ

- **name** データベース・オプションの名前。オプション名では大文字と小文字が区別されません。
- **value** オプションの新しい値。

備考

データベース・オプションを指定すると、コミットが実行されます。

識別されるオプションは次のとおりです。

オプション	説明
global_database_id	グローバル・オートインクリメント・カラムに使用する値。値は、[0,System.UInt32.MaxValue] の範囲内であることが必要です。デフォルトは ULConnection.INVALID_DATABASE_ID (現在のデータベースにデータベース ID が設定されていないことを示すのに使用される) です。

オプション	説明
isolation_level	あるトランザクションの操作が、同時に実行される他のトランザクションの操作で認識される程度を制御するのに使用される値。値は "read_uncommitted" または "read_committed" のいずれかです。デフォルトは "read_committed" です。 接続の isolation_level を "read_uncommited" に設定することは、BeginTransaction(System.Data.IsolationLevel.ReadUncommitted) および Commit() 呼び出してその接続におけるすべての操作をラップすることと同じです。同様に、"read_committed" は System.Data.IsolationLevel.ReadCommitted と同じです。現在のトランザクションの独立性レベルを設定する場合は、SetDatabaseOption() ではなく、BeginTransaction(IsolationLevel) を使用してください。 Ultra Light での独立性レベルの定義は、ADO.NET のマニュアルの IsolationLevel の説明とは若干異なります。詳細については、「Ultra Light の独立性レベル」 『Ultra Light データベース管理とリファレンス』を参照してください。 この値は、接続単位で設定します。
ml_remote_id	同期中にデータベースを識別するために使用する値。値として NULL 参照 (Visual Basic の Nothing) を使用して、データベースから ml_remote_id オプションを削除します。

参照

- 「ULDatabaseSchema クラス」 265 ページ
- 「ULDatabaseSchema メンバ」 265 ページ
- 「GetDatabaseProperty メソッド」 269 ページ
- UInt32.MaxValue
- 「INVALID_DATABASE_ID フィールド」 145 ページ
- 「BeginTransaction(IsolationLevel) メソッド」 154 ページ

ULDataReader クラス

Ultra Light データベースの読み込み専用の双方向カーソルを表します。カーソルとは、テーブルまたはクエリからの結果セットの一連のローです。

構文

Visual Basic
Public Class **ULDataReader**
  Inherits DbDataReader

C#
public class **ULDataReader**: DbDataReader

備考

ULDataReader にはコンストラクタがありません。ULDataReader オブジェクトを取得するには、次のように ULCommand を実行します。

```
' Visual Basic
Dim cmd As ULCommand = new ULCommand( _
    "SELECT emp_id FROM employee FOR READ ONLY", conn _
  )
Dim reader As ULDataReader = cmd.ExecuteReader()

// C#
ULCommand cmd = new ULCommand(
    "SELECT emp_id FROM employee FOR READ ONLY", conn
  );
ULDataReader reader = cmd.ExecuteReader();
```

UL 拡張：ADO.NET 標準 では、結果セットで前方への移動だけが必要ですが、ULDataReader は双方向です。ULDataReader の Move メソッドによって、結果セットを移動するときに最大限の柔軟性が得られます。

ULDataReader は、読み込み専用の結果セットです。より柔軟なオブジェクトで結果を操作する必要がある場合は、ULCommand.ExecuteResultSet()、ULCommand.ExecuteTable()、または ULDataAdapter を使用します。ULDataReader は必要に応じてローを取得しますが、ULDataAdapter の場合、結果セットのすべてのローを取得しないと、オブジェクトに対してアクションを実行できません。結果セットのサイズが大きい場合、この違いのために ULDataReader の方が応答時間が速くなります。

UL 拡張：ULDataReader のすべてのカラムは、GetString を使用して取得できます。

継承：System.Data.Common.DbDataReader

実装：System.Data.IDataReader、System.Data.IDataRecord、System.IDisposable、System.ComponentModel.IListSource

参照

- 「ULDataReader メンバ」 277 ページ
- 「ULCommand クラス」 91 ページ
- 「ExecuteResultSet() メソッド」 122 ページ
- 「ExecuteTable() メソッド」 125 ページ
- 「ULDataAdapter クラス」 242 ページ
- 「GetString メソッド」 305 ページ
- DbDataReader
- IDataReader
- IDataRecord
- IDisposable
- IListSource

ULDataReader メンバ

パブリック・プロパティ

メンバ名	説明
「Depth プロパティ」 281 ページ	現在のローのネストの深さを返します。最も外側のテーブルの深さは 0 です。
「FieldCount プロパティ」 281 ページ	このカーソル内のカラム数を返します。
「HasRows プロパティ」 282 ページ	ULDataReader に 1 つまたは複数のローがあるかどうかをチェックします。
「IsBOF プロパティ」 282 ページ	**UL 拡張：**現在のローの位置が最初のローの前かどうかをチェックします。
「IsClosed プロパティ」 283 ページ	カーソルが現在開いているかどうかを確認します。
「IsEOF プロパティ」 283 ページ	**UL 拡張：**現在のローの位置が最後のローの後かどうかをチェックします。
「Item プロパティ」 283 ページ	指定されたカラムの値を Object のインスタンスとして取得します。
「RecordsAffected プロパティ」 285 ページ	SQL 文の実行によって変更、挿入、または削除されたローの数を返します。SELECT 文または CommandType.TableDirect テーブルの場合、この値は -1 です。
「RowCount プロパティ」 286 ページ	**UL 拡張：**カーソル内のローの数を返します。
「Schema プロパティ」 286 ページ	**UL 拡張：**このカーソルのスキーマを保持します。

メンバ名	説明
VisibleFieldCount (DbDataReader から継承)	DbDataReader の非表示でないフィールドの数を取得します。

パブリック・メソッド

メンバ名	説明
「Close メソッド」 287 ページ	カーソルを閉じます。
Dispose (DbDataReader から継承)	この DbDataReader によって消費されたリソースを解放します。
「GetBoolean メソッド」 287 ページ	指定されたカラムの値を System.Boolean として返します。
「GetByte メソッド」 288 ページ	指定されたカラムの値を符号なし 8 ビット値 (System.Byte) として返します。
「GetBytes メソッド」 289 ページ	**UL 拡張：**指定されたカラムの値を System.Bytes 配列として返します。ULDbType.Binary 型、ULDbType.LongBinary 型、ULDbType.UniqueIdentifier 型のカラムの場合にのみ有効です。
「GetChar メソッド」 291 ページ	このメソッドは Ultra Light.NET ではサポートされていません。
「GetChars メソッド」 292 ページ	指定されたオフセットで始まる、指定された ULDbType.LongVarchar カラムの値のサブセットを、コピー先の System.Char 配列の指定されたオフセットにコピーします。
GetData (DbDataReader から継承)	要求されたカラムの順序の DbDataReader オブジェクトを返します。
「GetDataTypeName メソッド」 293 ページ	指定されたカラムのプロバイダのデータ型の名前を返します。
「GetDateTime メソッド」 294 ページ	指定されたカラムの値を、ミリ秒の精度の System.DateTime として返します。
「GetDecimal メソッド」 294 ページ	指定されたカラムの値を System.Decimal として返します。
「GetDouble メソッド」 295 ページ	指定されたカラムの値を System.Double として返します。

メンバ名	説明
「GetEnumerator メソッド」 296 ページ	ULDataReader の反復処理を実行する System.Collections.IEnumerator を返します。
「GetFieldType メソッド」 296 ページ	指定されたカラムに最適な System.Type を返します。
「GetFloat メソッド」 297 ページ	指定されたカラムの値を System.Single として返します。
「GetGuid メソッド」 298 ページ	指定されたカラムの値を UUID (System.Guid) として返します。
「GetInt16 メソッド」 298 ページ	指定されたカラムの値を System.Int16 として返します。
「GetInt32 メソッド」 299 ページ	指定されたカラムの値を Int32 として返します。
「GetInt64 メソッド」 300 ページ	指定されたカラムの値を Int64 として返します。
「GetName メソッド」 300 ページ	指定されたカラムの名前を返します。
「GetOrdinal メソッド」 301 ページ	指定されたカラムのカラム ID を返します。
GetProviderSpecificFieldType (DbDataReader から継承)	指定されたカラムのプロバイダ固有のフィールド・タイプを返します。
GetProviderSpecificValue (DbDataReader から継承)	指定されたカラムの値を Object のインスタンスとして取得します。
GetProviderSpecificValues (DbDataReader から継承)	現在のローのコレクション内のプロバイダ固有のすべての属性カラムを取得します。
「GetSchemaTable メソッド」 303 ページ	ULDataReader のカラムのメタデータが記述された System.Data.DataTable を返します。
「GetString メソッド」 305 ページ	指定されたカラムの値を System.String として返します。
「GetTimeSpan メソッド」 306 ページ	指定されたカラムの値を、ミリ秒の精度の System.TimeSpan として返します。

メンバ名	説明
「GetUInt16 メソッド」 306 ページ	指定されたカラムの値を System.UInt16 として返します。
「GetUInt32 メソッド」 307 ページ	指定されたカラムの値を UInt32 として返します。
「GetUInt64 メソッド」 308 ページ	指定されたカラムの値を System.UInt64 として返します。
「GetValue メソッド」 308 ページ	指定されたカラムの値をネイティブ・フォーマットで返します。
「GetValues メソッド」 309 ページ	現在のローのすべてのカラム値を返します。
「IsDBNull メソッド」 310 ページ	指定されたカラムの値が NULL かどうかをチェックします。
「MoveAfterLast メソッド」 311 ページ	**UL 拡張：**カーソルの最後のローの後に、カーソルを配置します。
「MoveBeforeFirst メソッド」 311 ページ	**UL 拡張：**カーソルの最初のローの前に、カーソルを配置します。
「MoveFirst メソッド」 311 ページ	**UL 拡張：**カーソルの最初のローに、カーソルを配置します。
「MoveLast メソッド」 312 ページ	**UL 拡張：**カーソルの最後のローに、カーソルを配置します。
「MoveNext メソッド」 312 ページ	**UL 拡張：**カーソルを次のローに配置します。ただし、カーソルがすでに最後のローにある場合は最後のローの後に配置します。
「MovePrevious メソッド」 313 ページ	**UL 拡張：**カーソルを前のローに配置するか、最初のローの前に配置します。
「MoveRelative メソッド」 313 ページ	**UL 拡張：**現在のローを基準としてカーソルを配置します。
「NextResult メソッド」 314 ページ	バッチ SQL 文の結果を読み込むときに ULDataReader を次の結果に進めます。
「Read メソッド」 314 ページ	カーソルを次のローに配置します。ただし、カーソルがすでに最後のローにある場合は最後のローの後に配置します。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULCommand クラス」 91 ページ
- 「ExecuteResultSet() メソッド」 122 ページ
- 「ExecuteTable() メソッド」 125 ページ
- 「ULDataAdapter クラス」 242 ページ
- 「GetString メソッド」 305 ページ
- DbDataReader
- IDataReader
- IDataRecord
- IDisposable

Depth プロパティ

現在のローのネストの深さを返します。最も外側のテーブルの深さは 0 です。

構文

Visual Basic
Public Overrides Readonly Property **Depth** As Integer

C#
public override int **Depth** { get;}

プロパティ値

Ultra Light.NET のすべての結果セットの深さは 0 です。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ

FieldCount プロパティ

このカーソル内のカラム数を返します。

構文

Visual Basic
Public Overrides Readonly Property **FieldCount** As Integer

C#
public override int **FieldCount** { get;}

プロパティ値

整数としてのカーソル内のカラム数。カーソルが閉じている場合は 0 を返します。

備考

このメソッドは、ULCursorSchema.ColumnCount メソッドと同じです。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ
- 「ColumnCount プロパティ」 234 ページ

HasRows プロパティ

ULDataReader に 1 つまたは複数のローがあるかどうかをチェックします。

構文

Visual Basic
Public Overrides Readonly Property **HasRows** As Boolean

C#
public override bool **HasRows** { get;}

プロパティ値

結果セットに少なくとも 1 つのローがある場合は True、ローがない場合は false。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ

IsBOF プロパティ

UL 拡張：現在のローの位置が最初のローの前かどうかをチェックします。

構文

Visual Basic
Public Readonly Property **IsBOF** As Boolean

C#
public bool **IsBOF** { get;}

プロパティ値

現在のローの位置が最初のローの前にくる場合は true、それ以外の場合は false。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ

IsClosed プロパティ

カーソルが現在開いているかどうかを確認します。

構文

Visual Basic
Public Overrides Readonly Property **IsClosed** As Boolean

C#
public override bool **IsClosed** { get;}

プロパティ値

カーソルが現在開いている場合は true、閉じている場合は false。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ

IsEOF プロパティ

UL 拡張：現在のローの位置が最後のローの後かどうかをチェックします。

構文

Visual Basic
Public Readonly Property **IsEOF** As Boolean

C#
public bool **IsEOF** { get;}

プロパティ値

現在のローの位置が最後のローの後にくる場合は true、それ以外の場合は false。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ

Item プロパティ

指定されたカラムの値を Object のインスタンスとして取得します。

Item(Int32) プロパティ

指定されたカラムの値をネイティブ・フォーマットで返します。C# では、このプロパティは ULDataReader クラスのインデクサです。

構文

Visual Basic
```
Public Overrides Default Readonly Property Item( _
  ByVal columnID As Integer _
) As Object
```

C#
```
public override object this[
  int columnID
] { get;}
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。

プロパティ値

そのカラムに最適な .NET 型としてのカラム値。カラムが NULL の場合は DBNull。

備考

このメソッドは、機能的には ULDataReader.GetValue(int) メソッドと同じです。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ
- 「Item プロパティ」 283 ページ
- 「GetFieldType メソッド」 296 ページ
- 「Item(String) プロパティ」 284 ページ
- 「GetValue メソッド」 308 ページ
- 「FieldCount プロパティ」 281 ページ

Item(String) プロパティ

指定された名前のカラムの値をネイティブ・フォーマットで返します。C# では、このプロパティは ULDataReader クラスのインデクサです。

構文

Visual Basic
```
Public Overrides Default Readonly Property Item( _
  ByVal name As String _
) As Object
```

C#
```
public override object this[
  string  name
] { get;}
```

パラメータ

- **name**　カラムの名前。

プロパティ値

そのカラムに最適な .NET 型としてのカラム値。カラムが NULL の場合は DBNull。

備考

結果セットでは、すべてのカラムに名前があるとは限らないほか、すべてのカラム名がユニークであるとは限りません。エイリアスを使用していない場合は、非計算カラムの名前には、そのカラムの元となるテーブルの名前がプレフィクスとして付けられます。たとえば、MyTable.ID は、クエリ "SELECT ID FROM MyTable" の結果セットに含まれる唯一のカラムの名前です。

カラムに何回もアクセスするときは、名前ではなく、カラム ID でアクセスすると効率が良くなります。

このメソッドは次と同じです。

```
dataReader.GetValue( dataReader.GetOrdinal( name ) )
```

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ
- 「Item プロパティ」 283 ページ
- 「Item(Int32) プロパティ」 283 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「GetValue メソッド」 308 ページ
- 「GetFieldType メソッド」 296 ページ

RecordsAffected プロパティ

SQL 文の実行によって変更、挿入、または削除されたローの数を返します。SELECT 文または CommandType.TableDirect テーブルの場合、この値は -1 です。

構文

Visual Basic

```
Public Overrides Readonly Property RecordsAffected As Integer
```

C#

```
public override int RecordsAffected { get;}
```

プロパティ値

SQL 文の実行によって変更、挿入、または削除されたローの数。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ
- CommandType.TableDirect

RowCount プロパティ

UL 拡張：カーソル内のローの数を返します。

構文

Visual Basic
Public Readonly Property **RowCount** As Integer

C#
public int **RowCount** { get;}

プロパティ値

カーソル内のロー数。

備考

RowCount は、古いローを削除して領域を節約するタイミングを決定するときに使用します。ULConnection.StopSynchronizationDelete メソッドを使用すると、統合データベースからは削除しないで Ultra Light データベースから古いローを削除できます。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ
- 「StartSynchronizationDelete メソッド」 181 ページ
- 「StopSynchronizationDelete メソッド」 182 ページ

Schema プロパティ

UL 拡張：このカーソルのスキーマを保持します。

構文

Visual Basic
Public Readonly Property **Schema** As ULCursorSchema

C#
public ULCursorSchema **Schema** { get;}

プロパティ値

結果セットの場合、結果セットのスキーマを表す ULResultSetSchema オブジェクト。テーブルの場合、テーブルのスキーマを表す ULTableSchema オブジェクト。

備考

このプロパティは、ULDataReader.GetSchemaTable からの結果に示されない Ultra Light.NET の詳細情報を含め、カーソルの完全なスキーマを表します。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ
- 「ULTableSchema クラス」 571 ページ
- 「GetSchemaTable メソッド」 303 ページ
- 「ULResultSetSchema クラス」 442 ページ

Close メソッド

カーソルを閉じます。

構文

Visual Basic
Public Overrides Sub **Close()**

C#
public override void **Close();**

備考

すでに閉じられているカーソルを閉じるのはエラーではありません。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ

GetBoolean メソッド

指定されたカラムの値を System.Boolean として返します。

構文

Visual Basic
```
Public Overrides Function GetBoolean( _
  ByVal columnID As Integer _
) As Boolean
```

C#
```
public override bool GetBoolean(
  int columnID
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。

戻り値

System.Boolean として返されるカラム値。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「GetFieldType メソッド」 296 ページ
- Boolean
- 「FieldCount プロパティ」 281 ページ

GetByte メソッド

指定されたカラムの値を符号なし 8 ビット値 (System.Byte) として返します。

構文

Visual Basic

```
Public Overrides Function GetByte( _
  ByVal columnID As Integer _
) As Byte
```

C#

```
public override byte GetByte(
  int columnID
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。

戻り値

System.Byte として返されるカラム値。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「GetFieldType メソッド」 296 ページ
- Byte
- 「FieldCount プロパティ」 281 ページ

GetBytes メソッド

UL 拡張：指定されたカラムの値を System.Bytes 配列として返します。ULDbType.Binary 型、ULDbType.LongBinary 型、ULDbType.UniqueIdentifier 型のカラムの場合にのみ有効です。

GetBytes(Int32, Int64, Byte[], Int32, Int32) メソッド

指定されたオフセットで始まる、指定された ULDbType.LongBinary カラムの値のサブセットを、コピー先の System.Byte 配列の指定されたオフセットにコピーします。

構文

Visual Basic

```
Public Overrides Function GetBytes( _
  ByVal columnID As Integer, _
  ByVal srcOffset As Long, _
  ByVal dst As Byte(), _
  ByVal dstOffset As Integer, _
  ByVal count As Integer _
) As Long
```

C#

```
public override long GetBytes(
  int columnID,
  long srcOffset,
  byte[] dst,
  int dstOffset,
  int count
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。
- **srcOffset**　カラム値の開始位置。最初の値は 0 です。
- **dst**　コピー先の配列。
- **dstOffset**　コピー先の配列の開始位置。
- **count**　コピーされるバイト数。

戻り値

実際にコピーされたバイト数。

備考

NULL 参照 (Visual Basic の Nothing) である *dst* バッファを渡すと、GetBytes はフィールドの長さをバイト数で返します。

値の *srcOffset* から *srcOffset*+*count*-1 までの位置のバイトが、コピー先の配列の *dstOffset* から *dstOffset*+*count*-1 までの位置に、それぞれコピーされます。*count* のバイト数がコピーされる前に、値の末尾が検出された場合は、コピー先の配列の残りは変更されないままになります。

次のいずれかに該当する場合、コード ULSQLCode.SQLE_INVALID_PARAMETER とともに ULException がスローされ、追加先は修正されません。

- *srcOffset* が負である。
- *dstOffset* が負である。
- *count* が負である。
- *dstOffset*+*count* が *dst*.Length. より長い。

その他のエラーの場合は、それに応じたエラー・コードとともに ULException がスローされます。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ
- 「GetBytes メソッド」 289 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「GetFieldType メソッド」 296 ページ
- 「GetBytes(Int32) メソッド」 290 ページ
- 「ULDbType 列挙体」 317 ページ
- Byte
- 「ULException クラス」 322 ページ
- 「ULSQLCode 列挙体」 462 ページ
- 「FieldCount プロパティ」 281 ページ

GetBytes(Int32) メソッド

UL 拡張：指定されたカラムの値を System.Bytes 配列として返します。ULDbType.Binary 型、ULDbType.LongBinary 型、ULDbType.UniqueIdentifier 型のカラムの場合にのみ有効です。

構文

Visual Basic

```
Public Function GetBytes( _
  ByVal columnID As Integer _
) As Byte()
```

C#

```
public byte[] GetBytes(
  int columnID
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。

戻り値

System.Bytes 配列として返されるカラム値。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ
- 「GetBytes メソッド」 289 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「GetFieldType メソッド」 296 ページ
- 「GetBytes(Int32, Int64, Byte[], Int32, Int32) メソッド」 289 ページ
- Byte
- 「ULDbType 列挙体」 317 ページ
- 「ULDbType 列挙体」 317 ページ
- 「ULDbType 列挙体」 317 ページ

GetChar メソッド

このメソッドは Ultra Light.NET ではサポートされていません。

構文

Visual Basic

```
Public Overrides Function GetChar( _
   ByVal columnID As Integer _
) As Char
```

C#

```
public override char GetChar(
   int columnID
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。

戻り値

このメソッドは Ultra Light.NET ではサポートされていません。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「GetFieldType メソッド」 296 ページ
- 「GetString メソッド」 305 ページ
- 「FieldCount プロパティ」 281 ページ

GetChars メソッド

指定されたオフセットで始まる、指定された ULDbType.LongVarchar カラムの値のサブセットを、コピー先の System.Char 配列の指定されたオフセットにコピーします。

構文

Visual Basic

```
Public Overrides Function GetChars( _
   ByVal columnID As Integer, _
   ByVal srcOffset As Long, _
   ByVal dst As Char(), _
   ByVal dstOffset As Integer, _
   ByVal count As Integer _
) As Long
```

C#

```
public override long GetChars(
   int columnID,
   long srcOffset,
   char[] dst,
   int dstOffset,
   int count
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。
- **srcOffset**　カラム値の開始位置。最初の値は 0 です。
- **dst**　コピー先の配列。
- **dstOffset**　コピー先の配列の開始位置。
- **count**　コピーされる文字数。

戻り値

実際にコピーされた文字数。

備考

NULL 参照 (Visual Basic の Nothing) である *dst* バッファを渡すと、GetChars はフィールドの長さを文字数として返します。

値の *srcOffset* から *srcOffset*+*count*-1 までの位置の文字が、コピー先の配列の *dstOffset* から *dstOffset* +*count*-1 までの位置に、それぞれコピーされます。*count* の文字数がコピーされる前に、値の末尾が検出された場合は、コピー先の配列の残りは変更されないままになります。

次のいずれかに該当する場合、コード ULSQLCode.SQLE_INVALID_PARAMETER とともに ULException がスローされ、追加先は修正されません。

- *srcOffset* が負である。
- *dstOffset* が負である。
- *count* が負である。
- *dstOffset*+*count* が *dst*.Length. より長い。

その他のエラーの場合は、それに応じたエラー・コードとともに ULException がスローされます。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「GetFieldType メソッド」 296 ページ
- 「ULDbType 列挙体」 317 ページ
- Char
- 「ULException クラス」 322 ページ
- 「ULSQLCode 列挙体」 462 ページ
- 「FieldCount プロパティ」 281 ページ

GetDataTypeName メソッド

指定されたカラムのプロバイダのデータ型の名前を返します。

構文

Visual Basic

```
Public Overrides Function GetDataTypeName( _
  ByVal columnID As Integer _
) As String
```

C#

```
public override string  GetDataTypeName(
  int columnID
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。

戻り値

カラムの ULDbType に対応する文字列。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「GetColumnULDbType メソッド」 240 ページ
- 「FieldCount プロパティ」 281 ページ
- 「ULDbType 列挙体」 317 ページ

GetDateTime メソッド

指定されたカラムの値を、ミリ秒の精度の System.DateTime として返します。

構文

Visual Basic

```
Public Overrides Function GetDateTime( _
  ByVal columnID As Integer _
) As Date
```

C#

```
public override DateTime GetDateTime(
  int columnID
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。

戻り値

System.DateTime として返されるカラム値。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「GetFieldType メソッド」 296 ページ
- DateTime
- 「FieldCount プロパティ」 281 ページ

GetDecimal メソッド

指定されたカラムの値を System.Decimal として返します。

構文

Visual Basic

```
Public Overrides Function GetDecimal( _
```

```
  ByVal columnID As Integer _
) As Decimal
```

C#

```
public override decimal GetDecimal(
  int columnID
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。

戻り値

System.Decimal として返されるカラム値。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「GetFieldType メソッド」 296 ページ
- Decimal
- 「FieldCount プロパティ」 281 ページ

GetDouble メソッド

指定されたカラムの値を System.Double として返します。

構文

Visual Basic

```
Public Overrides Function GetDouble( _
  ByVal columnID As Integer _
) As Double
```

C#

```
public override double GetDouble(
  int columnID
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。

戻り値

System.Double として返されるカラム値。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「GetFieldType メソッド」 296 ページ
- Double
- 「FieldCount プロパティ」 281 ページ

GetEnumerator メソッド

ULDataReader の反復処理を実行する System.Collections.IEnumerator を返します。

構文

Visual Basic
Public Overrides Function **GetEnumerator()** As IEnumerator

C#
public override IEnumerator **GetEnumerator();**

戻り値

ULDataReader の System.Collections.IEnumerator。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ
- IEnumerator

GetFieldType メソッド

指定されたカラムに最適な System.Type を返します。

構文

Visual Basic
```
Public Overrides Function GetFieldType( _
  ByVal columnID As Integer _
) As Type
```

C#
```
public override Type GetFieldType(
  int columnID
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。

戻り値

カラムの System.Type 値。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「GetDataTypeName メソッド」 293 ページ
- 「GetColumnULDbType メソッド」 240 ページ
- Type
- 「FieldCount プロパティ」 281 ページ

GetFloat メソッド

指定されたカラムの値を System.Single として返します。

構文

Visual Basic

```
Public Overrides Function GetFloat( _
  ByVal columnID As Integer _
) As Single
```

C#

```
public override float  GetFloat(
  int columnID
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。

戻り値

System.Single として返されるカラム値。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「GetFieldType メソッド」 296 ページ
- Single
- 「FieldCount プロパティ」 281 ページ

GetGuid メソッド

指定されたカラムの値を UUID (System.Guid) として返します。

構文

Visual Basic
```
Public Overrides Function GetGuid( _
  ByVal columnID As Integer _
) As Guid
```

C#
```
public override Guid GetGuid(
  int columnID
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。

戻り値

Guid として返されるカラム値。

備考

このメソッドが有効なのは、ULDbType.UniqueIdentifier 型のカラム、または長さが 16 の ULDbType.Binary 型のカラムの場合だけです。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「GetFieldType メソッド」 296 ページ
- 「GetColumnULDbType メソッド」 240 ページ
- 「GetColumnSize メソッド」 239 ページ
- Guid
- 「ULDbType 列挙体」 317 ページ
- 「ULDbType 列挙体」 317 ページ
- 「FieldCount プロパティ」 281 ページ

GetInt16 メソッド

指定されたカラムの値を System.Int16 として返します。

構文

Visual Basic
```
Public Overrides Function GetInt16( _
```

```
  ByVal columnID As Integer _
) As Short

C#
public override short GetInt16(
  int columnID
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。

戻り値

System.Int16 として返されるカラム値。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「GetFieldType メソッド」 296 ページ
- Int16
- 「FieldCount プロパティ」 281 ページ

GetInt32 メソッド

指定されたカラムの値を Int32 として返します。

構文

```
Visual Basic
Public Overrides Function GetInt32( _
  ByVal columnID As Integer _
) As Integer

C#
public override int GetInt32(
  int columnID
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。

戻り値

Int32 として返されるカラム値。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「GetFieldType メソッド」 296 ページ
- Int32
- 「FieldCount プロパティ」 281 ページ

GetInt64 メソッド

指定されたカラムの値を Int64 として返します。

構文

Visual Basic

```
Public Overrides Function GetInt64( _
  ByVal columnID As Integer _
) As Long
```

C#

```
public override long GetInt64(
  int columnID
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。

戻り値

Int64 として返されるカラム値。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「GetFieldType メソッド」 296 ページ
- Int64
- 「FieldCount プロパティ」 281 ページ

GetName メソッド

指定されたカラムの名前を返します。

構文

Visual Basic

```
Public Overrides Function GetName( _
```



```
  ByVal columnID As Integer _
) As String
```

C#

```
public override string  GetName(
  int columnID
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。

戻り値

カラムの名前。カラムに名前がない場合は NULL 参照 (Visual Basic の Nothing)。SQL クエリでカラムのエイリアスが使用されている場合は、そのエイリアスが返されます。

備考

結果セットでは、すべてのカラムに名前があるとは限らないほか、すべてのカラム名がユニークであるとは限りません。エイリアスを使用していない場合は、非計算カラムの名前には、そのカラムの元となるテーブルの名前がプレフィクスとして付けられます。たとえば、MyTable.ID は、クエリ "SELECT ID FROM MyTable" の結果セットに含まれる唯一のカラムの名前です。

このメソッドは、ULCursorSchema.GetColumnName メソッドと同じです。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ
- 「FieldCount プロパティ」 281 ページ
- 「GetSchemaTable メソッド」 303 ページ
- 「GetColumnName メソッド」 236 ページ
- 「FieldCount プロパティ」 281 ページ

GetOrdinal メソッド

指定されたカラムのカラム ID を返します。

構文

Visual Basic

```
Public Overrides Function GetOrdinal( _
  ByVal columnName As String _
) As Integer
```

C#

```
public override int GetOrdinal(
  string  columnName
);
```

パラメータ

- **columnName**　カラムの名前。

戻り値

指定されたカラムのカラム ID。

備考

カラム ID の範囲は、0 ～ ULDataReader.FieldCount-1 です。

結果セットでは、すべてのカラムに名前があるとは限らないほか、すべてのカラム名がユニークであるとは限りません。エイリアスを使用していない場合は、非計算カラムの名前には、そのカラムの元となるテーブルの名前がプレフィクスとして付けられます。たとえば、MyTable.ID は、クエリ "SELECT ID FROM MyTable" の結果セットに含まれる唯一のカラムの名前です。

カラムの ID とカウントは、スキーマのアップグレード中に変更されることがあります。カラムを正しく識別するには、名前でアクセスするか、キャッシュされている ID とカウントをスキーマのアップグレード後にリフレッシュします。

このメソッドは、ULCursorSchema.GetColumnID メソッドと同じです。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ
- 「GetSchemaTable メソッド」 303 ページ
- 「FieldCount プロパティ」 281 ページ
- 「GetColumnID メソッド」 235 ページ

GetRowCount メソッド

テーブルまたは結果セットのローの数を返します。特定の数より多いローが使用できるかどうかを判断するためだけに必要な場合は、スレッショルド値を指定して、すべてのローが列挙されるのを回避します。たとえば、10 個のローの情報を表示するアプリケーションの場合、さらに情報を表示するオプションをユーザに提示するために、使用できるローが 10 個以上あるかどうかの判断が必要になることがあります。

構文

Visual Basic

```
public int GetRowCount( _
ByVal threshold as Integer ) _
as Integer
```

C#

```
public int GetRowCount( int threshold );
```

戻り値

スレッショルド・パラメータに 0 (スレッショルドなし) を設定すると、テーブルまたは結果セット内のローの数が返されます。それ以外の場合は、スレッショルド値の最大数までのロー数が返されます。

備考

テーブルまたは結果セット内のすべてのローを列挙すると、サイズの大きいテーブルや結果セットの場合は、操作にかかるコストが高くなる可能性があります。特定の数より多いローが存在するかどうかだけを知る必要があるアプリケーションの場合は、このメソッドを使用します。

GetSchemaTable メソッド

ULDataReader のカラムのメタデータが記述された System.Data.DataTable を返します。

構文

Visual Basic
Public Overrides Function **GetSchemaTable()** As DataTable

C#
public override DataTable **GetSchemaTable();**

戻り値

ULDataReader の各カラムのスキーマが記述された System.Data.DataTable。

備考

GetSchemaTable メソッドは、各カラムに関するメタデータを次の順で返します。

DataTable カラム	説明
ColumnName	カラムの名前。カラムに名前がない場合は NULL 参照 (Visual Basic の Nothing)。SQL クエリでカラムのエイリアスが使用されている場合は、そのエイリアスが返されます。結果セットでは、すべてのカラムに名前があるとは限らないほか、すべてのカラム名がユニークであるとは限りません。
ColumnOrdinal	カラムの ID。値の範囲は、[0, FieldCount -1] です。
ColumnSize	サイズ指定されたカラムの場合、カラムの値の最大長。その他のカラムの場合、これは、そのデータ型のバイト単位のサイズです。
NumericPrecision	数値カラム (ProviderType ULDbType.Decimal または ULDbType.Numeric) の精度。カラムが数値型ではない場合は DBNull。

DataTable カラム	説明
NumericScale	数値カラム (ProviderType ULDbType.Decimal または ULDbType.Numeric) の位取り。カラムが数値型ではない場合は DBNull。
IsUnique	カラムが取得元のテーブル (BaseTableName) でユニークな非計算カラムである場合は true。
IsKey	カラムが、結果セットのユニーク・キーからともに取得された結果セットの一連のカラムのいずれかである場合は true。IsKey が true に設定されたカラムのセットは、結果セット内のローをユニークに識別する最小限のセットである必要はありません。
BaseCatalogName	カラムが含まれているデータベース内のカタログの名前。Ultra Light.NET の場合、この値は常に DBNull です。
BaseColumnName	データベースのテーブル BaseTableName にあるカラムの元の名前。カラムが計算される場合と、この情報を特定できない場合は、DBNull です。
BaseSchemaName	カラムが含まれているデータベース内のスキーマの名前。Ultra Light.NET の場合、この値は常に DBNull です。
BaseTableName	カラムが含まれているデータベースのテーブルの名前。カラムが計算される場合と、この情報を特定できない場合は、DBNull です。
DataType	この型のカラムに最適な .NET データ型。
AllowDBNull	カラムが NULL 入力可である場合は true、NULL 入力不可である場合またはこの情報を特定できない場合は false。
ProviderType	カラムの ULDbType。
IsIdentity	カラムが identity カラムである場合は true、identity カラムでない場合は false。Ultra Light.NET の場合、この値は常に false です。
IsAutoIncrement	カラムがオートインクリメント・カラムまたはグローバル・オートインクリメント・カラムである場合は true、それ以外のカラムである場合 (または、この情報を特定できない場合) は false。
IsRowVersion	書き込みできない永続的なロー識別子がカラムに含まれており、ローを識別する以外は意味を持たない値がカラムにある場合は true。Ultra Light.NET の場合、この値は常に false です。

DataTable カラム	説明
IsLong	カラムが ULDbType.LongVarchar カラムまたは ULDbType.LongBinary カラムである場合は true、それ以外のカラムの場合は false。
IsReadOnly	カラムが読み込み専用である場合は true、カラムが修正可能であるか、カラムのアクセス権を特定できない場合は false。
IsAliased	カラム名がエイリアスの場合は true、エイリアスでない場合は false。
IsExpression	カラムが式の場合は true、カラム値の場合は false。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ
- 「Schema プロパティ」 286 ページ
- 「FieldCount プロパティ」 281 ページ
- 「ULDbType 列挙体」 317 ページ
- 「ULDbType 列挙体」 317 ページ
- 「ULDbType 列挙体」 317 ページ
- 「ULDbType 列挙体」 317 ページ
- 「ULDbType 列挙体」 317 ページ
- DataTable

GetString メソッド

指定されたカラムの値を System.String として返します。

構文

Visual Basic

```
Public Overrides Function GetString( _
   ByVal columnID As Integer _
) As String
```

C#

```
public override string  GetString(
   int columnID
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。

戻り値

System.String として返されるカラム値。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「GetFieldType メソッド」 296 ページ
- String
- 「FieldCount プロパティ」 281 ページ

GetTimeSpan メソッド

指定されたカラムの値を、ミリ秒の精度の System.TimeSpan として返します。

構文

Visual Basic

```
Public Function GetTimeSpan( _
  ByVal columnID As Integer _
) As TimeSpan
```

C#

```
public TimeSpan GetTimeSpan(
  int columnID
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。

戻り値

System.TimeSpan として返されるカラム値。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「GetFieldType メソッド」 296 ページ
- TimeSpan
- 「FieldCount プロパティ」 281 ページ

GetUInt16 メソッド

指定されたカラムの値を System.UInt16 として返します。

構文

Visual Basic
```
Public Function GetUInt16( _
  ByVal columnID As Integer _
) As UInt16
```

C#
```
public ushort GetUInt16(
  int columnID
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。

戻り値

System.UInt16 として返されるカラム値。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「GetFieldType メソッド」 296 ページ
- UInt16
- 「FieldCount プロパティ」 281 ページ

GetUInt32 メソッド

指定されたカラムの値を UInt32 として返します。

構文

Visual Basic
```
Public Function GetUInt32( _
  ByVal columnID As Integer _
) As UInt32
```

C#
```
public uint GetUInt32(
  int columnID
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。

戻り値

UInt32 として返されるカラム値。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「GetFieldType メソッド」 296 ページ
- UInt32
- 「FieldCount プロパティ」 281 ページ

GetUInt64 メソッド

指定されたカラムの値を System.UInt64 として返します。

構文

Visual Basic

```
Public Function GetUInt64( _
  ByVal columnID As Integer _
) As UInt64
```

C#

```
public ulong GetUInt64(
  int columnID
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。

戻り値

System.UInt64 として返されるカラム値。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「GetFieldType メソッド」 296 ページ
- UInt64
- 「FieldCount プロパティ」 281 ページ

GetValue メソッド

指定されたカラムの値をネイティブ・フォーマットで返します。

構文

Visual Basic

```
Public Overrides Function GetValue( _
```

```
  ByVal columnID As Integer _
) As Object
```

C#

```
public override object GetValue(
  int columnID
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。

戻り値

そのカラムに最適な .NET 型としてのカラム値。カラムが NULL の場合は DBNull。

備考

このメソッドは、機能的には ULDataReader.this[int] と同じです。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「GetFieldType メソッド」 296 ページ
- 「FieldCount プロパティ」 281 ページ
- 「Item(Int32) プロパティ」 283 ページ

GetValues メソッド

現在のローのすべてのカラム値を返します。

構文

Visual Basic

```
Public Overrides Function GetValues( _
  ByVal values As Object() _
) As Integer
```

C#

```
public override int GetValues(
  object[] values
);
```

パラメータ

- **values**　ロー全体を保持する System.Objects 配列。

戻り値

取り出されるカラム値の数。配列の長さがカラムの数 (ULDataReader.FieldCount) よりも大きい場合は、FieldCount の項目のみが取り出され、配列の残りは変更されないままになります。

備考

ほとんどのアプリケーションについて、GetValues メソッドは、各カラムを個々に取り出すのではなく、すべてのカラムを取り出す効率的な方法を提供します。

結果のローに含まれるカラムの数より少ないカラムが含まれる System.Object 配列を渡すことができます。System.Object 配列が保持するデータ量のみが配列にコピーされます。また、結果のローに含まれるカラムの数より長い System.Object 配列を渡すこともできます。

このメソッドは、NULL データベース・カラムに対して DBNull を返します。その他のカラムの場合は、カラムの値をネイティブ・フォーマットで返します。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ
- 「FieldCount プロパティ」 281 ページ
- 「GetFieldType メソッド」 296 ページ
- 「GetValue メソッド」 308 ページ
- オブジェクト

IsDBNull メソッド

指定されたカラムの値が NULL かどうかをチェックします。

構文

Visual Basic

```
Public Overrides Function IsDBNull( _
  ByVal columnID As Integer _
) As Boolean
```

C#

```
public override bool IsDBNull(
  int columnID
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。

戻り値

値が NULL の場合は true、NULL でない場合は false。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「GetFieldType メソッド」 296 ページ
- 「FieldCount プロパティ」 281 ページ

MoveAfterLast メソッド

UL 拡張：カーソルの最後のローの後に、カーソルを配置します。

構文

Visual Basic
Public Sub **MoveAfterLast()**

C#
public void **MoveAfterLast();**

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ

MoveBeforeFirst メソッド

UL 拡張：カーソルの最初のローの前に、カーソルを配置します。

構文

Visual Basic
Public Sub **MoveBeforeFirst()**

C#
public void **MoveBeforeFirst();**

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ

MoveFirst メソッド

UL 拡張：カーソルの最初のローに、カーソルを配置します。

構文

Visual Basic
Public Function **MoveFirst()** As Boolean

C#
public bool **MoveFirst();**

戻り値

成功した場合は true、失敗した場合は false。たとえば、ローがない場合、メソッドは失敗します。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ

MoveLast メソッド

UL 拡張：カーソルの最後のローに、カーソルを配置します。

構文

Visual Basic
Public Function **MoveLast()** As Boolean

C#
public bool **MoveLast();**

戻り値

成功した場合は true、失敗した場合は false。たとえば、ローがない場合、メソッドは失敗します。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ

MoveNext メソッド

UL 拡張：カーソルを次のローに配置します。ただし、カーソルがすでに最後のローにある場合は最後のローの後に配置します。

構文

Visual Basic
Public Function **MoveNext()** As Boolean

C#
public bool **MoveNext();**

戻り値

成功した場合は true、失敗した場合は false。たとえば、それ以上ローがない場合、メソッドは失敗します。

備考

このメソッドは、ULDataReader.Read メソッドと同じです。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ
- 「Read メソッド」 314 ページ

MovePrevious メソッド

UL 拡張：カーソルを前のローに配置するか、最初のローの前に配置します。

構文

Visual Basic
Public Function **MovePrevious()** As Boolean

C#
public bool **MovePrevious();**

戻り値

成功した場合は true、失敗した場合は false。たとえば、それ以上ローがない場合、メソッドは失敗します。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ

MoveRelative メソッド

UL 拡張：現在のローを基準としてカーソルを配置します。

構文

Visual Basic
```
Public Function MoveRelative( _
  ByVal offset As Integer _
) As Boolean
```

C#
```
public bool MoveRelative(
  int offset
);
```

パラメータ

- **offset**　移動するローの数。負の値を指定すると、後方に移動します。

戻り値

成功した場合は true、失敗した場合は false。たとえば、最初または最後のローを超えて移動する場合、メソッドは失敗します。

備考

指定された移動先にローがない場合には、false が返されます。その場合のカーソル位置は、*offset* が正であるときには最後のローの後 (ULDataReader.IsEOF) になり、*offset* が負であるときには最初のローの前 (ULDataReader.IsBOF) になります。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ
- 「IsEOF プロパティ」 283 ページ
- 「IsBOF プロパティ」 282 ページ

NextResult メソッド

バッチ SQL 文の結果を読み込むときに ULDataReader を次の結果に進めます。

構文

Visual Basic
Public Overrides Function **NextResult()** As Boolean

C#
public override bool **NextResult();**

戻り値

さらに結果セットがある場合は true、そうでない場合は false。Ultra Light.NET では、常に false が返されます。

備考

UL 拡張：Ultra Light.NET ではバッチ SQL 文はサポートされていません。ULDataReader は最初 (唯一) の結果セットに常に配置されます。NextResult は、呼び出しても機能しません。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ

Read メソッド

カーソルを次のローに配置します。ただし、カーソルがすでに最後のローにある場合は最後のローの後に配置します。

構文

Visual Basic
Public Overrides Function **Read()** As Boolean

C#
public override bool **Read();**

戻り値

成功した場合は true、失敗した場合は false。たとえば、それ以上ローがない場合、メソッドは失敗します。

備考

このメソッドは、ULDataReader.MoveNext メソッドと同じです。

参照

- 「ULDataReader クラス」 276 ページ
- 「ULDataReader メンバ」 277 ページ
- 「MoveNext メソッド」 312 ページ

ULDateOrder 列挙体

UL 拡張：データベースがサポートできる日付順を列挙します。

構文

Visual Basic
Public Enum **ULDateOrder**

C#
public enum **ULDateOrder**

メンバ

メンバ名	説明	値
DMY	日 (日、月、年の順に示す)	2
MDY	月 (月、日、年の順に示す)	1
YMD	年 (年、月、日の順に示す)	0

参照

- 「DateOrder プロパティ」 225 ページ

ULDbType 列挙体

Ultra Light.NET データベースのデータ型を列挙します。

構文

Visual Basic
Public Enum **ULDbType**

C#
public enum **ULDbType**

備考

下の表には、各 ULDbType との互換性がある .NET 型がリストされています。整数型の場合、テーブルのカラムは、常により小さい整数型を使用して設定できるほか、実際の値がその型の範囲内にあるかぎり、より大きい型を使用して設定することも可能です。

ULDbType	互換性のある .NET 型	C# 組み込みタイプ	Visual Basic 組み込みタイプ
Binary, **VarBinary**	System.Byte[]、または System.Guid (サイズが 16 の場合)	byte[]	Byte()
Bit	System.Boolean	bool	Boolean
Char, **VarChar**	System.String	String	String
Date	System.DateTime	DateTime 組み込みタイプなし	Date
Double	System.Double	double	Double
LongBinary	System.Byte[]	byte[]	Byte()
LongVarchar	System.String	String	String
Decimal, **Numeric**	System.String	decimal	Decimal
Float, **Real**	System.Single	float	Single
BigInt	System.Int64	long	Long
Integer	System.Int32	int	Integer
SmallInt	System.Int16	short	Short

ULDbType	互換性のある .NET 型	C# 組み込みタイプ	Visual Basic 組み込みタイプ
Time	System.TimeSpan	TimeSpan 組み込みタイプなし	TimeSpan 組み込みタイプなし
DateTime, **TimeStamp**	System.DateTime	DateTime 組み込みタイプなし	Date
TinyInt	System.Byte	byte	Byte
UnsignedBigInt	System.UInt64	ulong	UInt64 組み込みタイプなし
UnsignedInt	System.UInt32	uint	UInt32 組み込みタイプなし
UnsignedSmallInt	System.UInt16	ushort	UInt16 組み込みタイプなし
UniqueIdentifier	System.Guid	Guid 組み込みタイプなし	Guid 組み込みタイプなし

長さが 16 のバイナリ・カラムには、UniqueIdentifier 型との完全な互換性があります。

メンバ

メンバ名	説明	値
BigInt	符号付き 64 ビット整数値	5
Binary	指定された最大長のバイナリ・データ。列挙値 **Binary** と **VarBinary** は同等のエイリアスです。	15
Bit	1 ビット・フラグ	8
Char	指定された長さの文字データ。Ultra Light.NET では、この型は常に Unicode 文字をサポートします。**Char** 型と **VarChar** 型は、完全に互換性があります。	0
Date	日付情報	10

メンバ名	説明	値
DateTime	タイムスタンプ情報 (日付、時刻)。列挙値 **DateTime** と **TimeStamp** は同等のエイリアスです。	9
Decimal	精度と桁数が指定された正確な数値データ。列挙値 **Decimal** と **Numeric** は同等のエイリアスです。	14
Double	倍精度浮動小数点数 (8 バイト)	12
Float	単精度浮動小数点数 (4 バイト)。列挙値 **Float** と **Real** は同等のエイリアスです。	13
Integer	符号なし 32 ビット整数値	1
LongBinary	可変長のバイナリ・データ	18
LongVarchar	可変長の文字データ。Ultra Light.NET では、この型は常に Unicode 文字をサポートします。	17
Numeric	精度と桁数が指定された正確な数値データ。列挙値 **Decimal** と **Numeric** は同等のエイリアスです。	14
Real	単精度浮動小数点数 (4 バイト)。列挙値 **Float** と **Real** は同等のエイリアスです。	13
SmallInt	符号付き 16 ビット整数値	3
Time	時刻情報	11
TimeStamp	タイムスタンプ情報 (日付、時刻)。列挙値 **DateTime** と **TimeStamp** は同等のエイリアスです。	9
TinyInt	符号なし 8 ビット整数値	7
UniqueIdentifier	ユニバーサル・ユニーク識別子 (UUID/GUID)	19
UnsignedBigInt	符号なし 64 ビット整数値	6
UnsignedInt	符号なし 32 ビット整数値	2
UnsignedSmallInt	符号なし 16 ビット整数値	4

メンバ名	説明	値
VarBinary	指定された最大長のバイナリ・データ。列挙値 **Binary** と **VarBinary** は同等のエイリアスです。	15
VarChar	指定された最大長の文字データ。Ultra Light.NET では、この型は常に Unicode 文字をサポートします。**Char** 型と **VarChar** 型は、完全に互換性があります。	16

参照

- 「GetFieldType メソッド」 296 ページ
- 「GetDataTypeName メソッド」 293 ページ
- 「GetColumnULDbType メソッド」 240 ページ
- Byte
- Guid
- Boolean
- String
- DateTime
- Single
- Int64
- Int32
- Int16
- TimeSpan
- UInt64
- UInt32
- UInt16

ULDBValid 列挙体

Ultra Light.NET データベース検証メソッドを列挙します。

構文

Visual Basic
Public Enum **ULDBValid**

C#
public enum **ULDBValid**

備考

データベースの検証については、「ValidateDatabase メソッド」 263 ページを参照してください。

メンバ

メンバ名	値
EXPRESS_VALIDATE	0
FULL_VALIDATE	1

参照

- 「ValidateDatabase メソッド」 263 ページ
- 「ValidateDatabase メソッド」 184 ページ

ULException クラス

Ultra Light.NET データベースによって返される SQL エラーを表します。このクラスは継承できません。

構文

Visual Basic
Public NotInheritable Class **ULException**
  Inherits ApplicationException

C#
public sealed class **ULException**: ApplicationException

備考

NativeError にエラーが返されたことを示す SQLCODE。

.NET Compact Framework では、このクラスはシリアル化可能ではありません。

参照

- 「ULException メンバ」 322 ページ
- 「NativeError プロパティ」 324 ページ

ULException メンバ

パブリック・コンストラクタ

メンバ名	説明
「ULException コンストラクタ」 323 ページ	指定されたエラー・コードを持つ ULException を作成します。

パブリック・プロパティ

メンバ名	説明
Data (Exception から継承)	例外に関するユーザ定義の追加情報を提供するキー／値のペアのコレクションを取得します。
HelpLink (Exception から継承)	この例外に関連付けられたヘルプ・ファイルへのリンクを取得または設定します。
InnerException (Exception から継承)	現在の例外を発生させた Exception インスタンスを取得します。
Message (Exception から継承)	現在の例外を説明するメッセージを取得します。

メンバ名	説明
「NativeError プロパティ」 324 ページ	データベースによって返される SQL コードを返します。
「Source プロパティ」 324 ページ	エラーを生成したプロバイダの名前を返します。
StackTrace (Exception から継承)	現在の例外がスローされた時点のコール・スタックのフレームを表す文字列を取得します。
TargetSite (Exception から継承)	現在の例外をスローしたメソッドを取得します。

パブリック・メソッド

メンバ名	説明
GetBaseException (Exception から継承)	派生クラスで上書きされる場合は、後続の 1 つ以上の例外の根本原因となる Exception を返します。
「GetObjectData メソッド」 325 ページ	この ULException のシリアル化に必要なデータを SerializationInfo に移植します。
GetType (Exception から継承)	現在のインスタンスのランタイム・タイプを取得します。
ToString (Exception から継承)	現在の例外を表す文字列を作成して返します。

参照

- 「ULException クラス」 322 ページ
- 「NativeError プロパティ」 324 ページ

ULException コンストラクタ

指定されたエラー・コードを持つ ULException を作成します。

構文

Visual Basic

```
Public Sub New( _
  ByVal code As ULSQLCode, _
  ByVal s1 As String, _
  ByVal s2 As String, _
  ByVal s3 As String _
)
```

C#

```
public  ULException(
  ULSQLCode code,
  string  s1,
  string  s2,
```

```
  string s3
);
```

パラメータ

- **code**　例外のコード。
- **s1**　フォーマットされたメッセージの 1 番目の文字列。
- **s2**　フォーマットされたメッセージの 2 番目の文字列。
- **s3**　フォーマットされたメッセージの 3 番目の文字列。

備考

指定された iAnywhere.Data.UltraLite.ULSQLCode に対応するメッセージ文字列は、**iAnywhere.Data.UltraLite.resources** アセンブリから取得されます。リソースは、CultureInfo.CurrentUICulture、CultureInfo.CurrentCulture、国 "EN" の順序で国別に検索します。

参照

- 「ULException クラス」 322 ページ
- 「ULException メンバ」 322 ページ
- 「ULSQLCode 列挙体」 462 ページ
- CultureInfo.CurrentUICulture
- CultureInfo.CurrentCulture

NativeError プロパティ

データベースによって返される SQLCODE を返します。

構文

Visual Basic
Public Readonly Property **NativeError** As ULSQLCode

C#
public ULSQLCode **NativeError** { get;}

プロパティ値

データベースによって返される ULSQLCode 値。

参照

- 「ULException クラス」 322 ページ
- 「ULException メンバ」 322 ページ
- 「ULSQLCode 列挙体」 462 ページ

Source プロパティ

エラーを生成したプロバイダの名前を返します。

構文

Visual Basic
Public Overrides Readonly Property **Source** As String

C#
public override string **Source** { get;}

プロパティ値

プロバイダが Ultra Light.NET であることを示す文字列値。

参照

- 「ULException クラス」 322 ページ
- 「ULException メンバ」 322 ページ

GetObjectData メソッド

この ULException のシリアル化に必要なデータを SerializationInfo に移植します。

構文

Visual Basic
```
Public Overrides Sub GetObjectData( _
  ByVal info As SerializationInfo, _
  ByVal context As StreamingContext _
)
```

C#
```
public override void GetObjectData(
  SerializationInfo info,
  StreamingContext context
);
```

パラメータ

- **info**　データを移植する SerializationInfo。
- **context**　このシリアル化の宛先。

備考

.NET Compact Framework では、このメソッドはサポートされていません。

参照

- 「ULException クラス」 322 ページ
- 「ULException メンバ」 322 ページ

ULFactory クラス

データ・ソース・クラスの iAnywhere.Data.UltraLite プロバイダの実装のインスタンスを作成する、メソッドのセットを表します。これは静的なクラスであるため、継承またはインスタンス化はできません。

構文

Visual Basic
Public NotInheritable Class **ULFactory**
  Inherits DbProviderFactory

C#
public sealed class **ULFactory**: DbProviderFactory

備考

ADO.NET 2.0 には System.Data.Common.DbProviderFactories と System.Data.Common.DbProviderFactory という 2 つクラスが新しく追加され、プロバイダに依存しないコードを簡単に作成できるようになりました。これらを Ultra Light.NET で使用するには、GetFactory に渡されるプロバイダのインバリアント名として iAnywhere.Data.UltraLite を指定します。次に例を示します。

```
' Visual Basic
Dim factory As DbProviderFactory = _
  DbProviderFactories.GetFactory( "iAnywhere.Data.UltraLite" )
Dim conn As DbConnection = _
  factory.CreateConnection()

// C#
DbProviderFactory factory =
      DbProviderFactories.GetFactory( "iAnywhere.Data.UltraLite" );
DbConnection conn = factory.CreateConnection();
```

この例の中の conn は、ULConnection オブジェクトとして作成されます。

ADO.NET 2.0 におけるプロバイダ・ファクトリと汎用プログラミングについては、http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnvs05/html/vsgenerics.asp を参照してください。

Ultra Light.NET では、CreateCommandBuilder()、CreateDataSourceEnumerator()、CreatePermission() をサポートしていません。

制限：ULFactory クラスは、.NET Compact Framework 2.0 では使用できません。

継承：System.Data.Common.DbProviderFactory

参照

- 「ULFactory メンバ」 327 ページ
- DbProviderFactories
- DbProviderFactory

ULFactory メンバ

パブリック・フィールド

メンバ名	説明
「Instance フィールド」 328 ページ	ULFactory クラスのシングルトン・インスタンスを表します。このフィールドは読み込み専用です。

パブリック・プロパティ

メンバ名	説明
「CanCreateDataSourceEnumerator プロパティ」 328 ページ	Ultra Light.NET で DbDataSourceEnumerator がサポートされないことを示す false を返します。

パブリック・メソッド

メンバ名	説明
「CreateCommand メソッド」 329 ページ	厳密に型指定された System.Data.Common.DbCommand インスタンスを返します。
「CreateCommandBuilder メソッド」 329 ページ	厳密に型指定された System.Data.Common.DbCommandBuilder インスタンスを返します。
「CreateConnection メソッド」 330 ページ	厳密に型指定された System.Data.Common.DbConnection インスタンスを返します。
「CreateConnectionStringBuilder メソッド」 330 ページ	厳密に型指定された System.Data.Common.DbConnectionStringBuilder インスタンスを返します。
「CreateDataAdapter メソッド」 330 ページ	厳密に型指定された System.Data.Common.DbDataAdapter インスタンスを返します。
CreateDataSourceEnumerator (DbProviderFactory から継承)	DbDataSourceEnumerator を実装するプロバイダのクラスの新しいインスタンスを返します。
「CreateParameter メソッド」 331 ページ	厳密に型指定された System.Data.Common.DbParameter インスタンスを返します。
CreatePermission (DbProviderFactory から継承)	CodeAccessPermission のプロバイダ・バージョンを実装するプロバイダのクラスの新しいインスタンスを返します。

参照

- 「ULFactory クラス」 326 ページ
- DbProviderFactories
- DbProviderFactory

Instance フィールド

ULFactory クラスのシングルトン・インスタンスを表します。このフィールドは読み込み専用です。

構文

Visual Basic
Public Shared Readonly **Instance** As ULFactory

C#
public const ULFactory **Instance**;

備考

ULFactory はシングルトン・クラスであり、このクラスにはこのインスタンスだけが存在できることを意味します。

通常、このフィールドは使用されません。代わりに、ULFactory のこのインスタンスへの参照を取得するには、System.Data.Common.DbProviderFactories.GetFactory(String) を使用します。例については、「ULFactory クラス」 326 ページを参照してください。

制限：ULFactory クラスは、.NET Compact Framework 2.0 では使用できません。

参照

- 「ULFactory クラス」 326 ページ
- 「ULFactory メンバ」 327 ページ
- DbProviderFactories.GetFactory

CanCreateDataSourceEnumerator プロパティ

Ultra Light.NET で DbDataSourceEnumerator がサポートされないことを示す false を返します。

構文

Visual Basic
Public Overrides Readonly Property **CanCreateDataSourceEnumerator** As Boolean

C#
public override bool **CanCreateDataSourceEnumerator** { get;}

プロパティ値

ULFactory で CreateDataSourceEnumerator() メソッドを実装していないことを示す false。

参照

- 「ULFactory クラス」 326 ページ
- 「ULFactory メンバ」 327 ページ

CreateCommand メソッド

厳密に型指定された System.Data.Common.DbCommand インスタンスを返します。

構文

Visual Basic
Public Overrides Function **CreateCommand()** As DbCommand

C#
public override DbCommand **CreateCommand();**

戻り値

DbCommand として型指定された新しい ULCommand インスタンス。

参照

- 「ULFactory クラス」 326 ページ
- 「ULFactory メンバ」 327 ページ
- DbCommand
- 「ULCommand クラス」 91 ページ

CreateCommandBuilder メソッド

厳密に型指定された System.Data.Common.DbCommandBuilder インスタンスを返します。

構文

Visual Basic
Public Overrides Function **CreateCommandBuilder()** As DbCommandBuilder

C#
public override DbCommandBuilder **CreateCommandBuilder();**

戻り値

DbCommandBuilder として型指定された新しい ULCommandBuilder インスタンス。

参照

- 「ULFactory クラス」 326 ページ
- 「ULFactory メンバ」 327 ページ
- DbCommandBuilder
- 「ULCommandBuilder クラス」 129 ページ

CreateConnection メソッド

厳密に型指定された System.Data.Common.DbConnection インスタンスを返します。

構文

Visual Basic
Public Overrides Function **CreateConnection()** As DbConnection

C#
public override DbConnection **CreateConnection();**

戻り値

DbConnection として型指定された新しい ULConnection インスタンス。

参照

- 「ULFactory クラス」 326 ページ
- 「ULFactory メンバ」 327 ページ
- DbConnection
- 「ULConnection クラス」 139 ページ

CreateConnectionStringBuilder メソッド

厳密に型指定された System.Data.Common.DbConnectionStringBuilder インスタンスを返します。

構文

Visual Basic
Public Overrides Function **CreateConnectionStringBuilder()** As DbConnectionStringBuilder

C#
public override DbConnectionstring Builder **CreateConnectionStringBuilder();**

戻り値

DbConnectionStringBuilder として型指定された新しい ULConnectionStringBuilder インスタンス。

参照

- 「ULFactory クラス」 326 ページ
- 「ULFactory メンバ」 327 ページ
- DbConnectionStringBuilder
- 「ULConnectionStringBuilder クラス」 203 ページ

CreateDataAdapter メソッド

厳密に型指定された System.Data.Common.DbDataAdapter インスタンスを返します。

構文

Visual Basic
Public Overrides Function **CreateDataAdapter()** As DbDataAdapter

C#
public override DbDataAdapter **CreateDataAdapter();**

戻り値

DbDataAdapter として型指定された新しい ULDataAdapter インスタンス。

参照

- 「ULFactory クラス」 326 ページ
- 「ULFactory メンバ」 327 ページ
- DbDataAdapter
- 「ULDataAdapter クラス」 242 ページ

CreateParameter メソッド

厳密に型指定された System.Data.Common.DbParameter インスタンスを返します。

構文

Visual Basic
Public Overrides Function **CreateParameter()** As DbParameter

C#
public override DbParameter **CreateParameter();**

戻り値

DbParameter として型指定された新しい ULParameter インスタンス。

参照

- 「ULFactory クラス」 326 ページ
- 「ULFactory メンバ」 327 ページ
- DbParameter
- 「ULParameter クラス」 375 ページ

ULFileTransfer クラス

UL 拡張： Mobile Link サーバを使用して、リモート・データベースからファイルを転送します。このクラスは継承できません。

構文

Visual Basic
Public NotInheritable Class **ULFileTransfer**

C#
public sealed class **ULFileTransfer**

備考

ファイルを転送するためにデータベースに接続する必要はありません。ただし、Ultra Light ランタイムを使用した Ultra Light データベースをアプリケーションで使用する場合は、この API や Ultra Light.NET API を使用する前に、ULDatabaseManager.RuntimeType に適切な値を設定する必要があります。

ファイルを転送するには、ULFileTransfer.FileName、ULFileTransfer.Stream、ULFileTransfer.UserName、ULFileTransfer.Version を設定する必要があります。

参照

- 「ULFileTransfer メンバ」 332 ページ
- 「RuntimeType プロパティ」 256 ページ
- 「FileName プロパティ」 338 ページ
- 「Stream プロパティ」 341 ページ
- 「UserName プロパティ」 343 ページ
- 「Version プロパティ」 344 ページ

ULFileTransfer メンバ

パブリック・コンストラクタ

メンバ名	説明
「ULFileTransfer コンストラクタ」 334 ページ	ULFileTransfer オブジェクトを初期化します。接続を開いてから、データベース操作を実行してください。

パブリック・プロパティ

メンバ名	説明
「AuthStatus プロパティ」 335 ページ	前回行われたファイル転送の認証ステータス・コードを返します。

メンバ名	説明
「AuthValue プロパティ」 335 ページ	カスタム・ユーザ認証同期スクリプトからの戻り値を返します。
「AuthenticationParms プロパティ」 336 ページ	カスタム・ユーザ認証スクリプト (Mobile Link authenticate_parameters 接続イベント) のパラメータを指定します。
「DestinationFileName プロパティ」 336 ページ	ダウンロード・ファイルのローカル名を指定します。
「DestinationPath プロパティ」 337 ページ	ファイルをダウンロードする場所を指定します。
「DownloadedFile プロパティ」 337 ページ	前回行われたファイル転送時にファイルが実際にダウンロードされたかどうかを確認します。
「FileAuthCode プロパティ」 338 ページ	前回行われたファイル転送の authenticate_file_transfer スクリプトからの戻り値を返します。
「FileName プロパティ」 338 ページ	ダウンロードするファイルの名前を指定します。
「ForceDownload プロパティ」 339 ページ	ファイルが既に存在する場合に、ファイルのダウンロードを強制するかどうかを指定します。
「Password プロパティ」 340 ページ	UserName で指定されたユーザの Mobile Link パスワードです。
「ResumePartialDownload プロパティ」 340 ページ	前の部分的なダウンロードを再開するか、破棄するかを指定します。
「Stream プロパティ」 341 ページ	ファイル転送に使用する Mobile Link 同期ストリームを指定します。
「StreamErrorCode プロパティ」 341 ページ	前回行われたファイル転送のストリーム自体によってレポートされるエラーを返します。
「StreamErrorSystem プロパティ」 342 ページ	ストリーム・エラー・システム固有のコードを返します。
「StreamParms プロパティ」 342 ページ	同期ストリームの設定パラメータを指定します。
「UserName プロパティ」 343 ページ	Mobile Link サーバが Mobile Link クライアントをユニークに識別するユーザ名です。

メンバ名	説明
「Version プロパティ」 344 ページ	使用する同期スクリプトを指定します。

パブリック・メソッド

メンバ名	説明
「DownloadFile メソッド」 344 ページ	このオブジェクトのプロパティで指定されたファイルをダウンロードします。

参照

- 「ULFileTransfer クラス」 332 ページ
- 「RuntimeType プロパティ」 256 ページ
- 「FileName プロパティ」 338 ページ
- 「Stream プロパティ」 341 ページ
- 「UserName プロパティ」 343 ページ
- 「Version プロパティ」 344 ページ

ULFileTransfer コンストラクタ

ULFileTransfer オブジェクトを初期化します。接続を開いてから、データベース操作を実行してください。

構文

Visual Basic
Public Sub **New()**

C#
public **ULFileTransfer();**

備考

ファイルを転送するためにデータベースに接続する必要はありません。ただし、Ultra Light ランタイムを使用した Ultra Light データベースをアプリケーションで使用する場合は、この API や Ultra Light.NET API を使用する前に、ULDatabaseManager.RuntimeType に適切な値を設定する必要があります。

ULFileTransfer オブジェクトに ULFileTransfer.FileName、ULFileTransfer.Stream、ULFileTransfer.UserName、ULFileTransfer.Version が設定されていないと、ファイルを転送できません。

参照

- 「ULFileTransfer クラス」 332 ページ
- 「ULFileTransfer メンバ」 332 ページ
- 「DownloadFile() メソッド」 344 ページ
- 「RuntimeType プロパティ」 256 ページ
- 「FileName プロパティ」 338 ページ
- 「Stream プロパティ」 341 ページ
- 「UserName プロパティ」 343 ページ
- 「Version プロパティ」 344 ページ

AuthStatus プロパティ

前回行われたファイル転送の認証ステータス・コードを返します。

構文

Visual Basic
Public Readonly Property **AuthStatus** As ULAuthStatusCode

C#
public ULAuthStatusCode **AuthStatus** { get;}

プロパティ値

前回行われたファイル転送の認証ステータスを示す ULAuthStatusCode 値の 1 つ。

参照

- 「ULFileTransfer クラス」 332 ページ
- 「ULFileTransfer メンバ」 332 ページ
- 「ULAuthStatusCode 列挙体」 61 ページ

AuthValue プロパティ

カスタム・ユーザ認証同期スクリプトからの戻り値を返します。

構文

Visual Basic
Public Readonly Property **AuthValue** As Long

C#
public long **AuthValue** { get;}

プロパティ値

カスタム・ユーザ認証同期スクリプトから返された long integer。

参照

- 「ULFileTransfer クラス」 332 ページ
- 「ULFileTransfer メンバ」 332 ページ

AuthenticationParms プロパティ

カスタム・ユーザ認証スクリプト (Mobile Link authenticate_parameters 接続イベント) のパラメータを指定します。

構文

Visual Basic
Public Property **AuthenticationParms** As String

C#
public string **AuthenticationParms** { get; set; }

プロパティ値

それぞれに認証パラメータが格納された、文字列の配列 (配列のエントリが NULL であると、同期エラーになります)。デフォルト値は NULL 参照 (Visual Basic の Nothing) で、認証パラメータは指定されません。

備考

最初の 255 文字のみが使用されます。また、各文字列は 128 文字以下であることが必要です (長すぎる文字列は、Mobile Link サーバに送信されるときにトランケートされます)。

参照

- 「ULFileTransfer クラス」 332 ページ
- 「ULFileTransfer メンバ」 332 ページ

DestinationFileName プロパティ

ダウンロード・ファイルのローカル名を指定します。

構文

Visual Basic
Public Property **DestinationFileName** As String

C#
public string **DestinationFileName** { get; set; }

プロパティ値

ダウンロード・ファイルのローカル名を指定する文字列。値が NULL 参照 (Visual Basic の Nothing) の場合、FileName が使用されます。デフォルト値は NULL 参照 (Visual Basic の Nothing) です。

参照

- 「ULFileTransfer クラス」 332 ページ
- 「ULFileTransfer メンバ」 332 ページ
- 「FileName プロパティ」 338 ページ

DestinationPath プロパティ

ファイルをダウンロードする場所を指定します。

構文

Visual Basic
Public Property **DestinationPath** As String

C#
public string **DestinationPath** { get; set; }

プロパティ値

ファイルのダウンロード先ディレクトリを指定する文字列。デフォルト値は NULL 参照 (Visual Basic の Nothing) です。

備考

デフォルトのディレクトリは、デバイスのオペレーティング・システムによって異なります。

- Windows モバイル・デバイスの場合は、DestinationPath は NULL 参照 (Visual Basic の Nothing) で、ファイルはルート (\) ディレクトリに格納されます。
- デスクトップ・アプリケーションの場合は、DestinationPath は NULL 参照 (Visual Basic の Nothing) で、ファイルは現在のディレクトリに格納されます。

参照

- 「ULFileTransfer クラス」 332 ページ
- 「ULFileTransfer メンバ」 332 ページ
- 「ForceDownload プロパティ」 339 ページ
- 「DownloadFile() メソッド」 344 ページ

DownloadedFile プロパティ

前回行われたファイル転送時にファイルが実際にダウンロードされたかどうかを確認します。

構文

Visual Basic
Public Readonly Property **DownloadedFile** As Boolean

C#
public bool **DownloadedFile** { get;}

プロパティ値

ファイルがダウンロードされている場合は true、それ以外の場合は false。

備考

DownloadFile() が呼び出された時点でファイルがすでに最新だった場合、結果は true になりますが、DownloadedFile は false に設定されます。エラーが発生して DownloadFile() が false を返した場合は、DownloadedFile は false に設定されます。

参照

- 「ULFileTransfer クラス」 332 ページ
- 「ULFileTransfer メンバ」 332 ページ
- 「DownloadFile() メソッド」 344 ページ
- 「DownloadFile() メソッド」 344 ページ

FileAuthCode プロパティ

前回行われたファイル転送の authenticate_file_transfer スクリプトからの戻り値を返します。

構文

Visual Basic
Public Readonly Property **FileAuthCode** As UInt16

C#
public ushort **FileAuthCode** { get;}

プロパティ値

前回行われたファイル転送の authenticate_file_transfer スクリプトから戻された unsigned short の整数。

参照

- 「ULFileTransfer クラス」 332 ページ
- 「ULFileTransfer メンバ」 332 ページ

FileName プロパティ

ダウンロードするファイルの名前を指定します。

構文

Visual Basic
Public Property **FileName** As String

C#
public string **FileName** { get; set; }

プロパティ値

Mobile Link サーバによって認識されたファイルの名前を指定する文字列。このプロパティにはデフォルト値がないので、明示的に設定してください。

備考

FileName は、Mobile Link を実行するサーバ上にあるファイルの名前です。指定されたファイルは、まず UserName サブディレクトリで、次にルート・ディレクトリで検索されます (ルート・ディレクトリは、Mobile Link サーバの -ftr オプションで指定)。FileName にはドライブまたはパスの情報を含めないでください。そのような情報を含めると、ファイルが見からなくなります。たとえば、"myfile.txt" は有効ですが、"somedir¥myfile.txt"、"..¥myfile.txt"、"c:¥myfile.txt" は無効です。

参照

- 「ULFileTransfer クラス」 332 ページ
- 「ULFileTransfer メンバ」 332 ページ
- 「DestinationFileName プロパティ」 336 ページ
- 「DownloadFile() メソッド」 344 ページ

ForceDownload プロパティ

ファイルが既に存在する場合に、ファイルのダウンロードを強制するかどうかを指定します。

構文

Visual Basic
Public Property **ForceDownload** As Boolean

C#
public bool **ForceDownload** { get; set; }

プロパティ値

ファイルが存在する場合に、ファイルのダウンロードを強制する場合は true、通常のダウンロード・チェックを実行する場合は false。デフォルトは false です。

備考

ForceDownload が true であると、変更が加えられていない場合でもファイルは常にダウンロードされ、部分的なダウンロードは破棄されます。ForceDownload が false であると、サーバのバージョンとクライアントのバージョンが異なる場合にのみ、ファイルはダウンロードされます。クライアントまたはサーバでファイルが変更されている場合、ForceDownload を true に設定すると、クライアント・バージョンがサーバ・バージョンで上書きされることに注意してください。

参照

- 「ULFileTransfer クラス」 332 ページ
- 「ULFileTransfer メンバ」 332 ページ
- 「ResumePartialDownload プロパティ」 340 ページ
- 「DownloadFile() メソッド」 344 ページ

Password プロパティ

UserName で指定されたユーザの Mobile Link パスワードです。

構文

Visual Basic
Public Property **Password** As String

C#
public string **Password** { get; set; }

プロパティ値

Mobile Link パスワードを指定する文字列。デフォルト値は NULL 参照 (Visual Basic の Nothing) で、パスワードは指定されません。

備考

Mobile Link ユーザ名とパスワードは他のデータベース・ユーザ ID やパスワードとは別のもので、アプリケーションを Mobile Link サーバに対して識別し、認証するために使用されます。

参照

- 「ULFileTransfer クラス」 332 ページ
- 「ULFileTransfer メンバ」 332 ページ
- 「UserName プロパティ」 343 ページ
- 「DownloadFile() メソッド」 344 ページ

ResumePartialDownload プロパティ

前の部分的なダウンロードを再開するか、破棄するかを指定します。

構文

Visual Basic
Public Property **ResumePartialDownload** As Boolean

C#
public bool **ResumePartialDownload** { get; set; }

プロパティ値

前の部分的なダウンロードを再開する場合は true、破棄する場合は false。デフォルトは false です。

備考

Ultra Light.NET では、通信エラーや、ユーザによる ULFileTransferListener からのアボートが原因で失敗したダウンロードを再起動できます。Ultra Light.NET は、ダウンロードを受信しながら処理します。ダウンロードが中断した場合は、部分的なダウンロード・ファイルが保持されるため、次の転送中に再開できます。

ファイルがサーバ上で更新されている場合は、部分的なダウンロードは破棄され、新しくダウンロードが開始されます。

参照

- 「ULFileTransfer クラス」 332 ページ
- 「ULFileTransfer メンバ」 332 ページ
- 「ForceDownload プロパティ」 339 ページ
- 「DownloadFile() メソッド」 344 ページ

Stream プロパティ

ファイル転送に使用する Mobile Link 同期ストリームを指定します。

構文

Visual Basic
Public Property **Stream** As ULStreamType

C#
public ULStreamType **Stream** { get; set; }

プロパティ値

使用する同期ストリームのタイプを指定する ULStreamType 値の 1 つ。デフォルトは ULStreamType.TCPIP です。

備考

ほとんどの同期ストリームでは、Mobile Link サーバのアドレスを識別したり、その他の動作を制御したりするパラメータが必要です。これらのパラメータは、ULFileTransfer.StreamParms で指定します。

ストリーム・タイプが、プラットフォームに適さない無効な値に設定されていると、ストリーム・タイプは ULStreamType.TCPIP に設定されます。

参照

- 「ULFileTransfer クラス」 332 ページ
- 「ULFileTransfer メンバ」 332 ページ
- 「ULStreamType 列挙体」 512 ページ
- 「StreamParms プロパティ」 342 ページ
- 「DownloadFile() メソッド」 344 ページ
- 「ULStreamType 列挙体」 512 ページ

StreamErrorCode プロパティ

前回行われたファイル転送のストリーム自体によってレポートされるエラーを返します。

構文

Visual Basic
Public Readonly Property **StreamErrorCode** As ULStreamErrorCode

C#
public ULStreamErrorCode **StreamErrorCode** { get;}

プロパティ値

ストリーム自体によってレポートされるエラーを示す ULStreamErrorCode 値の 1 つ。ただし、エラーが発生しなかった場合は ULStreamErrorCode.NONE。

参照

- 「ULFileTransfer クラス」 332 ページ
- 「ULFileTransfer メンバ」 332 ページ
- 「ULStreamErrorCode 列挙体」 482 ページ
- 「ULStreamErrorCode 列挙体」 482 ページ

StreamErrorSystem プロパティ

ストリーム・エラー・システム固有のコードを返します。

構文

Visual Basic
Public Readonly Property **StreamErrorSystem** As Integer

C#
public int **StreamErrorSystem** { get;}

プロパティ値

ストリーム・エラー・システム固有のコードを示す整数。

参照

- 「ULFileTransfer クラス」 332 ページ
- 「ULFileTransfer メンバ」 332 ページ

StreamParms プロパティ

同期ストリームの設定パラメータを指定します。

構文

Visual Basic
Public Property **StreamParms** As String

C#
public string **StreamParms** { get; set; }

プロパティ値

キーワードと値の組み合わせがセミコロンで区切られたリスト形式の、ストリームのパラメータを指定する文字列。デフォルト値は NULL 参照 (Visual Basic の Nothing) です。

備考

特定のストリームのタイプの設定方法については、「Ultra Light 同期ストリームのネットワーク・プロトコルのオプション」『Ultra Light データベース管理とリファレンス』を参照してください。

StreamParms は、同期ストリームに使用されるすべてのパラメータが含まれている文字列です。パラメータは、「名前=値」のペアをセミコロンで区切ったリスト ("param1=value1;param2=value2") で指定します。

参照

- 「ULFileTransfer クラス」 332 ページ
- 「ULFileTransfer メンバ」 332 ページ
- 「Stream プロパティ」 341 ページ
- 「DownloadFile() メソッド」 344 ページ
- 「ULStreamType 列挙体」 512 ページ

UserName プロパティ

Mobile Link サーバが Mobile Link クライアントをユニークに識別するユーザ名です。

構文

Visual Basic
Public Property **UserName** As String

C#
public string **UserName** { get; set; }

プロパティ値

ユーザ名を指定する文字列。このプロパティにはデフォルト値がないので、明示的に設定してください。

備考

Mobile Link サーバは、この値を使用して、ダウンロードするファイルを特定します。Mobile Link ユーザ名とパスワードは他のデータベース・ユーザ ID やパスワードとは別のもので、アプリケーションを Mobile Link サーバに対して識別し、認証するために使用されます。

参照

- 「ULFileTransfer クラス」 332 ページ
- 「ULFileTransfer メンバ」 332 ページ
- 「Password プロパティ」 340 ページ
- 「DownloadFile() メソッド」 344 ページ

Version プロパティ

使用する同期スクリプトを指定します。

構文

Visual Basic
Public Property **Version** As String

C#
public string **Version** { get; set; }

プロパティ値

使用する同期スクリプトのバージョンを指定する文字列。このプロパティにはデフォルト値がないので、明示的に設定してください。

備考

統合データベースの同期スクリプトは、それぞれバージョン文字列でマーク付けされます。Ultra Light アプリケーションは、バージョン文字列により、同期スクリプトのセットから選択できます。

参照

- 「ULFileTransfer クラス」 332 ページ
- 「ULFileTransfer メンバ」 332 ページ
- 「DownloadFile() メソッド」 344 ページ

DownloadFile メソッド

このオブジェクトのプロパティで指定されたファイルをダウンロードします。

DownloadFile() メソッド

このオブジェクトのプロパティで指定されたファイルをダウンロードします。

構文

Visual Basic
Public Function **DownloadFile()** As Boolean

C#
public bool **DownloadFile();**

戻り値

成功であった場合は true、それ以外の場合は false (理由については ULFileTransfer.StreamErrorCode とその他のステータスに関するプロパティを確認)。

備考

ULFileTransfer.FileName で指定されたファイルが、Mobile Link サーバによって ULFileTransfer.DestinationPath にダウンロードされます。このとき、ULFileTransfer.Stream、ULFileTransfer.UserName、ULFileTransfer.Password、および ULFileTransfer.Version が使用されます。ダウンロードに影響を及ぼすその他のプロパティは、ULFileTransfer.DestinationFileName、ULFileTransfer.AuthenticationParms、ULFileTransfer.ForceDownload、ULFileTransfer.ResumePartialDownload です。

ファイルが破損することを防ぐため、Ultra Light.NET はテンポラリ・ファイルにダウンロードし、ダウンロードが完了したときのみ対象ファイルを置換します。

結果の詳細なステータスは、このオブジェクトの ULFileTransfer.AuthStatus、ULFileTransfer.AuthStatus、ULFileTransfer.FileAuthCode、ULFileTransfer.DownloadedFile、ULFileTransfer.StreamErrorCode、および ULFileTransfer.StreamErrorSystem でレポートされます。

参照

- 「ULFileTransfer クラス」 332 ページ
- 「ULFileTransfer メンバ」 332 ページ
- 「DownloadFile メソッド」 344 ページ
- 「DownloadFile(ULFileTransferProgressListener) メソッド」 345 ページ
- 「FileName プロパティ」 338 ページ
- 「DestinationPath プロパティ」 337 ページ
- 「Stream プロパティ」 341 ページ
- 「UserName プロパティ」 343 ページ
- 「Password プロパティ」 340 ページ
- 「Version プロパティ」 344 ページ
- 「DestinationFileName プロパティ」 336 ページ
- 「AuthenticationParms プロパティ」 336 ページ
- 「ForceDownload プロパティ」 339 ページ
- 「ResumePartialDownload プロパティ」 340 ページ
- 「AuthStatus プロパティ」 335 ページ
- 「AuthValue プロパティ」 335 ページ
- 「FileAuthCode プロパティ」 338 ページ
- 「DownloadedFile プロパティ」 337 ページ
- 「StreamErrorCode プロパティ」 341 ページ
- 「StreamErrorSystem プロパティ」 342 ページ
- 「StreamErrorCode プロパティ」 341 ページ

DownloadFile(ULFileTransferProgressListener) メソッド

指定されたリスナに送信されるプログレス・イベントとともに、このオブジェクトのプロパティで指定されたファイルをダウンロードします。

構文

Visual Basic
Public Function **DownloadFile(_**

```
  ByVal listener As ULFileTransferProgressListener _
) As Boolean

C#
public bool DownloadFile(
  ULFileTransferProgressListener listener
);
```

パラメータ

- **listener**　ファイル転送プログレス・イベントを受信するオブジェクト。

戻り値

成功であった場合は true、それ以外の場合は false (理由については ULFileTransfer.StreamErrorCode とその他のステータスに関するプロパティを確認)。

備考

ULFileTransfer.FileName で指定されたファイルが、Mobile Link サーバによって ULFileTransfer.DestinationPath にダウンロードされます。このとき、ULFileTransfer.Stream、ULFileTransfer.UserName、ULFileTransfer.Password、および ULFileTransfer.Version が使用されます。ダウンロードに影響を及ぼすその他のプロパティは、ULFileTransfer.DestinationFileName、ULFileTransfer.AuthenticationParms、ULFileTransfer.ForceDownload、ULFileTransfer.ResumePartialDownload です。

ファイルが破損することを防ぐため、Ultra Light.NET はテンポラリ・ファイルにダウンロードし、ダウンロードが完了したときのみ対象ファイルを置換します。

結果の詳細なステータスは、このオブジェクトの ULFileTransfer.AuthStatus、ULFileTransfer.AuthStatus、ULFileTransfer.FileAuthCode、ULFileTransfer.DownloadedFile、ULFileTransfer.StreamErrorCode、および ULFileTransfer.StreamErrorSystem でレポートされます。

エラーが発生すると、リスナにはデータが送信されないことがあります。

参照

- 「ULFileTransfer クラス」 332 ページ
- 「ULFileTransfer メンバ」 332 ページ
- 「DownloadFile メソッド」 344 ページ
- 「FileName プロパティ」 338 ページ
- 「DestinationPath プロパティ」 337 ページ
- 「Stream プロパティ」 341 ページ
- 「UserName プロパティ」 343 ページ
- 「Password プロパティ」 340 ページ
- 「Version プロパティ」 344 ページ
- 「DestinationFileName プロパティ」 336 ページ
- 「AuthenticationParms プロパティ」 336 ページ
- 「ForceDownload プロパティ」 339 ページ
- 「ResumePartialDownload プロパティ」 340 ページ
- 「AuthStatus プロパティ」 335 ページ
- 「AuthValue プロパティ」 335 ページ
- 「FileAuthCode プロパティ」 338 ページ
- 「DownloadedFile プロパティ」 337 ページ
- 「StreamErrorCode プロパティ」 341 ページ
- 「StreamErrorSystem プロパティ」 342 ページ
- 「StreamErrorCode プロパティ」 341 ページ

ULFileTransferProgressData クラス

UL 拡張：ファイル転送の進行状況のモニタリング・データを返します。

構文

Visual Basic
Public Class **ULFileTransferProgressData**

C#
public class **ULFileTransferProgressData**

参照

- 「ULFileTransferProgressData メンバ」 348 ページ
- 「ULFileTransferProgressListener インタフェース」 351 ページ

ULFileTransferProgressData メンバ

パブリック・フィールド

メンバ名	説明
「FLAG_IS_BLOCKING フィールド」 349 ページ	Mobile Link サーバからの応答の待機中、ファイル転送はブロックされていることを示すフラグです。このフィールドは定数で、読み込み専用です。

パブリック・プロパティ

メンバ名	説明
「BytesReceived プロパティ」 349 ページ	現在までに受信したバイト数を返します。
「FileSize プロパティ」 349 ページ	転送されるファイルのサイズを返します。
「Flags プロパティ」 350 ページ	現在の状態に関連する追加情報を示す、現在のファイル転送フラグを返します。
「ResumedAtSize プロパティ」 350 ページ	転送が再開されるファイル内の場所を返します。

参照

- 「ULFileTransferProgressData クラス」 348 ページ
- 「ULFileTransferProgressListener インタフェース」 351 ページ

FLAG_IS_BLOCKING フィールド

Mobile Link サーバからの応答の待機中、ファイル転送はブロックされていることを示すフラグです。このフィールドは定数で、読み込み専用です。

構文

Visual Basic
Public Shared **FLAG_IS_BLOCKING** As Integer

C#
public const int **FLAG_IS_BLOCKING**;

参照

- 「ULFileTransferProgressData クラス」 348 ページ
- 「ULFileTransferProgressData メンバ」 348 ページ

BytesReceived プロパティ

現在までに受信したバイト数を返します。

構文

Visual Basic
Public Readonly Property **BytesReceived** As UInt64

C#
public ulong **BytesReceived** { get;}

プロパティ値

現在までに受信したバイト数。

参照

- 「ULFileTransferProgressData クラス」 348 ページ
- 「ULFileTransferProgressData メンバ」 348 ページ

FileSize プロパティ

転送されるファイルのサイズを返します。

構文

Visual Basic
Public Readonly Property **FileSize** As UInt64

C#
public ulong **FileSize** { get;}

プロパティ値

ファイルのサイズ (バイト単位)。

参照

- 「ULFileTransferProgressData クラス」 348 ページ
- 「ULFileTransferProgressData メンバ」 348 ページ

Flags プロパティ

現在の状態に関連する追加情報を示す、現在のファイル転送フラグを返します。

構文

Visual Basic
Public Readonly Property **Flags** As Integer

C#
public int **Flags** { get;}

プロパティ値

フラグの組み合わせを保持する整数。

参照

- 「ULFileTransferProgressData クラス」 348 ページ
- 「ULFileTransferProgressData メンバ」 348 ページ
- 「FLAG_IS_BLOCKING フィールド」 349 ページ

ResumedAtSize プロパティ

転送が再開されるファイル内の場所を返します。

構文

Visual Basic
Public Readonly Property **ResumedAtSize** As UInt64

C#
public ulong **ResumedAtSize** { get;}

プロパティ値

以前に転送されたバイト数。

参照

- 「ULFileTransferProgressData クラス」 348 ページ
- 「ULFileTransferProgressData メンバ」 348 ページ

ULFileTransferProgressListener インタフェース

UL 拡張：ファイル転送プログレス・イベントを受信するリスナ・インタフェースです。

構文

Visual Basic
Public Interface **ULFileTransferProgressListener**

C#
public interface **ULFileTransferProgressListener**

参照

- 「ULFileTransferProgressListener メンバ」 351 ページ
- 「DownloadFile(ULFileTransferProgressListener) メソッド」 345 ページ

ULFileTransferProgressListener メンバ

パブリック・メソッド

メンバ名	説明
「FileTransferProgressed メソッド」 351 ページ	ユーザに進行状況を通知するために、ファイル転送中に起動されます。このメソッドは、転送をキャンセルする場合は true を、続行する場合は false を返す必要があります。

参照

- 「ULFileTransferProgressListener インタフェース」 351 ページ
- 「DownloadFile(ULFileTransferProgressListener) メソッド」 345 ページ

FileTransferProgressed メソッド

ユーザに進行状況を通知するために、ファイル転送中に起動されます。このメソッドは、転送をキャンセルする場合は true を、続行する場合は false を返す必要があります。

構文

Visual Basic
```
Public Function FileTransferProgressed( _
  ByVal data As ULFileTransferProgressData _
) As Boolean
```

C#
```
public bool FileTransferProgressed(
  ULFileTransferProgressData data
);
```

パラメータ

- **data** 最新の同期のプログレス・データを保持している ULFileTransferProgressData オブジェクト。

戻り値

このメソッドは、転送をキャンセルする場合は true を、続行する場合は false を返す必要があります。

備考

FileTransferProgressed の呼び出し中に、Ultra Light.NET API のメソッドを呼び出さないでください。

参照

- 「ULFileTransferProgressListener インタフェース」 351 ページ
- 「ULFileTransferProgressListener メンバ」 351 ページ
- 「ULFileTransferProgressData クラス」 348 ページ

ULIndexSchema クラス

UL 拡張：Ultra Light テーブルのインデックスのスキーマを表します。このクラスは継承できません。

構文

Visual Basic
Public NotInheritable Class **ULIndexSchema**

C#
public sealed class **ULIndexSchema**

備考

このクラスにはコンストラクタがありません。インデックスのスキーマは、ULTableSchema の ULTableSchema.PrimaryKey、ULTableSchema.GetIndex(string)、および ULTableSchema.GetOptimalIndex(int) を使用して作成されます。

参照

- 「ULIndexSchema メンバ」 353 ページ
- 「PrimaryKey プロパティ」 575 ページ
- 「GetIndex メソッド」 577 ページ
- 「GetOptimalIndex メソッド」 579 ページ
- 「ULTableSchema クラス」 571 ページ

ULIndexSchema メンバ

パブリック・プロパティ

メンバ名	説明
「ColumnCount プロパティ」 354 ページ	インデックス内のカラム数を返します。
「IsForeignKey プロパティ」 355 ページ	インデックスが外部キーであるかどうかをチェックします。
「IsForeignKeyCheckOnCommit プロパティ」 355 ページ	外部キーの参照整合性のチェックが、コミット時に行われるか、挿入時と更新時に行われるかを確認します。
「IsForeignKeyNullable プロパティ」 356 ページ	外部キーが NULL 入力可であるかどうかをチェックします。
「IsOpen プロパティ」 356 ページ	インデックス・スキーマが開いているか、閉じているかを調べます。

メンバ名	説明
「IsPrimaryKey プロパティ」 357 ページ	インデックスがプライマリ・キーであるかどうかをチェックします。
「IsUniqueIndex プロパティ」 357 ページ	インデックスがユニークであるかどうかをチェックします。
「IsUniqueKey プロパティ」 358 ページ	インデックスがユニーク・キーであるかどうかをチェックします。
「Name プロパティ」 358 ページ	インデックスの名前を返します。
「ReferencedIndexName プロパティ」 359 ページ	インデックスが外部キーである場合、参照されるプライマリ・インデックスの名前です。
「ReferencedTableName プロパティ」 359 ページ	インデックスが外部キーである場合、参照されるプライマリ・テーブルの名前です。

パブリック・メソッド

メンバ名	説明
「GetColumnName メソッド」 359 ページ	このインデックス内の *colOrdinalInIndex* 番目のカラム名を返します。
「IsColumnDescending メソッド」 360 ページ	指定されたカラムが、インデックスによって降順で使用されるかどうかをチェックします。

参照

- 「ULIndexSchema クラス」 353 ページ
- 「PrimaryKey プロパティ」 575 ページ
- 「GetIndex メソッド」 577 ページ
- 「GetOptimalIndex メソッド」 579 ページ
- 「ULTableSchema クラス」 571 ページ

ColumnCount プロパティ

インデックス内のカラム数を返します。

構文

Visual Basic
Public Readonly Property **ColumnCount** As Short

C#
public short **ColumnCount** { get;}

プロパティ値

インデックス内のカラム数。

備考

インデックス内のカラムの順序の範囲は、1 ～ ColumnCount です。

カラムの順序とカウントは、スキーマのアップグレード中に変更されることがあります。インデックスからのカラムの順序は、特定のテーブルの同じ物理的カラムを参照する場合であっても、テーブルまたは別のインデックスのカラム ID とは異なります。

参照

- 「ULIndexSchema クラス」 353 ページ
- 「ULIndexSchema メンバ」 353 ページ

IsForeignKey プロパティ

インデックスが外部キーであるかどうかをチェックします。

構文

Visual Basic
Public Readonly Property **IsForeignKey** As Boolean

C#
public bool **IsForeignKey** { get;}

プロパティ値

インデックスが外部キーである場合は true、外部キーでない場合は false。

備考

外部キー内のカラムは、別のテーブルの NULL 以外のユニーク・インデックスを参照することができます。

参照

- 「ULIndexSchema クラス」 353 ページ
- 「ULIndexSchema メンバ」 353 ページ

IsForeignKeyCheckOnCommit プロパティ

外部キーの参照整合性のチェックが、コミット時に行われるか、挿入時と更新時に行われるかを確認します。

構文

Visual Basic
Public Readonly Property **IsForeignKeyCheckOnCommit** As Boolean

C#
public bool **IsForeignKeyCheckOnCommit** { get;}

プロパティ値

参照整合性がコミット時にチェックされる場合は true、挿入時と更新時にチェックされる場合は false。

参照

- 「ULIndexSchema クラス」 353 ページ
- 「ULIndexSchema メンバ」 353 ページ
- 「IsForeignKey プロパティ」 355 ページ

IsForeignKeyNullable プロパティ

外部キーが NULL 入力可であるかどうかをチェックします。

構文

Visual Basic
Public Readonly Property **IsForeignKeyNullable** As Boolean

C#
public bool **IsForeignKeyNullable** { get;}

プロパティ値

外部キーが NULL 入力可の場合は true、NULL 入力不可の場合は false。

参照

- 「ULIndexSchema クラス」 353 ページ
- 「ULIndexSchema メンバ」 353 ページ
- 「IsForeignKey プロパティ」 355 ページ

IsOpen プロパティ

インデックス・スキーマが開いているか、閉じているかを調べます。

構文

Visual Basic
Public Readonly Property **IsOpen** As Boolean

C#
public bool **IsOpen** { get;}

プロパティ値

インデックス・スキーマが開いている場合は true、閉じている場合は false。

参照

- 「ULIndexSchema クラス」 353 ページ
- 「ULIndexSchema メンバ」 353 ページ

IsPrimaryKey プロパティ

インデックスがプライマリ・キーであるかどうかをチェックします。

構文

Visual Basic
Public Readonly Property **IsPrimaryKey** As Boolean

C#
public bool **IsPrimaryKey** { get;}

プロパティ値

インデックスがプライマリ・キーである場合は true、プライマリ・キーでない場合は false。

備考

プライマリ・キー内のカラムでは NULL は許可されません。

参照

- 「ULIndexSchema クラス」 353 ページ
- 「ULIndexSchema メンバ」 353 ページ

IsUniqueIndex プロパティ

インデックスがユニークであるかどうかをチェックします。

構文

Visual Basic
Public Readonly Property **IsUniqueIndex** As Boolean

C#
public bool **IsUniqueIndex** { get;}

プロパティ値

インデックスがユニークである場合は true、ユニークでない場合は false。

備考

ユニークなインデックス内のカラムは NULL であることがあります。

参照

- 「ULIndexSchema クラス」 353 ページ
- 「ULIndexSchema メンバ」 353 ページ

IsUniqueKey プロパティ

インデックスがユニーク・キーであるかどうかをチェックします。

構文

Visual Basic
Public Readonly Property **IsUniqueKey** As Boolean

C#
public bool **IsUniqueKey** { get;}

プロパティ値

インデックスがユニーク・キーである場合は true、ユニーク・キーでない場合は false。

備考

ユニーク・キー内のカラムでは NULL は許可されません。

参照

- 「ULIndexSchema クラス」 353 ページ
- 「ULIndexSchema メンバ」 353 ページ

Name プロパティ

インデックスの名前を返します。

構文

Visual Basic
Public Readonly Property **Name** As String

C#
public string **Name** { get;}

プロパティ値

インデックスの名前を指定する文字列。

参照

- 「ULIndexSchema クラス」 353 ページ
- 「ULIndexSchema メンバ」 353 ページ

ReferencedIndexName プロパティ

インデックスが外部キーである場合、参照されるプライマリ・インデックスの名前です。

構文

Visual Basic
Public Readonly Property **ReferencedIndexName** As String

C#
public string **ReferencedIndexName** { get;}

プロパティ値

参照されるプライマリ・インデックスの名前を指定する文字列。

参照

- 「ULIndexSchema クラス」 353 ページ
- 「ULIndexSchema メンバ」 353 ページ
- 「IsForeignKey プロパティ」 355 ページ

ReferencedTableName プロパティ

インデックスが外部キーである場合、参照されるプライマリ・テーブルの名前です。

構文

Visual Basic
Public Readonly Property **ReferencedTableName** As String

C#
public string **ReferencedTableName** { get;}

プロパティ値

参照されるプライマリ・テーブルの名前を指定する文字列。

参照

- 「ULIndexSchema クラス」 353 ページ
- 「ULIndexSchema メンバ」 353 ページ
- 「IsForeignKey プロパティ」 355 ページ

GetColumnName メソッド

このインデックス内の

colOrdinalInIndex 番目のカラム名を返します。

構文

Visual Basic
```
Public Function GetColumnName( _
  ByVal colOrdinalInIndex As Short _
) As String
```

C#
```
public string  GetColumnName(
  short colOrdinalInIndex
);
```

パラメータ

- **colOrdinalInIndex**　インデックス内の対象のカラムの順序。値は、[1,ColumnCount] の範囲内であることが必要です。

戻り値

カラムの名前。

備考

カラムの順序とカウントは、スキーマのアップグレード中に変更されることがあります。インデックスからのカラムの順序は、特定のテーブルの同じ物理的カラムを参照する場合であっても、テーブルまたは別のインデックスのカラム ID とは異なります。

参照

- 「ULIndexSchema クラス」 353 ページ
- 「ULIndexSchema メンバ」 353 ページ
- 「ColumnCount プロパティ」 354 ページ
- 「ColumnCount プロパティ」 354 ページ

IsColumnDescending メソッド

指定されたカラムが、インデックスによって降順で使用されるかどうかをチェックします。

構文

Visual Basic
```
Public Function IsColumnDescending( _
  ByVal name As String _
) As Boolean
```

C#
```
public bool IsColumnDescending(
  string  name
);
```

パラメータ

- **name**　カラムの名前。

戻り値

カラムが降順で使用される場合は true、昇順で使用される場合は false。

参照

- 「ULIndexSchema クラス」 353 ページ
- 「ULIndexSchema メンバ」 353 ページ
- 「GetColumnName メソッド」 359 ページ
- 「ColumnCount プロパティ」 354 ページ

ULInfoMessageEventArgs クラス

ULConnection.InfoMessage イベントのデータを提供します。このクラスは継承できません。

構文

Visual Basic
Public NotInheritable Class **ULInfoMessageEventArgs**
  Inherits EventArgs

C#
public sealed class **ULInfoMessageEventArgs**: EventArgs

参照

- 「ULInfoMessageEventArgs メンバ」 362 ページ
- 「InfoMessage イベント」 185 ページ

ULInfoMessageEventArgs メンバ

パブリック・プロパティ

メンバ名	説明
「Message プロパティ」 363 ページ	データベースによって返される情報メッセージまたは警告メッセージの文字列です。
「NativeError プロパティ」 363 ページ	データベースによって返される情報メッセージまたは警告に対応する SQL コードです。
「Source プロパティ」 363 ページ	メッセージを返す ADO.NET データ・プロバイダの名前です。

パブリック・メソッド

メンバ名	説明
「ToString メソッド」 364 ページ	ULConnection.InfoMessage イベントの文字列表現を返します。

参照

- 「ULInfoMessageEventArgs クラス」 362 ページ
- 「InfoMessage イベント」 185 ページ

Message プロパティ

データベースによって返される情報メッセージまたは警告メッセージの文字列です。

構文

Visual Basic
Public Readonly Property **Message** As String

C#
public string **Message** { get;}

プロパティ値

情報メッセージまたは警告メッセージが含まれる文字列。

参照

- 「ULInfoMessageEventArgs クラス」 362 ページ
- 「ULInfoMessageEventArgs メンバ」 362 ページ

NativeError プロパティ

データベースによって返される情報メッセージまたは警告に対応する SQLCODE です。

構文

Visual Basic
Public Readonly Property **NativeError** As ULSQLCode

C#
public ULSQLCode **NativeError** { get;}

プロパティ値

情報メッセージまたは警告の ULSQLCode 値。

参照

- 「ULInfoMessageEventArgs クラス」 362 ページ
- 「ULInfoMessageEventArgs メンバ」 362 ページ
- 「ULSQLCode 列挙体」 462 ページ

Source プロパティ

メッセージを返す ADO.NET データ・プロバイダの名前です。

構文

Visual Basic
Public Readonly Property **Source** As String

C#
public string **Source** { get;}

プロパティ値

文字列 "Ultra Light.NET Data Provider"。

参照

- 「ULInfoMessageEventArgs クラス」 362 ページ
- 「ULInfoMessageEventArgs メンバ」 362 ページ

ToString メソッド

ULConnection.InfoMessage イベントの文字列表現を返します。

構文

Visual Basic
Public Overrides Function **ToString()** As String

C#
public override string **ToString();**

戻り値

情報メッセージまたは警告メッセージの文字列。

参照

- 「ULInfoMessageEventArgs クラス」 362 ページ
- 「ULInfoMessageEventArgs メンバ」 362 ページ
- 「InfoMessage イベント」 185 ページ

ULInfoMessageEventHandler デリゲート

ULConnection.InfoMessage イベントを処理するメソッドを表します。

構文

Visual Basic

```
Public Delegate Sub ULInfoMessageEventHandler( _
   ByVal obj As Object, _
   ByVal args As ULInfoMessageEventArgs _
)
```

C#

```
public delegate void ULInfoMessageEventHandler(
   object obj,
   ULInfoMessageEventArgs args
);
```

参照

- 「InfoMessage イベント」 185 ページ
- 「ULInfoMessageEventArgs クラス」 362 ページ

ULMetaDataCollectionNames クラス

メタデータ・コレクションを取得する ULConnection.GetSchema(String,String[]) で使用する定数のリストを提供します。このクラスは継承できません。

構文

Visual Basic
Public NotInheritable Class **ULMetaDataCollectionNames**

C#
public sealed class **ULMetaDataCollectionNames**

参照

- 「ULMetaDataCollectionNames メンバ」 366 ページ
- 「GetSchema(String, String[]) メソッド」 174 ページ

ULMetaDataCollectionNames メンバ

パブリック・プロパティ

メンバ名	説明
「Columns プロパティ」 367 ページ	Columns コレクションを表す ULConnection.GetSchema(String) で使用する定数を提供します。
「DataSourceInformation プロパティ」 368 ページ	DataSourceInformation コレクションを表す ULConnection.GetSchema(String) で使用する定数を提供します。
「DataTypes プロパティ」 368 ページ	DataTypes コレクションを表す ULConnection.GetSchema(String) で使用する定数を提供します。
「ForeignKeys プロパティ」 369 ページ	ForeignKeys コレクションを表す ULConnection.GetSchema(String) で使用する定数を提供します。
「IndexColumns プロパティ」 370 ページ	IndexColumns コレクションを表す ULConnection.GetSchema(String) で使用する定数を提供します。
「Indexes プロパティ」 370 ページ	Indexes コレクションを表す ULConnection.GetSchema(String) で使用する定数を提供します。

メンバ名	説明
「MetaDataCollections プロパティ」 371 ページ	MetaDataCollections コレクションを表す ULConnection.GetSchema(String) で使用する定数を提供します。
「Publications プロパティ」 372 ページ	メタデータ・コレクションを取得する ULConnection.GetSchema(String,String[]) で使用する定数のリストを提供します。
「ReservedWords プロパティ」 372 ページ	ReservedWords コレクションを表す ULConnection.GetSchema(String) で使用する定数を提供します。
「Restrictions プロパティ」 373 ページ	Restrictions コレクションを表す ULConnection.GetSchema(String) で使用する定数を提供します。
「Tables プロパティ」 373 ページ	Tables コレクションを表す ULConnection.GetSchema(String) で使用する定数を提供します。

参照

- 「ULMetaDataCollectionNames クラス」 366 ページ
- 「GetSchema(String, String[]) メソッド」 174 ページ

Columns プロパティ

Columns コレクションを表す ULConnection.GetSchema(String) で使用する定数を提供します。

構文

Visual Basic
Public Shared Readonly Property **Columns** As String

C#
public const string **Columns** { get;}

プロパティ値

Columns コレクションの名前を表す文字列。

例

次のコードは、Columns コレクションを使用して DataTable を設定します。

```
' Visual Basic
Dim schema As DataTable =_
  conn.GetSchema( ULMetaDataCollectionNames.Columns )
```

```
// C#
DataTable schema =
  conn.GetSchema( ULMetaDataCollectionNames.Columns );
```

参照

- 「ULMetaDataCollectionNames クラス」 366 ページ
- 「ULMetaDataCollectionNames メンバ」 366 ページ
- 「GetSchema(String) メソッド」 174 ページ

DataSourceInformation プロパティ

DataSourceInformation コレクションを表す ULConnection.GetSchema(String) で使用する定数を提供します。

構文

Visual Basic
Public Shared Readonly Property **DataSourceInformation** As String

C#
public const string **DataSourceInformation** { get;}

プロパティ値

DataSourceInformation コレクションの名前を表す文字列。

例

次のコードは、DataSourceInformation コレクションを使用して DataTable を設定します。

```
' Visual Basic
Dim schema As DataTable = _
  conn.GetSchema( ULMetaDataCollectionNames.DataSourceInformation )

// C#
DataTable schema =
  conn.GetSchema( ULMetaDataCollectionNames.DataSourceInformation );
```

参照

- 「ULMetaDataCollectionNames クラス」 366 ページ
- 「ULMetaDataCollectionNames メンバ」 366 ページ
- 「GetSchema(String) メソッド」 174 ページ

DataTypes プロパティ

DataTypes コレクションを表す ULConnection.GetSchema(String) で使用する定数を提供します。

構文

Visual Basic
Public Shared Readonly Property **DataTypes** As String

C#
public const string **DataTypes** { get;}

プロパティ値

DataTypes コレクションの名前を表す文字列。

例

次のコードは、DataTypes コレクションを使用して DataTable を設定します。

```
' Visual Basic
Dim schema As DataTable = _
  conn.GetSchema( ULMetaDataCollectionNames.DataTypes )

// C#
DataTable schema =
  conn.GetSchema( ULMetaDataCollectionNames.DataTypes );
```

参照

- 「ULMetaDataCollectionNames クラス」 366 ページ
- 「ULMetaDataCollectionNames メンバ」 366 ページ
- 「GetSchema(String) メソッド」 174 ページ

ForeignKeys プロパティ

ForeignKeys コレクションを表す ULConnection.GetSchema(String) で使用する定数を提供します。

構文

Visual Basic
Public Shared Readonly Property **ForeignKeys** As String

C#
public const string **ForeignKeys** { get;}

プロパティ値

ForeignKeys コレクションの名前を表す文字列。

例

次のコードは、ForeignKeys コレクションを使用して DataTable を設定します。

```
' Visual Basic
Dim schema As DataTable = _
  conn.GetSchema( ULMetaDataCollectionNames.ForeignKeys )

// C#
```

```
DataTable schema =
  conn.GetSchema( ULMetaDataCollectionNames.ForeignKeys );
```

参照

- 「ULMetaDataCollectionNames クラス」 366 ページ
- 「ULMetaDataCollectionNames メンバ」 366 ページ
- 「GetSchema(String) メソッド」 174 ページ

IndexColumns プロパティ

IndexColumns コレクションを表す ULConnection.GetSchema(String) で使用する定数を提供します。

構文

Visual Basic
Public Shared Readonly Property **IndexColumns** As String

C#
public const string **IndexColumns** { get;}

プロパティ値

IndexColumns コレクションの名前を表す文字列。

例

次のコードは、IndexColumns コレクションを使用して DataTable を設定します。

```
' Visual Basic
Dim schema As DataTable = _
  conn.GetSchema( ULMetaDataCollectionNames.IndexColumns )

// C#
DataTable schema =
  conn.GetSchema( ULMetaDataCollectionNames.IndexColumns );
```

参照

- 「ULMetaDataCollectionNames クラス」 366 ページ
- 「ULMetaDataCollectionNames メンバ」 366 ページ
- 「GetSchema(String) メソッド」 174 ページ

Indexes プロパティ

Indexes コレクションを表す ULConnection.GetSchema(String) で使用する定数を提供します。

構文

Visual Basic
Public Shared Readonly Property **Indexes** As String

C#
public const string **Indexes** { get;}

プロパティ値

Indexes コレクションの名前を表す文字列。

例

次のコードは、Indexes コレクションを使用して DataTable を設定します。

```
' Visual Basic
Dim schema As DataTable = _
  conn.GetSchema( ULMetaDataCollectionNames.Indexes )

// C#
DataTable schema =
  conn.GetSchema( ULMetaDataCollectionNames.Indexes );
```

参照

- 「ULMetaDataCollectionNames クラス」 366 ページ
- 「ULMetaDataCollectionNames メンバ」 366 ページ
- 「GetSchema(String) メソッド」 174 ページ

MetaDataCollections プロパティ

MetaDataCollections コレクションを表す ULConnection.GetSchema(String) で使用する定数を提供します。

構文

Visual Basic
Public Shared Readonly Property **MetaDataCollections** As String

C#
public const string **MetaDataCollections** { get;}

プロパティ値

MetaDataCollections コレクションの名前を表す文字列。

例

次のコードは、MetaDataCollections コレクションを使用して DataTable を設定します。

```
' Visual Basic
Dim schema As DataTable = _
  conn.GetSchema( ULMetaDataCollectionNames.MetaDataCollections )

// C#
DataTable schema =
  conn.GetSchema( ULMetaDataCollectionNames.MetaDataCollections );
```

参照

- 「ULMetaDataCollectionNames クラス」 366 ページ
- 「ULMetaDataCollectionNames メンバ」 366 ページ
- 「GetSchema(String) メソッド」 174 ページ

Publications プロパティ

メタデータ・コレクションを取得する ULConnection.GetSchema(String,String[]) で使用する定数のリストを提供します。

構文

Visual Basic
Public Shared Readonly Property **Publications** As String

C#
public const string **Publications** { get;}

参照

- 「ULMetaDataCollectionNames クラス」 366 ページ
- 「ULMetaDataCollectionNames メンバ」 366 ページ
- 「GetSchema(String, String[]) メソッド」 174 ページ

ReservedWords プロパティ

ReservedWords コレクションを表す ULConnection.GetSchema(String) で使用する定数を提供します。

構文

Visual Basic
Public Shared Readonly Property **ReservedWords** As String

C#
public const string **ReservedWords** { get;}

プロパティ値

ReservedWords コレクションの名前を表す文字列。

例

次のコードは、ReservedWords コレクションを使用して DataTable を設定します。

```
' Visual Basic
Dim schema As DataTable = _
  conn.GetSchema( ULMetaDataCollectionNames.ReservedWords )

// C#
```



```
DataTable schema =
  conn.GetSchema( ULMetaDataCollectionNames.ReservedWords );
```

参照

- 「ULMetaDataCollectionNames クラス」 366 ページ
- 「ULMetaDataCollectionNames メンバ」 366 ページ
- 「GetSchema(String) メソッド」 174 ページ

Restrictions プロパティ

Restrictions コレクションを表す ULConnection.GetSchema(String) で使用する定数を提供します。

構文

Visual Basic
Public Shared Readonly Property **Restrictions** As String

C#
public const string **Restrictions** { get;}

プロパティ値

Restrictions コレクションの名前を表す文字列。

例

次のコードは、Restrictions コレクションを使用して DataTable を設定します。

```
' Visual Basic
Dim schema As DataTable = _
  conn.GetSchema( ULMetaDataCollectionNames.Restrictions )

// C#
DataTable schema =
  conn.GetSchema( ULMetaDataCollectionNames.Restrictions );
```

参照

- 「ULMetaDataCollectionNames クラス」 366 ページ
- 「ULMetaDataCollectionNames メンバ」 366 ページ
- 「GetSchema(String) メソッド」 174 ページ

Tables プロパティ

Tables コレクションを表す ULConnection.GetSchema(String) で使用する定数を提供します。

構文

Visual Basic
Public Shared Readonly Property **Tables** As String

C#
public const string **Tables** { get;}

プロパティ値

Tables コレクションの名前を表す文字列。

例

次のコードは、Tables コレクションを使用して DataTable を設定します。

```
' Visual Basic
Dim schema As DataTable = _
  conn.GetSchema( ULMetaDataCollectionNames.Tables )

// C#
DataTable schema =
  conn.GetSchema( ULMetaDataCollectionNames.Tables );
```

参照

- 「ULMetaDataCollectionNames クラス」 366 ページ
- 「ULMetaDataCollectionNames メンバ」 366 ページ
- 「GetSchema(String) メソッド」 174 ページ

ULParameter クラス

ULCommand のパラメータを表します。このクラスは継承できません。

構文

Visual Basic

```
Public NotInheritable Class ULParameter
  Inherits DbParameter
  Implements ICloneable
```

C#

```
public sealed class ULParameter: DbParameter,
  ICloneable
```

備考

ULParameter オブジェクトは、その多数のコンストラクタのいずれかを使用したり、ULCommand.CreateParameter メソッドを使用したりして直接作成できます。定数 0 および 0.0 の特別な処理と、オーバロードされたメソッドの解決方法のため、ULParameter(string,object) コンストラクタを使用するときは、定数値を型オブジェクトに明示的にキャストすることを強くおすすめします。次に例を示します。

```
' Visual Basic
Dim p As ULParameter = New ULParameter( "", CType( 0, Object ) )

// C#
ULParameter p = new ULParameter( "", (object)0 );
```

パラメータ (ULCommand.CreateParameter によって作成されたものを含む) は、使用される ULCommand.Parameters コレクションに追加する必要があります。すべてのパラメータは、位置パラメータとして扱われ、コマンドによって追加された順序で使用されます。

Ultra Light.NET では、パラメータは IN パラメータとしてのみ使用できるほか、マッピング情報はすべて無視されます。重要なのは、ULParameter.Value だけです。

継承：System.Data.Common.DbParameter

実装：System.Data.IDbDataParameter、System.Data.IDataParameter

参照

- 「ULParameter メンバ」 376 ページ
- 「ULCommand クラス」 91 ページ
- 「CreateParameter メソッド」 111 ページ
- 「ULParameter(String, Object) コンストラクタ」 378 ページ
- 「Parameters プロパティ」 103 ページ
- 「Value プロパティ」 390 ページ
- DbParameter
- IDbDataParameter
- IDataParameter

ULParameter メンバ

パブリック・コンストラクタ

メンバ名	説明
「ULParameter コンストラクタ」 377 ページ	「ULParameter クラス」 375 ページの新しいインスタンスを初期化します。

パブリック・プロパティ

メンバ名	説明
「DbType プロパティ」 383 ページ	パラメータの System.Data.DbType を指定します。
「Direction プロパティ」 384 ページ	パラメータが入力専用、出力専用、双方向性、またはストアド・プロシージャ戻り値パラメータのいずれであるかを示す値です。
「IsNullable プロパティ」 384 ページ	パラメータが NULL 値を受け入れるかどうかを指定します。
「Offset プロパティ」 385 ページ	ULParameter.Value のオフセットを指定します。
「ParameterName プロパティ」 386 ページ	パラメータの名前を指定します。
「Precision プロパティ」 386 ページ	ULParameter.Value を表すために使用される最大桁数を指定します。
「Scale プロパティ」 387 ページ	ULParameter.Value が解決される小数点の桁の数を指定します。
「Size プロパティ」 387 ページ	カラム内のデータの最大サイズを指定します。
「SourceColumn プロパティ」 388 ページ	DataSet にマッピングされ、値をロードしたり返したりするときに使用するソース・カラムの名前を指定します。
「SourceColumnNullMapping プロパティ」 388 ページ	
「SourceVersion プロパティ」 389 ページ	ULParameter.Value をロードするときに使用する System.Data.DataRowVersion を指定します。
「ULDbType プロパティ」 389 ページ	パラメータの iAnywhere.Data.UltraLite.ULDbType を指定します。

メンバ名	説明
「Value プロパティ」 390 ページ	パラメータの値を指定します。

パブリック・メソッド

メンバ名	説明
「ResetDbType メソッド」 390 ページ	このメソッドは Ultra Light.NET ではサポートされていません。
「ToString メソッド」 391 ページ	このインスタンスの文字列表現を返します。

参照

- 「ULParameter クラス」 375 ページ
- 「ULCommand クラス」 91 ページ
- 「CreateParameter メソッド」 111 ページ
- 「ULParameter(String, Object) コンストラクタ」 378 ページ
- 「Parameters プロパティ」 103 ページ
- 「Value プロパティ」 390 ページ
- DbParameter
- IDbDataParameter
- IDataParameter

ULParameter コンストラクタ

「ULParameter クラス」 375 ページの新しいインスタンスを初期化します。

ULParameter() コンストラクタ

値として NULL (Visual Basic の Nothing) を使用して、ULParameter オブジェクトを初期化します。

構文

Visual Basic
Public Sub **New()**

C#
public **ULParameter();**

例

次のコードでは、値が 3 である ULParameter パラメータが作成され、cmd という ULCommand が追加されます。

```
' Visual Basic
```

```
Dim p As ULParameter = New ULParameter
p.Value = 3
cmd.Parameters.Add( p )

// C#
ULParameter p = new ULParameter();
p.Value = 3;
cmd.Parameters.Add( p );
```

参照

- 「ULParameter クラス」 375 ページ
- 「ULParameter メンバ」 376 ページ
- 「ULParameter コンストラクタ」 377 ページ
- 「Value プロパティ」 390 ページ
- 「ULParameter(String, Object) コンストラクタ」 378 ページ
- 「ULCommand クラス」 91 ページ

ULParameter(String, Object) コンストラクタ

ULParameter オブジェクトを、指定されたパラメータ名と値で初期化します。

構文

Visual Basic

```
Public Sub New( _
  ByVal parameterName As String, _
  ByVal value As Object _
)
```

C#

```
public  ULParameter(
  string  parameterName,
  object value
);
```

パラメータ

- **parameterName**　パラメータの名前。名前のないパラメータの場合、この値に空の文字列 ("") または NULL 参照 (Visual Basic の Nothing) を使用します。Ultra Light.NET では、ULCommand はパラメータの名前を使用しません。
- **value**　パラメータの値である System.Object。

備考

定数 0 と 0.0 の特別な処理と、オーバロードされたメソッドの解決方法のため、このコンストラクタを使用するときは、定数値を型オブジェクトに明示的にキャストすることを強くおすすめします。

例

次のコードでは、値が 0 である ULParameter パラメータが作成され、cmd という ULCommand が追加されます。

```
' Visual Basic
cmd.Parameters.Add( New ULParameter( "", CType( 0, Object ) ) )

// C#
cmd.Parameters.Add( new ULParameter( "", (object)0 ) );
```

参照

- 「ULParameter クラス」 375 ページ
- 「ULParameter メンバ」 376 ページ
- 「ULParameter コンストラクタ」 377 ページ
- 「ULParameter() コンストラクタ」 377 ページ
- 「ULCommand クラス」 91 ページ
- オブジェクト

ULParameter(String, ULDbType) コンストラクタ

ULParameter オブジェクトを、指定されたパラメータ名とデータ型で初期化します。このコンストラクタの使用はおすすめしません。これは、他のデータ・プロバイダとの互換性のために用意されています。

構文

Visual Basic

```
Public Sub New( _
  ByVal parameterName As String, _
  ByVal dbType As ULDbType _
)
```

C#

```
public  ULParameter(
  string  parameterName,
  ULDbType dbType
);
```

パラメータ

- **parameterName**　パラメータの名前。名前のないパラメータの場合、この値に空の文字列 ("") または NULL 参照 (Visual Basic の Nothing) を使用します。Ultra Light.NET では、ULCommand はパラメータの名前を使用しません。
- **dbType**　iAnywhere.Data.UltraLite.ULDbType 値の 1 つ。

備考

Ultra Light.NET では、パラメータは IN パラメータとしてのみ使用できるほか、マッピング情報はすべて無視されます。重要なのは、ULParameter.Value だけです。

参照

- 「ULParameter クラス」 375 ページ
- 「ULParameter メンバ」 376 ページ
- 「ULParameter コンストラクタ」 377 ページ
- 「ULParameter() コンストラクタ」 377 ページ
- 「ULParameter(String, Object) コンストラクタ」 378 ページ
- 「Value プロパティ」 390 ページ
- 「ULCommand クラス」 91 ページ
- 「ULDbType 列挙体」 317 ページ

ULParameter(String, ULDbType, Int32) コンストラクタ

ULParameter オブジェクトを、指定されたパラメータ名とデータ型で初期化します。このコンストラクタの使用はおすすめしません。これは、他のデータ・プロバイダとの互換性のために用意されています。

構文

Visual Basic

```
Public Sub New( _
  ByVal parameterName As String, _
  ByVal dbType As ULDbType, _
  ByVal size As Integer _
)
```

C#

```
public  ULParameter(
  string  parameterName,
  ULDbType dbType,
  int size
);
```

パラメータ

- **parameterName**　パラメータの名前。名前のないパラメータの場合、この値に空の文字列 ("") または NULL 参照 (Visual Basic の Nothing) を使用します。Ultra Light.NET では、ULCommand はパラメータの名前を使用しません。
- **dbType**　iAnywhere.Data.UltraLite.ULDbType 値の 1 つ。
- **size**　パラメータの長さ。

備考

Ultra Light.NET では、パラメータは IN パラメータとしてのみ使用できるほか、マッピング情報はすべて無視されます。重要なのは、ULParameter.Value だけです。

参照

- 「ULParameter クラス」 375 ページ
- 「ULParameter メンバ」 376 ページ
- 「ULParameter コンストラクタ」 377 ページ
- 「ULParameter() コンストラクタ」 377 ページ
- 「ULParameter(String, Object) コンストラクタ」 378 ページ
- 「Value プロパティ」 390 ページ
- 「ULDbType 列挙体」 317 ページ

ULParameter(String, ULDbType, Int32, String) コンストラクタ

ULParameter オブジェクトを、指定されたパラメータ名、データ型、長さで初期化します。このコンストラクタの使用はおすすめしません。これは、他のデータ・プロバイダとの互換性のために用意されています。

構文

Visual Basic

```
Public Sub New( _
  ByVal parameterName As String, _
  ByVal dbType As ULDbType, _
  ByVal size As Integer, _
  ByVal sourceColumn As String _
)
```

C#

```
public  ULParameter(
  string  parameterName,
  ULDbType dbType,
  int size,
  string  sourceColumn
);
```

パラメータ

- **parameterName**　パラメータの名前。名前のないパラメータの場合、この値に空の文字列 ("") または NULL 参照 (Visual Basic の Nothing) を使用します。Ultra Light.NET では、ULCommand はパラメータの名前を使用しません。
- **dbType**　iAnywhere.Data.UltraLite.ULDbType 値の 1 つ。
- **size**　パラメータの長さ。
- **sourceColumn**　マッピングするソース・カラムの名前。

備考

Ultra Light.NET では、パラメータは IN パラメータとしてのみ使用できるほか、マッピング情報はすべて無視されます。重要なのは、ULParameter.Value だけです。

参照

- 「ULParameter クラス」 375 ページ
- 「ULParameter メンバ」 376 ページ
- 「ULParameter コンストラクタ」 377 ページ
- 「ULParameter() コンストラクタ」 377 ページ
- 「ULParameter(String, Object) コンストラクタ」 378 ページ
- 「Value プロパティ」 390 ページ
- 「ULDbType 列挙体」 317 ページ
- 「ULCommand クラス」 91 ページ

ULParameter(String, ULDbType, Int32, ParameterDirection, Boolean, Byte, Byte, String, DataRowVersion, Object) コンストラクタ

指定されたパラメータ名、データ型、長さ、方向、NULL 入力属性、数値精度、数値の位取り、ソース・カラム、ソースのバージョン、値で、ULParameter オブジェクトを初期化します。このコンストラクタの使用はおすすめしません。これは、他のデータ・プロバイダとの互換性のために用意されています。

構文

Visual Basic

```
Public Sub New( _
  ByVal parameterName As String, _
  ByVal dbType As ULDbType, _
  ByVal size As Integer, _
  ByVal direction As ParameterDirection, _
  ByVal isNullable As Boolean, _
  ByVal precision As Byte, _
  ByVal scale As Byte, _
  ByVal sourceColumn As String, _
  ByVal sourceVersion As DataRowVersion, _
  ByVal value As Object _
)
```

C#

```
public  ULParameter(
  string  parameterName,
  ULDbType dbType,
  int size,
  ParameterDirection direction,
  bool isNullable,
  byte precision,
  byte scale,
  string  sourceColumn,
  DataRowVersion sourceVersion,
  object value
);
```


パラメータ

- **parameterName**　パラメータの名前。名前のないパラメータの場合、この値に空の文字列 ("") または NULL 参照 (Visual Basic の Nothing) を使用します。Ultra Light.NET では、ULCommand はパラメータの名前を使用しません。
- **dbType**　iAnywhere.Data.UltraLite.ULDbType 値の 1 つ。
- **size**　パラメータの長さ。
- **direction**　System.Data.ParameterDirection 値の 1 つ。
- **isNullable**　フィールドの値を NULL にできる場合は true、できない場合は false。
- **precision**　Value が解決される小数点の左右の桁の合計数。
- **scale**　Value が解決される小数点までの桁の合計数。
- **sourceColumn**　マッピングするソース・カラムの名前。
- **sourceVersion**　System.Data.DataRowVersion 値の 1 つ。
- **value**　パラメータの値である System.Object。

参照

- 「ULParameter クラス」 375 ページ
- 「ULParameter メンバ」 376 ページ
- 「ULParameter コンストラクタ」 377 ページ
- 「ULParameter() コンストラクタ」 377 ページ
- 「ULParameter(String, Object) コンストラクタ」 378 ページ
- 「ULDbType 列挙体」 317 ページ
- ParameterDirection
- DataRowVersion
- オブジェクト
- ParameterDirection.Input
- 「ULCommand クラス」 91 ページ

DbType プロパティ

パラメータの System.Data.DbType を指定します。

構文

Visual Basic
Public Overrides Property **DbType** As DbType

C#
public override DbType **DbType** { get; set; }

プロパティ値

System.Data.DbType 値の 1 つ。

備考

ULParameter.ULDbType と DbType プロパティはリンクされます。このため、DbType を設定すると、ULParameter.ULDbType がサポートされている iAnywhere.Data.UltraLite.ULDbType に変更されます。

参照

- 「ULParameter クラス」 375 ページ
- 「ULParameter メンバ」 376 ページ
- DbType
- DbType
- 「ULDbType プロパティ」 389 ページ
- 「ULDbType 列挙体」 317 ページ

Direction プロパティ

パラメータが入力専用、出力専用、双方向性、またはストアド・プロシージャ戻り値パラメータのいずれであるかを示す値です。

構文

Visual Basic
Public Overrides Property **Direction** As ParameterDirection

C#
public override ParameterDirection **Direction** { get; set; }

プロパティ値

System.Data.ParameterDirection 値の 1 つ。

備考

Ultra Light.NET では、パラメータは IN パラメータとしてのみ使用できるほか、マッピング情報はすべて無視されます。重要なのは、ULParameter.Value だけです。

参照

- 「ULParameter クラス」 375 ページ
- 「ULParameter メンバ」 376 ページ
- ParameterDirection
- ParameterDirection.Input
- 「Value プロパティ」 390 ページ

IsNullable プロパティ

パラメータが NULL 値を受け入れるかどうかを指定します。

構文

Visual Basic
Public Overrides Property **IsNullable** As Boolean

C#
public override bool **IsNullable** { get; set; }

プロパティ値

NULL 値を受け入れる場合は true、受け入れない場合は false。デフォルトは false です。NULL 値は DBNull クラスを使用して処理されます。

備考

Ultra Light.NET では、パラメータは IN パラメータとしてのみ使用できるほか、マッピング情報はすべて無視されます。重要なのは、ULParameter.Value だけです。

参照

- 「ULParameter クラス」 375 ページ
- 「ULParameter メンバ」 376 ページ
- 「Value プロパティ」 390 ページ

Offset プロパティ

ULParameter.Value のオフセットを指定します。

構文

Visual Basic
Public Property **Offset** As Integer

C#
public int **Offset** { get; set; }

プロパティ値

値のオフセット。デフォルトは 0 です。

備考

Ultra Light.NET では、パラメータは IN パラメータとしてのみ使用できるほか、マッピング情報はすべて無視されます。重要なのは、ULParameter.Value だけです。

参照

- 「ULParameter クラス」 375 ページ
- 「ULParameter メンバ」 376 ページ
- 「Value プロパティ」 390 ページ

ParameterName プロパティ

パラメータの名前を指定します。

構文

Visual Basic
Public Overrides Property **ParameterName** As String

C#
public override string **ParameterName** { get; set; }

プロパティ値

パラメータの名前を表す文字列。名前のないパラメータの場合は空の文字列 ("")。NULL 参照 (Visual Basic の Nothing) を指定すると、空の文字列が使用されます。

備考

Ultra Light.NET では、ULCommand はパラメータの名前を使用しません。すべてのパラメータは、位置パラメータとして扱われ、コマンドによって追加された順序で使用されます。

参照

- 「ULParameter クラス」 375 ページ
- 「ULParameter メンバ」 376 ページ
- 「ULCommand クラス」 91 ページ

Precision プロパティ

ULParameter.Value を表すために使用される最大桁数を指定します。

構文

Visual Basic
Public Property **Precision** As Byte

C#
public byte **Precision** { get; set; }

プロパティ値

ULParameter.Value を表すために使用される最大桁数。デフォルト値は 0 です。これは、データ・プロバイダが ULParameter.Value の精度を設定することを示します。

備考

Ultra Light.NET では、パラメータは IN パラメータとしてのみ使用できるほか、マッピング情報はすべて無視されます。重要なのは、ULParameter.Value だけです。

参照

- 「ULParameter クラス」 375 ページ
- 「ULParameter メンバ」 376 ページ
- 「Value プロパティ」 390 ページ

Scale プロパティ

ULParameter.Value が解決される小数点の桁の数を指定します。

構文

Visual Basic
Public Property **Scale** As Byte

C#
public byte **Scale** { get; set; }

プロパティ値

ULParameter.Value が解決される小数点の桁の数。デフォルトは 0 です。

備考

Ultra Light.NET では、パラメータは IN パラメータとしてのみ使用できるほか、マッピング情報はすべて無視されます。重要なのは、ULParameter.Value だけです。

参照

- 「ULParameter クラス」 375 ページ
- 「ULParameter メンバ」 376 ページ
- 「Value プロパティ」 390 ページ

Size プロパティ

カラム内のデータの最大サイズを指定します。

構文

Visual Basic
Public Overrides Property **Size** As Integer

C#
public override int **Size** { get; set; }

プロパティ値

カラム内のデータの最大サイズ。デフォルト値はパラメータ値から推測されます。Size プロパティは、バイナリと文字列型に対して使用されます。

備考

Ultra Light.NET では、パラメータは IN パラメータとしてのみ使用できるほか、マッピング情報はすべて無視されます。重要なのは、ULParameter.Value だけです。

参照

- 「ULParameter クラス」 375 ページ
- 「ULParameter メンバ」 376 ページ
- 「Value プロパティ」 390 ページ

SourceColumn プロパティ

DataSet にマッピングされ、値をロードしたり返したりするときに使用するソース・カラムの名前を指定します。

構文

Visual Basic
Public Overrides Property **SourceColumn** As String

C#
public override string **SourceColumn** { get; set; }

プロパティ値

DataSet にマッピングされ、値をロードしたり返したりするときに使用するソース・カラムの名前を指定する文字列。

備考

Ultra Light.NET では、パラメータは IN パラメータとしてのみ使用できるほか、マッピング情報はすべて無視されます。重要なのは、ULParameter.Value だけです。

参照

- 「ULParameter クラス」 375 ページ
- 「ULParameter メンバ」 376 ページ
- 「Value プロパティ」 390 ページ

SourceColumnNullMapping プロパティ

構文

Visual Basic
Public Overrides Property **SourceColumnNullMapping** As Boolean

C#
public override bool **SourceColumnNullMapping** { get; set; }

参照

- 「ULParameter クラス」 375 ページ
- 「ULParameter メンバ」 376 ページ

SourceVersion プロパティ

ULParameter.Value をロードするときに使用する System.Data.DataRowVersion を指定します。

構文

Visual Basic
Public Overrides Property **SourceVersion** As DataRowVersion

C#
public override DataRowVersion **SourceVersion** { get; set; }

参照

- 「ULParameter クラス」 375 ページ
- 「ULParameter メンバ」 376 ページ
- 「Value プロパティ」 390 ページ
- DataRowVersion

ULDbType プロパティ

パラメータの iAnywhere.Data.UltraLite.ULDbType を指定します。

構文

Visual Basic
Public Property **ULDbType** As ULDbType

C#
public ULDbType **ULDbType** { get; set; }

プロパティ値

iAnywhere.Data.UltraLite.ULDbType 値の 1 つ。

備考

ULDbType と ULParameter.DbType はリンクされます。このため、ULDbType を設定すると、ULParameter.DbType がサポートされている System.Data.DbType を変更します。

Ultra Light.NET では、パラメータは IN パラメータとしてのみ使用できるほか、マッピング情報はすべて無視されます。重要なのは、ULParameter.Value だけです。

参照

- 「ULParameter クラス」 375 ページ
- 「ULParameter メンバ」 376 ページ
- 「Value プロパティ」 390 ページ
- 「DbType プロパティ」 383 ページ
- DbType
- 「ULDbType 列挙体」 317 ページ

Value プロパティ

パラメータの値を指定します。

構文

Visual Basic
Public Overrides Property **Value** As Object

C#
public override object **Value** { get; set; }

プロパティ値

パラメータの値を指定する System.Object。

備考

値は、型変換やマッピングは行われずに、データ・プロバイダにそのまま送信されます。コマンドが実行されると、コマンドは必要な型に値を変換しますが、値を変換できない場合は ULSQLCode.SQLE_CONVERSION_ERROR とともに ULException を通知します。

参照

- 「ULParameter クラス」 375 ページ
- 「ULParameter メンバ」 376 ページ
- 「ULException クラス」 322 ページ
- 「ULSQLCode 列挙体」 462 ページ
- オブジェクト

ResetDbType メソッド

このメソッドは Ultra Light.NET ではサポートされていません。

構文

Visual Basic
Public Overrides Sub **ResetDbType()**

C#
public override void **ResetDbType();**

備考

Ultra Light.NET では、パラメータは IN パラメータとしてのみ使用できるほか、マッピング情報はすべて無視されます。重要なのは、ULParameter.Value だけです。

参照

- 「ULParameter クラス」 375 ページ
- 「ULParameter メンバ」 376 ページ
- 「Value プロパティ」 390 ページ

ToString メソッド

このインスタンスの文字列表現を返します。

構文

Visual Basic
Public Overrides Function **ToString()** As String

C#
public override string **ToString();**

戻り値

パラメータの名前。

参照

- 「ULParameter クラス」 375 ページ
- 「ULParameter メンバ」 376 ページ

ULParameterCollection クラス

ULCommand のすべてのパラメータを表します。このクラスは継承できません。

構文

Visual Basic
Public NotInheritable Class **ULParameterCollection**
  Inherits DbParameterCollection

C#
public sealed class **ULParameterCollection**: DbParameterCollection

備考

コレクション内のすべてのパラメータは、位置パラメータとして扱われ、ULCommand.CommandText の疑問符のプレースホルダと同じ順序で指定されます。たとえば、コレクション内の最初のパラメータは、SQL 文の最初の疑問符に対応し、コレクション内の 2 番目のパラメータは、SQL 文の 2 番目の疑問符に対応します。ULCommand.CommandText 内の疑問符の数は、少なくともコレクション内のパラメータの数と同じでなければなりません。パラメータが不足している場合は、NULL が代用されます。

ULParameterCollection にはコンストラクタがありません。ULParameterCollection は、ULCommand.Parameters から取得します。

継承：System.Data.Common.DbParameterCollection

実装：System.Data.IDataParameterCollection

参照

- 「ULParameterCollection メンバ」 392 ページ
- 「ULCommand クラス」 91 ページ
- 「CommandText プロパティ」 99 ページ
- 「Parameters プロパティ」 103 ページ
- DbParameterCollection
- IDataParameterCollection

ULParameterCollection メンバ

パブリック・プロパティ

メンバ名	説明
「Count プロパティ」 394 ページ	コレクション内の ULParameter オブジェクトの数を返します。
「IsFixedSize プロパティ」 394 ページ	ULParameterCollection のサイズが固定かどうかを示します。

メンバ名	説明
「IsReadOnly プロパティ」 395 ページ	ULParameterCollection が読み取り専用であるかどうかを示します。
「IsSynchronized プロパティ」 395 ページ	ULParameterCollection が同期されるかどうかを示します。
「Item プロパティ」 395 ページ	コレクション内の DbParameter を取得または設定します。
「SyncRoot プロパティ」 397 ページ	SAParameterCollection へのアクセスを同期するために使用するオブジェクトを返します。

パブリック・メソッド

メンバ名	説明
「Add メソッド」 397 ページ	ULParameter をコレクションに追加します。
「AddRange メソッド」 403 ページ	ULParameterCollection の末尾に値の配列を追加します。
「Clear メソッド」 405 ページ	すべてのパラメータをコレクションから削除します。
「Contains メソッド」 405 ページ	特定のプロパティが設定された DbParameter がコレクション内に存在するかどうかを示します。
「CopyTo メソッド」 406 ページ	ULParameter オブジェクトを ULParameterCollection から指定された配列にコピーします。
「GetEnumerator メソッド」 407 ページ	コレクションの列挙子を返します。
「IndexOf メソッド」 407 ページ	指定された DbParameter オブジェクトのインデックスを返します。
「Insert メソッド」 409 ページ	コレクション内の指定されたインデックス位置に ULParameter を挿入します。
「Remove メソッド」 409 ページ	ULParameter をコレクションから削除します。
「RemoveAt メソッド」 410 ページ	指定された DbParameter オブジェクトをコレクションから削除します。

参照

- 「ULParameterCollection クラス」 392 ページ
- 「ULCommand クラス」 91 ページ
- 「CommandText プロパティ」 99 ページ
- 「Parameters プロパティ」 103 ページ
- DbParameterCollection
- IDataParameterCollection

Count プロパティ

コレクション内の ULParameter オブジェクトの数を返します。

構文

Visual Basic
Public Overrides Readonly Property **Count** As Integer

C#
public override int **Count** { get;}

プロパティ値

コレクション内の ULParameter オブジェクトの数。

参照

- 「ULParameterCollection クラス」 392 ページ
- 「ULParameterCollection メンバ」 392 ページ

IsFixedSize プロパティ

ULParameterCollection のサイズが固定かどうかを示します。

構文

Visual Basic
Public Overrides Readonly Property **IsFixedSize** As Boolean

C#
public override bool **IsFixedSize** { get;}

プロパティ値

コレクションのサイズが固定の場合は true、そうでない場合は false。

参照

- 「ULParameterCollection クラス」 392 ページ
- 「ULParameterCollection メンバ」 392 ページ

IsReadOnly プロパティ

ULParameterCollection が読み取り専用であるかどうかを示します。

構文

Visual Basic
Public Overrides Readonly Property **IsReadOnly** As Boolean

C#
public override bool **IsReadOnly** { get;}

プロパティ値

コレクションが読み込み専用の場合は true、そうでない場合は false。

参照

- 「ULParameterCollection クラス」 392 ページ
- 「ULParameterCollection メンバ」 392 ページ

IsSynchronized プロパティ

ULParameterCollection が同期されるかどうかを示します。

構文

Visual Basic
Public Overrides Readonly Property **IsSynchronized** As Boolean

C#
public override bool **IsSynchronized** { get;}

プロパティ値

コレクションが同期している場合は true、そうでない場合は false。

参照

- 「ULParameterCollection クラス」 392 ページ
- 「ULParameterCollection メンバ」 392 ページ

Item プロパティ

コレクション内の DbParameter を取得または設定します。

Item(Int32) プロパティ

指定されたインデックスの ULParameter を返します。C# では、このプロパティは ULParameterCollection クラスのインデクサです。

構文

Visual Basic
```
Public Property Item( _
  ByVal index As Integer _
) As ULParameter
```

C#
```
public ULParameter this[
  int index
] { get; set; }
```

パラメータ

- **index**　取り出すパラメータの 0 から始まるインデックス。値は、[0,ULParameterCollection.Count-1] の範囲内であることが必要です。コレクションの最初のパラメータのインデックス値は 0 です。

プロパティ値

指定されたインデックス位置の ULParameter。

備考

これは、DbParameterCollection.this[int] が厳密に型指定されたものです。

参照

- 「ULParameterCollection クラス」 392 ページ
- 「ULParameterCollection メンバ」 392 ページ
- 「Item プロパティ」 395 ページ
- 「ULParameter クラス」 375 ページ
- DbParameterCollection.Item

Item(String) プロパティ

指定された名前の ULParameter を返します。C# では、このプロパティは ULParameterCollection クラスのインデクサです。

構文

Visual Basic
```
Public Property Item( _
  ByVal parameterName As String _
) As ULParameter
```

C#
```
public ULParameter this[
  string  parameterName
] { get; set; }
```

パラメータ

- **parameterName**　取り出すパラメータの名前。

プロパティ値

指定された名前の ULParameter。

備考

これは、DbParameterCollection.this[string] が厳密に型指定されたものです。

参照

- 「ULParameterCollection クラス」 392 ページ
- 「ULParameterCollection メンバ」 392 ページ
- 「Item プロパティ」 395 ページ
- 「Item(Int32) プロパティ」 283 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「GetValue メソッド」 308 ページ
- 「GetFieldType メソッド」 296 ページ
- 「ULParameter クラス」 375 ページ

SyncRoot プロパティ

SAParameterCollection へのアクセスを同期するために使用するオブジェクトを返します。

構文

Visual Basic
Public Overrides Readonly Property **SyncRoot** As Object

C#
public override object **SyncRoot** { get;}

プロパティ値

このコレクションへのアクセスの同期に使用されるオブジェクト。

参照

- 「ULParameterCollection クラス」 392 ページ
- 「ULParameterCollection メンバ」 392 ページ

Add メソッド

ULParameter をコレクションに追加します。

Add(Object) メソッド

ULParameter をコレクションに追加します。

構文

Visual Basic
```
Public Overrides Function Add( _
  ByVal value As Object _
) As Integer
```

C#
```
public override int Add(
  object value
);
```

パラメータ

- **value**　コレクションに追加される ULParameter オブジェクト。

戻り値

新しい ULParameter オブジェクトのインデックス。

備考

コレクション内のすべてのパラメータは、位置パラメータとして扱われ、ULCommand.CommandText の疑問符のプレースホルダと同じ順序で指定する必要があります。たとえば、コレクション内の最初のパラメータは、SQL 文の最初の疑問符に対応し、コレクション内の 2 番目のパラメータは、SQL 文の 2 番目の疑問符に対応します。ULCommand.CommandText 内の疑問符の数は、少なくともコレクション内のパラメータの数と同じでなければなりません。パラメータが不足している場合は、NULL が代用されます。

参照

- 「ULParameterCollection クラス」 392 ページ
- 「ULParameterCollection メンバ」 392 ページ
- 「Add メソッド」 397 ページ
- 「Add(ULParameter) メソッド」 398 ページ
- 「Add(String, Object) メソッド」 399 ページ
- 「CommandText プロパティ」 99 ページ
- 「ULParameter クラス」 375 ページ

Add(ULParameter) メソッド

ULParameter をコレクションに追加します。

構文

Visual Basic
```
Public Function Add( _
  ByVal value As ULParameter _
) As ULParameter
```

C#
```
public ULParameter Add(
```

```
  ULParameter value
);
```

パラメータ

- **value**　コレクションに追加される ULParameter オブジェクト。

戻り値

新しい ULParameter オブジェクト。

備考

コレクション内のすべてのパラメータは、位置パラメータとして扱われ、ULCommand.CommandText の疑問符のプレースホルダと同じ順序で指定する必要があります。たとえば、コレクション内の最初のパラメータは、SQL 文の最初の疑問符に対応し、コレクション内の 2 番目のパラメータは、SQL 文の 2 番目の疑問符に対応します。ULCommand.CommandText 内の疑問符の数は、少なくともコレクション内のパラメータの数と同じでなければなりません。パラメータが不足している場合は、NULL が代用されます。

参照

- 「ULParameterCollection クラス」 392 ページ
- 「ULParameterCollection メンバ」 392 ページ
- 「Add メソッド」 397 ページ
- 「Add(String, Object) メソッド」 399 ページ
- 「ULParameter クラス」 375 ページ
- 「CommandText プロパティ」 99 ページ

Add(String, Object) メソッド

指定されたパラメータ名と値を使用して作成された新しい ULParameter をコレクションに追加します。

構文

Visual Basic

```
Public Function Add( _
  ByVal parameterName As String, _
  ByVal value As Object _
) As ULParameter
```

C#

```
public ULParameter Add(
  string  parameterName,
  object value
);
```

パラメータ

- **parameterName**　パラメータの名前。名前のないパラメータの場合、この値に空の文字列 ("") または NULL 参照 (Visual Basic の Nothing) を使用します。Ultra Light.NET では、ULCommand はパラメータの名前を使用しません。

- **value**　パラメータの値である System.Object。

戻り値

新しい ULParameter オブジェクト。

備考

コレクション内のすべてのパラメータは、位置パラメータとして扱われ、ULCommand.CommandText の疑問符のプレースホルダと同じ順序で指定する必要があります。たとえば、コレクション内の最初のパラメータは、SQL 文の最初の疑問符に対応し、コレクション内の 2 番目のパラメータは、SQL 文の 2 番目の疑問符に対応します。ULCommand.CommandText 内の疑問符の数は、少なくともコレクション内のパラメータの数と同じでなければなりません。パラメータが不足している場合は、NULL が代用されます。

定数 0 と 0.0 の特別な処理と、オーバロードされたメソッドの解決方法のため、このメソッドを使用するときは、定数値を型オブジェクトに明示的にキャストすることを強くおすすめします。

例

次のコードでは、値が 0 である ULParameter パラメータが cmd という ULCommand に追加されます。

```
' Visual Basic
cmd.Parameters.Add( "", CType( 0, Object ) )

// C#
cmd.Parameters.Add( "", (object)0 );
```

参照

- 「ULParameterCollection クラス」 392 ページ
- 「ULParameterCollection メンバ」 392 ページ
- 「Add メソッド」 397 ページ
- 「Add(ULParameter) メソッド」 398 ページ
- 「ULParameter クラス」 375 ページ
- 「CommandText プロパティ」 99 ページ
- 「ULCommand クラス」 91 ページ
- オブジェクト

Add(String, ULDbType) メソッド

指定されたパラメータ名とデータ型を使用して作成された新しい ULParameter をコレクションに追加します。

構文

Visual Basic

```
Public Function Add( _
   ByVal parameterName As String, _
   ByVal ulDbType As ULDbType _
) As ULParameter
```

C#
```
public ULParameter Add(
  string parameterName,
  ULDbType ulDbType
);
```

パラメータ

- **parameterName**　パラメータの名前。名前のないパラメータの場合、この値に空の文字列 ("") または NULL 参照 (Visual Basic の Nothing) を使用します。Ultra Light.NET では、ULCommand はパラメータの名前を使用しません。
- **ulDbType**　iAnywhere.Data.UltraLite.ULDbType 値の 1 つ。

戻り値

新しい ULParameter オブジェクト。

備考

コレクション内のすべてのパラメータは、位置パラメータとして扱われ、ULCommand.CommandText の疑問符のプレースホルダと同じ順序で指定する必要があります。たとえば、コレクション内の最初のパラメータは、SQL 文の最初の疑問符に対応し、コレクション内の 2 番目のパラメータは、SQL 文の 2 番目の疑問符に対応します。ULCommand.CommandText 内の疑問符の数は、少なくともコレクション内のパラメータの数と同じでなければなりません。パラメータが不足している場合は、NULL が代用されます。

参照

- 「ULParameterCollection クラス」 392 ページ
- 「ULParameterCollection メンバ」 392 ページ
- 「Add メソッド」 397 ページ
- 「Add(ULParameter) メソッド」 398 ページ
- 「Add(String, Object) メソッド」 399 ページ
- 「ULParameter クラス」 375 ページ
- 「CommandText プロパティ」 99 ページ
- 「ULCommand クラス」 91 ページ
- 「ULDbType 列挙体」 317 ページ

Add(String, ULDbType, Int32) メソッド

指定されたパラメータ名、データ型、長さを使用して作成された新しい ULParameter をコレクションに追加します。

構文

Visual Basic
```
Public Function Add( _
  ByVal parameterName As String, _
  ByVal ulDbType As ULDbType, _
  ByVal size As Integer _
) As ULParameter
```

C#
```
public ULParameter Add(
  string parameterName,
  ULDbType ulDbType,
  int size
);
```

パラメータ

- **parameterName** パラメータの名前。名前のないパラメータの場合、この値に空の文字列 ("") または NULL 参照 (Visual Basic の Nothing) を使用します。Ultra Light.NET では、ULCommand はパラメータの名前を使用しません。
- **ulDbType** iAnywhere.Data.UltraLite.ULDbType 値の 1 つ。
- **size** パラメータの長さ。

戻り値

新しい ULParameter オブジェクト。

備考

コレクション内のすべてのパラメータは、位置パラメータとして扱われ、ULCommand.CommandText の疑問符のプレースホルダと同じ順序で指定する必要があります。たとえば、コレクション内の最初のパラメータは、SQL 文の最初の疑問符に対応し、コレクション内の 2 番目のパラメータは、SQL 文の 2 番目の疑問符に対応します。ULCommand.CommandText 内の疑問符の数は、少なくともコレクション内のパラメータの数と同じでなければなりません。パラメータが不足している場合は、NULL が代用されます。

参照

- 「ULParameterCollection クラス」 392 ページ
- 「ULParameterCollection メンバ」 392 ページ
- 「Add メソッド」 397 ページ
- 「Add(ULParameter) メソッド」 398 ページ
- 「Add(String, Object) メソッド」 399 ページ
- 「ULParameter クラス」 375 ページ
- 「CommandText プロパティ」 99 ページ
- 「ULCommand クラス」 91 ページ
- 「ULDbType 列挙体」 317 ページ

Add(String, ULDbType, Int32, String) メソッド

指定されたパラメータ名、データ型、長さ、ソース・カラム名を使用して作成された新しい ULParameter をコレクションに追加します。

構文

Visual Basic
```
Public Function Add( _
  ByVal parameterName As String, _
```



```
  ByVal ulDbType As ULDbType, _
  ByVal size As Integer, _
  ByVal sourceColumn As String _
) As ULParameter

C#
public ULParameter Add(
  string  parameterName,
  ULDbType ulDbType,
  int size,
  string  sourceColumn
);
```

パラメータ

- **parameterName**　パラメータの名前。名前のないパラメータの場合、この値に空の文字列 ("") または NULL 参照 (Visual Basic の Nothing) を使用します。Ultra Light.NET では、ULCommand はパラメータの名前を使用しません。
- **ulDbType**　iAnywhere.Data.UltraLite.ULDbType 値の 1 つ。
- **size**　パラメータの長さ。
- **sourceColumn**　マッピングするソース・カラムの名前。

戻り値

新しい ULParameter オブジェクト。

備考

コレクション内のすべてのパラメータは、位置パラメータとして扱われ、ULCommand.CommandText の疑問符のプレースホルダと同じ順序で指定する必要があります。たとえば、コレクション内の最初のパラメータは、SQL 文の最初の疑問符に対応し、コレクション内の 2 番目のパラメータは、SQL 文の 2 番目の疑問符に対応します。ULCommand.CommandText 内の疑問符の数は、少なくともコレクション内のパラメータの数と同じでなければなりません。パラメータが不足している場合は、NULL が代用されます。

参照

- 「ULParameterCollection クラス」 392 ページ
- 「ULParameterCollection メンバ」 392 ページ
- 「Add メソッド」 397 ページ
- 「Add(ULParameter) メソッド」 398 ページ
- 「Add(String, Object) メソッド」 399 ページ
- 「ULParameter クラス」 375 ページ
- 「CommandText プロパティ」 99 ページ
- 「ULCommand クラス」 91 ページ
- 「ULDbType 列挙体」 317 ページ

AddRange メソッド

ULParameterCollection の末尾に値の配列を追加します。

AddRange(Array) メソッド

ULParameterCollection の末尾に値の配列を追加します。

構文

Visual Basic
```
Public Overrides Sub AddRange( _
  ByVal values As Array _
)
```

C#
```
public override void AddRange(
  Array values
);
```

パラメータ

- **values** このコレクションの末尾に追加する ULParameter オブジェクトの配列。

参照

- 「ULParameterCollection クラス」 392 ページ
- 「ULParameterCollection メンバ」 392 ページ
- 「AddRange メソッド」 403 ページ
- 「ULParameter クラス」 375 ページ

AddRange(ULParameter[]) メソッド

ULParameterCollection の末尾に値の配列を追加します。

構文

Visual Basic
```
Public Sub AddRange( _
  ByVal values As ULParameter() _
)
```

C#
```
public void AddRange(
  ULParameter[] values
);
```

パラメータ

- **values** このコレクションの末尾に追加する ULParameter オブジェクトの配列。

備考

これは、DbParameterCollection.AddRange(Array) が厳密に型指定されたものです。

参照

- 「ULParameterCollection クラス」 392 ページ
- 「ULParameterCollection メンバ」 392 ページ
- 「AddRange メソッド」 403 ページ
- 「ULParameter クラス」 375 ページ
- DbParameterCollection.AddRange
- 「ULParameterCollection クラス」 392 ページ

Clear メソッド

すべてのパラメータをコレクションから削除します。

構文

Visual Basic
Public Overrides Sub **Clear()**

C#
public override void **Clear();**

参照

- 「ULParameterCollection クラス」 392 ページ
- 「ULParameterCollection メンバ」 392 ページ

Contains メソッド

特定のプロパティが設定された DbParameter がコレクション内に存在するかどうかを示します。

Contains(Object) メソッド

ULParameter がコレクション内に存在するかどうかをチェックします。

構文

Visual Basic
```
Public Overrides Function Contains( _
  ByVal value As Object _
) As Boolean
```

C#
```
public override bool Contains(
  object value
);
```

パラメータ

- **value**　チェック対象の ULParameter オブジェクト。

戻り値

コレクションに、その ULParameter がある場合は true、ない場合は false。

参照

- 「ULParameterCollection クラス」 392 ページ
- 「ULParameterCollection メンバ」 392 ページ
- 「Contains メソッド」 405 ページ
- 「Contains(String) メソッド」 406 ページ
- 「ULParameter クラス」 375 ページ

Contains(String) メソッド

指定された名前の ULParameter がコレクション内に存在するかどうかをチェックします。

構文

Visual Basic

```
Public Overrides Function Contains( _
  ByVal value As String _
) As Boolean
```

C#

```
public override bool Contains(
  string  value
);
```

パラメータ

- **value**　検索対象のパラメータの名前。

戻り値

コレクションに、その ULParameter がある場合は true、ない場合は false。

参照

- 「ULParameterCollection クラス」 392 ページ
- 「ULParameterCollection メンバ」 392 ページ
- 「Contains メソッド」 405 ページ
- 「Contains(Object) メソッド」 405 ページ
- 「ULParameter クラス」 375 ページ

CopyTo メソッド

ULParameter オブジェクトを ULParameterCollection から指定された配列にコピーします。

構文

Visual Basic

```
Public Overrides Sub CopyTo( _
```

```
  ByVal array As Array, _
  ByVal index As Integer _
)

C#
public override void CopyTo(
  Array array,
  int index
);
```

パラメータ

- **array**　ULParameter オブジェクトのコピー先の配列。
- **index**　配列の開始インデックス。

参照

- 「ULParameterCollection クラス」 392 ページ
- 「ULParameterCollection メンバ」 392 ページ
- 「ULParameter クラス」 375 ページ

GetEnumerator メソッド

コレクションの列挙子を返します。

構文

Visual Basic
Public Overrides Function **GetEnumerator()** As IEnumerator

C#
public override IEnumerator **GetEnumerator();**

戻り値

コレクション内のパラメータを列挙する ArrayList 列挙子。

参照

- 「ULParameterCollection クラス」 392 ページ
- 「ULParameterCollection メンバ」 392 ページ

IndexOf メソッド

指定された DbParameter オブジェクトのインデックスを返します。

IndexOf(Object) メソッド

コレクション内の ULParameter のロケーションを返します。

構文

Visual Basic
```
Public Overrides Function IndexOf( _
  ByVal value As Object _
) As Integer
```

C#
```
public override int IndexOf(
  object value
);
```

パラメータ

- **value**　検索する ULParameter オブジェクト。

戻り値

コレクション内の ULParameter の 0 から始まるインデックス。パラメータが見つからない場合は -1。

参照

- 「ULParameterCollection クラス」 392 ページ
- 「ULParameterCollection メンバ」 392 ページ
- 「IndexOf メソッド」 407 ページ
- 「IndexOf(String) メソッド」 408 ページ
- 「ULParameter クラス」 375 ページ

IndexOf(String) メソッド

指定された名前を持つ ULParameter のコレクション内のロケーションを返します。

構文

Visual Basic
```
Public Overrides Function IndexOf( _
  ByVal parameterName As String _
) As Integer
```

C#
```
public override int IndexOf(
  string  parameterName
);
```

パラメータ

- **parameterName**　検索するパラメータの名前。

戻り値

コレクション内の ULParameter の 0 から始まるインデックス。パラメータが見つからない場合は -1。

参照

- 「ULParameterCollection クラス」 392 ページ
- 「ULParameterCollection メンバ」 392 ページ
- 「IndexOf メソッド」 407 ページ
- 「IndexOf(Object) メソッド」 407 ページ
- 「ULParameter クラス」 375 ページ

Insert メソッド

コレクション内の指定されたインデックス位置に ULParameter を挿入します。

構文

Visual Basic

```
Public Overrides Sub Insert( _
  ByVal index As Integer, _
  ByVal value As Object _
)
```

C#

```
public override void Insert(
  int index,
  object value
);
```

パラメータ

- **index**　コレクション内にパラメータを挿入するロケーションの 0 から始まるインデックス。
- **value**　挿入する ULParameter オブジェクト。

参照

- 「ULParameterCollection クラス」 392 ページ
- 「ULParameterCollection メンバ」 392 ページ
- 「ULParameter クラス」 375 ページ

Remove メソッド

ULParameter をコレクションから削除します。

構文

Visual Basic

```
Public Overrides Sub Remove( _
  ByVal value As Object _
)
```

C#

```
public override void Remove(
```

```
  object value
);
```

パラメータ

- **value**　削除する ULParameter オブジェクト。

参照

- 「ULParameterCollection クラス」 392 ページ
- 「ULParameterCollection メンバ」 392 ページ
- 「ULParameter クラス」 375 ページ

RemoveAt メソッド

指定された DbParameter オブジェクトをコレクションから削除します。

RemoveAt(Int32) メソッド

コレクションの指定されたインデックス位置のパラメータを削除します。

構文

Visual Basic

```
Public Overrides Sub RemoveAt( _
  ByVal index As Integer _
)
```

C#

```
public override void RemoveAt(
  int index
);
```

パラメータ

- **index**　削除するパラメータの 0 から始まるインデックス。値は、[0,ULParameterCollection.Count-1] の範囲内であることが必要です。コレクションの最初のパラメータのインデックス値は 0 です。

参照

- 「ULParameterCollection クラス」 392 ページ
- 「ULParameterCollection メンバ」 392 ページ
- 「RemoveAt メソッド」 410 ページ
- 「RemoveAt(String) メソッド」 410 ページ
- 「Count プロパティ」 394 ページ

RemoveAt(String) メソッド

指定された名前のパラメータをコレクションから削除します。

構文

Visual Basic

```
Public Overrides Sub RemoveAt( _
  ByVal parameterName As String _
)
```

C#

```
public override void RemoveAt(
  string  parameterName
);
```

パラメータ

- **parameterName**　取り出すパラメータの名前。

参照

- 「ULParameterCollection クラス」 392 ページ
- 「ULParameterCollection メンバ」 392 ページ
- 「RemoveAt メソッド」 410 ページ
- 「RemoveAt(Int32) メソッド」 410 ページ

ULPublicationSchema クラス

UL 拡張： Ultra Light パブリケーションのスキーマを表します。このクラスは継承できません。

構文

Visual Basic
Public NotInheritable Class **ULPublicationSchema**

C#
public sealed class **ULPublicationSchema**

備考

このクラスにはコンストラクタがありません。パブリケーションのスキーマは、ULDatabaseSchema クラスの ULDatabaseSchema.GetPublicationSchema メソッドを使用して作成されます。

パブリケーションが必要な一部のメソッドでは、カンマで区切られたパブリケーションのリストも受け取ります。

このクラスでは、次の特別な 2 つのパブリケーション値も提供されます。
ULPublicationSchema.SYNC_ALL_DB は、データベース全体に対応します。
ULPublicationSchema.SYNC_ALL_PUBS は、すべてのパブリケーションに対応します。

参照

- 「ULPublicationSchema メンバ」 412 ページ
- 「GetLastDownloadTime メソッド」 169 ページ
- 「CountUploadRows(String, UInt32) メソッド」 159 ページ
- 「Schema プロパティ」 151 ページ
- 「SYNC_ALL_DB フィールド」 413 ページ
- 「SYNC_ALL_PUBS フィールド」 413 ページ
- 「GetPublicationSchema メソッド」 272 ページ
- 「ULDatabaseSchema クラス」 265 ページ

ULPublicationSchema メンバ

パブリック・フィールド

メンバ名	説明
「SYNC_ALL_DB フィールド」 413 ページ	データベース全体を表す空のパブリケーション・リストです。このフィールドは定数で、読み込み専用です。
「SYNC_ALL_PUBS フィールド」 413 ページ	すべてのパブリケーションを表すパブリケーション名 "*" です。このフィールドは定数で、読み込み専用です。

パブリック・プロパティ

メンバ名	説明
「IsOpen プロパティ」 414 ページ	パブリケーション・スキーマが開いているか、閉じているかを調べます。
「Name プロパティ」 414 ページ	このパブリケーションの名前を返します。

参照

- 「ULPublicationSchema クラス」 412 ページ
- 「GetLastDownloadTime メソッド」 169 ページ
- 「CountUploadRows(String, UInt32) メソッド」 159 ページ
- 「Schema プロパティ」 151 ページ
- 「SYNC_ALL_DB フィールド」 413 ページ
- 「SYNC_ALL_PUBS フィールド」 413 ページ
- 「GetPublicationSchema メソッド」 272 ページ
- 「ULDatabaseSchema クラス」 265 ページ

SYNC_ALL_DB フィールド

データベース全体を表す空のパブリケーション・リストです。このフィールドは定数で、読み込み専用です。

構文

Visual Basic
Public Shared **SYNC_ALL_DB** As String

C#
public const string **SYNC_ALL_DB**;

参照

- 「ULPublicationSchema クラス」 412 ページ
- 「ULPublicationSchema メンバ」 412 ページ

SYNC_ALL_PUBS フィールド

すべてのパブリケーションを表すパブリケーション名 "*" です。このフィールドは定数で、読み込み専用です。

構文

Visual Basic
Public Shared **SYNC_ALL_PUBS** As String

C#
public const string **SYNC_ALL_PUBS**;

参照

- 「ULPublicationSchema クラス」 412 ページ
- 「ULPublicationSchema メンバ」 412 ページ

IsOpen プロパティ

パブリケーション・スキーマが開いているか、閉じているかを調べます。

構文

Visual Basic
Public Readonly Property **IsOpen** As Boolean

C#
public bool **IsOpen** { get;}

プロパティ値

パブリケーション・スキーマが開いている場合は true、閉じている場合は false。

参照

- 「ULPublicationSchema クラス」 412 ページ
- 「ULPublicationSchema メンバ」 412 ページ

Name プロパティ

このパブリケーションの名前を返します。

構文

Visual Basic
Public Readonly Property **Name** As String

C#
public string **Name** { get;}

プロパティ値

パブリケーションの名前を指定する文字列。

参照

- 「ULPublicationSchema クラス」 412 ページ
- 「ULPublicationSchema メンバ」 412 ページ

ULResultSet クラス

UL 拡張：Ultra Light データベースの編集可能な結果セットを表します。

構文

Visual Basic
Public Class **ULResultSet**
  Inherits ULDataReader

C#
public class **ULResultSet**: ULDataReader

備考

このクラスにはコンストラクタがありません。結果セットは、ULCommand クラスの ULCommand.ExecuteResultSet() メソッドを使用して作成されます。

```
' Visual Basic
Dim cmd As ULCommand = new ULCommand( _
    "SELECT emp_id FROM employee", conn _
  )
Dim resultSet As ULResultSet = cmd.ExecuteResultSet()

// C#
ULCommand cmd = new ULCommand(
    "SELECT emp_id FROM employee", conn
  );
ULResultSet resultSet = cmd.ExecuteResultSet();
```

ULResultSet オブジェクトは、位置付け更新や削除の実行対象となる編集可能な結果を表します。編集可能な結果セットには、ULCommand.ExecuteTable() または ULDataAdapter を使用します。

継承：ULDataReader

実装：System.Data.IDataReader、System.Data.IDataRecord、System.IDisposable

参照

- 「ULResultSet メンバ」 416 ページ
- 「ExecuteResultSet() メソッド」 122 ページ
- 「ULCommand クラス」 91 ページ
- 「ULResultSet クラス」 415 ページ
- 「ExecuteTable() メソッド」 125 ページ
- 「ULDataAdapter クラス」 242 ページ
- 「ULDataReader クラス」 276 ページ
- IDataReader
- IDataRecord
- IDisposable

ULResultSet メンバ

パブリック・プロパティ

メンバ名	説明
「Depth プロパティ」 281 ページ (ULDataReader から継承)	現在のローのネストの深さを返します。最も外側のテーブルの深さは 0 です。
「FieldCount プロパティ」 281 ページ (ULDataReader から継承)	このカーソル内のカラム数を返します。
「HasRows プロパティ」 282 ページ (ULDataReader から継承)	ULDataReader に 1 つまたは複数のローがあるかどうかをチェックします。
「IsBOF プロパティ」 282 ページ (ULDataReader から継承)	**UL 拡張：**現在のローの位置が最初のローの前かどうかをチェックします。
「IsClosed プロパティ」 283 ページ (ULDataReader から継承)	カーソルが現在開いているかどうかを確認します。
「IsEOF プロパティ」 283 ページ (ULDataReader から継承)	**UL 拡張：**現在のローの位置が最後のローの後かどうかをチェックします。
「Item プロパティ」 283 ページ (ULDataReader から継承)	指定されたカラムの値を Object のインスタンスとして取得します。
「RecordsAffected プロパティ」 285 ページ (ULDataReader から継承)	SQL 文の実行によって変更、挿入、または削除されたローの数を返します。SELECT 文または CommandType.TableDirect テーブルの場合、この値は -1 です。
「RowCount プロパティ」 286 ページ (ULDataReader から継承)	**UL 拡張：**カーソル内のローの数を返します。
「Schema プロパティ」 286 ページ (ULDataReader から継承)	**UL 拡張：**このカーソルのスキーマを保持します。
VisibleFieldCount (DbDataReader から継承)	DbDataReader の非表示でないフィールドの数を取得します。

パブリック・メソッド

メンバ名	説明
「AppendBytes メソッド」 422 ページ	指定された System.Bytes 配列の指定されたサブセットを、指定された ULDbType.LongBinary カラムの新しい値に追加します。

メンバ名	説明
「AppendChars メソッド」 423 ページ	指定された System.Chars 配列の指定されたサブセットを、指定された ULDbType.LongVarchar カラムの新しい値に追加します。
「Close メソッド」 287 ページ (ULDataReader から継承)	カーソルを閉じます。
「Delete メソッド」 425 ページ	現在の行を削除します。
Dispose (DbDataReader から継承)	この DbDataReader によって消費されたリソースを解放します。
「GetBoolean メソッド」 287 ページ (ULDataReader から継承)	指定されたカラムの値を System.Boolean として返します。
「GetByte メソッド」 288 ページ (ULDataReader から継承)	指定されたカラムの値を符号なし 8 ビット値 (System.Byte) として返します。
「GetBytes メソッド」 289 ページ (ULDataReader から継承)	**UL 拡張：**指定されたカラムの値を System.Bytes 配列として返します。ULDbType.Binary 型、ULDbType.LongBinary 型、ULDbType.UniqueIdentifier 型のカラムの場合にのみ有効です。
「GetChar メソッド」 291 ページ (ULDataReader から継承)	このメソッドは Ultra Light.NET ではサポートされていません。
「GetChars メソッド」 292 ページ (ULDataReader から継承)	指定されたオフセットで始まる、指定された ULDbType.LongVarchar カラムの値のサブセットを、コピー先の System.Char 配列の指定されたオフセットにコピーします。
GetData (DbDataReader から継承)	要求されたカラムの順序の DbDataReader オブジェクトを返します。
「GetDataTypeName メソッド」 293 ページ (ULDataReader から継承)	指定されたカラムのプロバイダのデータ型の名前を返します。
「GetDateTime メソッド」 294 ページ (ULDataReader から継承)	指定されたカラムの値を、ミリ秒の精度の System.DateTime として返します。

メンバ名	説明
「GetDecimal メソッド」 294 ページ (ULDataReader から継承)	指定されたカラムの値を System.Decimal として返します。
「GetDouble メソッド」 295 ページ (ULDataReader から継承)	指定されたカラムの値を System.Double として返します。
「GetEnumerator メソッド」 296 ページ (ULDataReader から継承)	ULDataReader の反復処理を実行する System.Collections.IEnumerator を返します。
「GetFieldType メソッド」 296 ページ (ULDataReader から継承)	指定されたカラムに最適な System.Type を返します。
「GetFloat メソッド」 297 ページ (ULDataReader から継承)	指定されたカラムの値を System.Single として返します。
「GetGuid メソッド」 298 ページ (ULDataReader から継承)	指定されたカラムの値を UUID (System.Guid) として返します。
「GetInt16 メソッド」 298 ページ (ULDataReader から継承)	指定されたカラムの値を System.Int16 として返します。
「GetInt32 メソッド」 299 ページ (ULDataReader から継承)	指定されたカラムの値を Int32 として返します。
「GetInt64 メソッド」 300 ページ (ULDataReader から継承)	指定されたカラムの値を Int64 として返します。
「GetName メソッド」 300 ページ (ULDataReader から継承)	指定されたカラムの名前を返します。
「GetOrdinal メソッド」 301 ページ (ULDataReader から継承)	指定されたカラムのカラム ID を返します。
GetProviderSpecificFieldType (DbDataReader から継承)	指定されたカラムのプロバイダ固有のフィールド・タイプを返します。

メンバ名	説明
GetProviderSpecificValue (DbDataReader から継承)	指定されたカラムの値を Object のインスタンスとして取得します。
GetProviderSpecificValues (DbDataReader から継承)	現在のローのコレクション内のプロバイダ固有のすべての属性カラムを取得します。
「GetSchemaTable メソッド」 303 ページ (ULDataReader から継承)	ULDataReader のカラムのメタデータが記述された System.Data.DataTable を返します。
「GetString メソッド」 305 ページ (ULDataReader から継承)	指定されたカラムの値を System.String として返します。
「GetTimeSpan メソッド」 306 ページ (ULDataReader から継承)	指定されたカラムの値を、ミリ秒の精度の System.TimeSpan として返します。
「GetUInt16 メソッド」 306 ページ (ULDataReader から継承)	指定されたカラムの値を System.UInt16 として返します。
「GetUInt32 メソッド」 307 ページ (ULDataReader から継承)	指定されたカラムの値を UInt32 として返します。
「GetUInt64 メソッド」 308 ページ (ULDataReader から継承)	指定されたカラムの値を System.UInt64 として返します。
「GetValue メソッド」 308 ページ (ULDataReader から継承)	指定されたカラムの値をネイティブ・フォーマットで返します。
「GetValues メソッド」 309 ページ (ULDataReader から継承)	現在のローのすべてのカラム値を返します。
「IsDBNull メソッド」 310 ページ (ULDataReader から継承)	指定されたカラムの値が NULL かどうかをチェックします。
「MoveAfterLast メソッド」 311 ページ (ULDataReader から継承)	**UL 拡張：**カーソルの最後のローの後に、カーソルを配置します。

メンバ名	説明
「MoveBeforeFirst メソッド」 311 ページ (ULDataReader から継承)	**UL 拡張：**カーソルの最初のローの前に、カーソルを配置します。
「MoveFirst メソッド」 311 ページ (ULDataReader から継承)	**UL 拡張：**カーソルの最初のローに、カーソルを配置します。
「MoveLast メソッド」 312 ページ (ULDataReader から継承)	**UL 拡張：**カーソルの最後のローに、カーソルを配置します。
「MoveNext メソッド」 312 ページ (ULDataReader から継承)	**UL 拡張：**カーソルを次のローに配置します。ただし、カーソルがすでに最後のローにある場合は最後のローの後に配置します。
「MovePrevious メソッド」 313 ページ (ULDataReader から継承)	**UL 拡張：**カーソルを前のローに配置するか、最初のローの前に配置します。
「MoveRelative メソッド」 313 ページ (ULDataReader から継承)	**UL 拡張：**現在のローを基準としてカーソルを配置します。
「NextResult メソッド」 314 ページ (ULDataReader から継承)	バッチ SQL 文の結果を読み込むときに ULDataReader を次の結果に進めます。
「Read メソッド」 314 ページ (ULDataReader から継承)	カーソルを次のローに配置します。ただし、カーソルがすでに最後のローにある場合は最後のローの後に配置します。
「SetBoolean メソッド」 425 ページ	指定されたカラムの値を、System.Boolean を使用して設定します。
「SetByte メソッド」 426 ページ	指定されたカラムの値を、System.Byte (符号なし 8 ビット整数) を使用して設定します。
「SetBytes メソッド」 427 ページ	指定されたカラムの値を、System.Bytes を使用して設定します。
「SetDBNull メソッド」 428 ページ	カラムを NULL に設定します。
「SetDateTime メソッド」 429 ページ	指定されたカラムの値を、="System.DateTime を使用して設定します。

メンバ名	説明
「SetDecimal メソッド」 429 ページ	指定されたカラムの値を、System.Decimal を使用して設定します。
「SetDouble メソッド」 430 ページ	指定されたカラムの値を、System.Double を使用して設定します。
「SetFloat メソッド」 431 ページ	指定されたカラムの値を、System.Single を使用して設定します。
「SetGuid メソッド」 432 ページ	指定されたカラムの値を、System.Guid を使用して設定します。
「SetInt16 メソッド」 433 ページ	指定されたカラムの値を、System.Int16 を使用して設定します。
「SetInt32 メソッド」 434 ページ	指定されたカラムの値を、System.Int32 を使用して設定します。
「SetInt64 メソッド」 435 ページ	指定されたカラムの値を、Int64 を使用して設定します。
「SetString メソッド」 436 ページ	指定されたカラムの値を、System.String を使用して設定します。
「SetTimeSpan メソッド」 437 ページ	指定されたカラムの値を、System.TimeSpan を使用して設定します。
「SetToDefault メソッド」 437 ページ	指定されたカラムの値を、そのデフォルト値に設定します。
「SetUInt16 メソッド」 438 ページ	指定されたカラムの値を、System.UInt16 を使用して設定します。
「SetUInt32 メソッド」 439 ページ	指定されたカラムの値を、System.UInt32 を使用して設定します。
「SetUInt64 メソッド」 440 ページ	指定されたカラムの値を、System.UInt64 を使用して設定します。
「Update メソッド」 441 ページ	現在のカラム値 (set メソッドを使用して指定されます) で新しいローを更新します。

参照

- 「ULResultSet クラス」 415 ページ
- 「ExecuteResultSet() メソッド」 122 ページ
- 「ULCommand クラス」 91 ページ
- 「ULResultSet クラス」 415 ページ
- 「ExecuteTable() メソッド」 125 ページ
- 「ULDataAdapter クラス」 242 ページ
- 「ULDataReader クラス」 276 ページ
- IDataReader
- IDataRecord
- IDisposable

AppendBytes メソッド

指定された System.Bytes 配列の指定されたサブセットを、指定された ULDbType.LongBinary カラムの新しい値に追加します。

構文

Visual Basic

```
Public Sub AppendBytes( _
  ByVal columnID As Integer, _
  ByVal val As Byte(), _
  ByVal srcOffset As Integer, _
  ByVal count As Integer _
)
```

C#

```
public void AppendBytes(
  int columnID,
  byte[] val,
  int srcOffset,
  int count
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。
- **val**　カラムの現在の新しい値に追加する値。
- **srcOffset**　ソース配列の開始位置。
- **count**　コピーされるバイト数。

備考

srcOffset (0 から始まります) から、配列 *val* の *srcOffset*+*count*-1 までの位置のバイトが、指定されたカラムの値に追加されます。

挿入時には、ULTable.InsertBegin は新しい値をカラムのデフォルト値に初期化します。ローのデータは、ULTable.Insert が実行されるまで実際には変更されません。また、その変更は、コミットされないかぎり、永続化されません。

更新時の、カラムに対する最初の追加では、現在のカラム値がクリアされてから新しい値が追加されます。

次のいずれかに該当する場合、コード ULSQLCode.SQLE_INVALID_PARAMETER とともに ULException がスローされ、追加先は修正されません。

- *val* が NULL である。
- *srcOffset* が負である。
- *count* が負である。
- *srcOffset*+*count* が *val*.Length. より長い。

その他のエラーの場合は、それに応じたエラー・コードとともに ULException がスローされます。

参照

- 「ULResultSet クラス」 415 ページ
- 「ULResultSet メンバ」 416 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「Schema プロパティ」 286 ページ
- 「GetFieldType メソッド」 296 ページ
- Byte
- 「ULDbType 列挙体」 317 ページ
- 「InsertBegin メソッド」 565 ページ
- 「Insert メソッド」 564 ページ
- 「ULException クラス」 322 ページ
- 「ULSQLCode 列挙体」 462 ページ
- 「ULException クラス」 322 ページ
- 「FieldCount プロパティ」 281 ページ

AppendChars メソッド

指定された System.Chars 配列の指定されたサブセットを、指定された ULDbType.LongVarchar カラムの新しい値に追加します。

構文

Visual Basic

```
Public Sub AppendChars( _
  ByVal columnID As Integer, _
  ByVal val As Char(), _
  ByVal srcOffset As Integer, _
  ByVal count As Integer _
)
```

C#
```
public void AppendChars(
  int columnID,
  char[] val,
  int srcOffset,
  int count
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。
- **val**　カラムの現在の新しい値に追加する値。
- **srcOffset**　ソース配列の開始位置。
- **count**　コピーされるバイト数。

備考

srcOffset (0 から始まります) から、配列 *val* の *srcOffset*+*count*-1 までの位置の文字が、指定されたカラムの値に追加されます。挿入時には、ULTable.InsertBegin は新しい値をカラムのデフォルト値に初期化します。ローのデータは、ULTable.Insert が実行されるまで実際には変更されません。また、その変更は、コミットされないかぎり、永続化されません。

更新時の、カラムに対する最初の追加では、現在のカラム値がクリアされてから新しい値が追加されます。

次のいずれかに該当する場合、コード ULSQLCode.SQLE_INVALID_PARAMETER とともに ULException がスローされ、追加先は修正されません。

- *val* が NULL である。
- *srcOffset* が負である。
- *count* が負である。
- *srcOffset*+*count* が *value*.Length. より長い。

その他のエラーの場合は、それに応じたエラー・コードとともに ULException がスローされます。

参照

- 「ULResultSet クラス」 415 ページ
- 「ULResultSet メンバ」 416 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「Schema プロパティ」 286 ページ
- 「GetFieldType メソッド」 296 ページ
- Char
- 「ULDbType 列挙体」 317 ページ
- 「InsertBegin メソッド」 565 ページ
- 「Insert メソッド」 564 ページ
- 「ULException クラス」 322 ページ
- 「ULSQLCode 列挙体」 462 ページ
- 「FieldCount プロパティ」 281 ページ

Delete メソッド

現在の行を削除します。

構文

Visual Basic
Public Sub **Delete()**

C#
public void **Delete();**

参照

- 「ULResultSet クラス」 415 ページ
- 「ULResultSet メンバ」 416 ページ
- 「StartSynchronizationDelete メソッド」 181 ページ
- 「StopSynchronizationDelete メソッド」 182 ページ

SetBoolean メソッド

指定されたカラムの値を、System.Boolean を使用して設定します。

構文

Visual Basic
```
Public Sub SetBoolean( _
  ByVal columnID As Integer, _
  ByVal val As Boolean _
)
```

C#
```
public void SetBoolean(
  int columnID,
```

```
  bool val
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。
- **val**　カラムの新しい値。

備考

ローのデータは、ULTable.Insert または Update が実行されるまで実際には変更されません。また、その変更は、コミットされないかぎり、永続化されません。

参照

- 「ULResultSet クラス」 415 ページ
- 「ULResultSet メンバ」 416 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「Schema プロパティ」 286 ページ
- 「GetFieldType メソッド」 296 ページ
- Boolean
- 「Insert メソッド」 564 ページ
- 「Update メソッド」 441 ページ
- 「FieldCount プロパティ」 281 ページ

SetByte メソッド

指定されたカラムの値を、System.Byte (符号なし 8 ビット整数) を使用して設定します。

構文

Visual Basic

```
Public Sub SetByte( _
  ByVal columnID As Integer, _
  ByVal val As Byte _
)
```

C#

```
public void SetByte(
  int columnID,
  byte val
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。
- **val**　カラムの新しい値。

備考

ローのデータは、ULTable.Insert または Update が実行されるまで実際には変更されません。また、その変更は、コミットされないかぎり、永続化されません。

参照

- 「ULResultSet クラス」 415 ページ
- 「ULResultSet メンバ」 416 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「Schema プロパティ」 286 ページ
- 「GetFieldType メソッド」 296 ページ
- Byte
- 「Insert メソッド」 564 ページ
- 「Update メソッド」 441 ページ
- 「FieldCount プロパティ」 281 ページ

SetBytes メソッド

指定されたカラムの値を、System.Bytes を使用して設定します。

構文

Visual Basic

```
Public Sub SetBytes( _
  ByVal columnID As Integer, _
  ByVal val As Byte() _
)
```

C#

```
public void SetBytes(
  int columnID,
  byte[] val
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。
- **val**　カラムの新しい値。

備考

ULDbType.Binary 型または ULDbType.LongBinary 型のカラム、あるいは val の長さが 16 の場合の ULDbType.UniqueIdentifier 型のカラムにのみ適しています。ローのデータは、ULTable.Insert または Update が実行されるまで実際には変更されません。また、その変更は、コミットされないかぎり、永続化されません。

参照

- 「ULResultSet クラス」 415 ページ
- 「ULResultSet メンバ」 416 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「Schema プロパティ」 286 ページ
- 「GetFieldType メソッド」 296 ページ
- Byte
- 「Insert メソッド」 564 ページ
- 「Update メソッド」 441 ページ
- 「FieldCount プロパティ」 281 ページ
- 「ULDbType 列挙体」 317 ページ
- 「ULDbType 列挙体」 317 ページ
- 「ULDbType 列挙体」 317 ページ

SetDBNull メソッド

カラムを NULL に設定します。

構文

Visual Basic

```
Public Sub SetDBNull( _
  ByVal columnID As Integer _
)
```

C#

```
public void SetDBNull(
  int columnID
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。

備考

データは、ULTable.Insert または Update を実行するまでは、実際には変更されません。また、その変更は、コミットされないかぎり、永続化されません。

参照

- 「ULResultSet クラス」 415 ページ
- 「ULResultSet メンバ」 416 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「Schema プロパティ」 286 ページ
- 「IsColumnNullable メソッド」 584 ページ
- 「Insert メソッド」 564 ページ
- 「Update メソッド」 441 ページ
- 「FieldCount プロパティ」 281 ページ

SetDateTime メソッド

指定されたカラムの値を、System.DateTime を使用して設定します。

構文

Visual Basic
```
Public Sub SetDateTime( _
  ByVal columnID As Integer, _
  ByVal val As Date _
)
```

C#
```
public void SetDateTime(
  int columnID,
  DateTime val
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。
- **val**　カラムの新しい値。

備考

設定された値の精度はミリ秒です。ローのデータは、ULTable.Insert または Update が実行されるまで実際には変更されません。また、その変更は、コミットされないかぎり、永続化されません。

参照

- 「ULResultSet クラス」 415 ページ
- 「ULResultSet メンバ」 416 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「Schema プロパティ」 286 ページ
- 「GetFieldType メソッド」 296 ページ
- DateTime
- 「Insert メソッド」 564 ページ
- 「Update メソッド」 441 ページ
- 「FieldCount プロパティ」 281 ページ

SetDecimal メソッド

指定されたカラムの値を、System.Decimal を使用して設定します。

構文

Visual Basic
```
Public Sub SetDecimal( _
  ByVal columnID As Integer, _
```

```
  ByVal val As Decimal _
)

C#
public void SetDecimal(
  int columnID,
  decimal val
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。
- **val**　カラムの新しい値。

備考

ローのデータは、ULTable.Insert または Update が実行されるまで実際には変更されません。また、その変更は、コミットされないかぎり、永続化されません。

参照

- 「ULResultSet クラス」 415 ページ
- 「ULResultSet メンバ」 416 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「Schema プロパティ」 286 ページ
- 「GetFieldType メソッド」 296 ページ
- Decimal
- 「Insert メソッド」 564 ページ
- 「Update メソッド」 441 ページ
- 「FieldCount プロパティ」 281 ページ

SetDouble メソッド

指定されたカラムの値を、System.Double を使用して設定します。

構文

```
Visual Basic
Public Sub SetDouble( _
  ByVal columnID As Integer, _
  ByVal val As Double _
)

C#
public void SetDouble(
  int columnID,
  double val
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。
- **val**　カラムの新しい値。

備考

ローのデータは、ULTable.Insert または Update が実行されるまで実際には変更されません。また、その変更は、コミットされないかぎり、永続化されません。

参照

- 「ULResultSet クラス」 415 ページ
- 「ULResultSet メンバ」 416 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「Schema プロパティ」 286 ページ
- 「GetFieldType メソッド」 296 ページ
- Double
- 「Insert メソッド」 564 ページ
- 「Update メソッド」 441 ページ
- 「FieldCount プロパティ」 281 ページ

SetFloat メソッド

指定されたカラムの値を、System.Single を使用して設定します。

構文

Visual Basic

```
Public Sub SetFloat( _
  ByVal columnID As Integer, _
  ByVal val As Single _
)
```

C#

```
public void SetFloat(
  int columnID,
  float  val
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。
- **val**　カラムの新しい値。

備考

ローのデータは、ULTable.Insert または Update が実行されるまで実際には変更されません。また、その変更は、コミットされないかぎり、永続化されません。

参照

- 「ULResultSet クラス」 415 ページ
- 「ULResultSet メンバ」 416 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「Schema プロパティ」 286 ページ
- 「GetFieldType メソッド」 296 ページ
- Single
- 「Insert メソッド」 564 ページ
- 「Update メソッド」 441 ページ
- 「FieldCount プロパティ」 281 ページ

SetGuid メソッド

指定されたカラムの値を、System.Guid を使用して設定します。

構文

Visual Basic

```
Public Sub SetGuid( _
  ByVal columnID As Integer, _
  ByVal val As Guid _
)
```

C#

```
public void SetGuid(
  int columnID,
  Guid val
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。
- **val**　カラムの新しい値。

備考

ローのデータは、ULTable.Insert または Update が実行されるまで実際には変更されません。また、その変更は、コミットされないかぎり、永続化されません。ULDbType.UniqueIdentifier 型のカラム、または長さが 16 の ULDbType.Binary 型のカラムの場合にのみ有効です。

参照

- 「ULResultSet クラス」 415 ページ
- 「ULResultSet メンバ」 416 ページ
- 「GetNewUUID メソッド」 170 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「Schema プロパティ」 286 ページ
- 「GetFieldType メソッド」 296 ページ
- 「GetColumnSize メソッド」 239 ページ
- Guid
- 「Insert メソッド」 564 ページ
- 「Update メソッド」 441 ページ
- 「ULDbType 列挙体」 317 ページ
- 「ULDbType 列挙体」 317 ページ
- 「FieldCount プロパティ」 281 ページ

SetInt16 メソッド

指定されたカラムの値を、System.Int16 を使用して設定します。

構文

Visual Basic

```
Public Sub SetInt16( _
  ByVal columnID As Integer, _
  ByVal val As Short _
)
```

C#

```
public void SetInt16(
  int columnID,
  short val
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。
- **val**　カラムの新しい値。

備考

ローのデータは、ULTable.Insert または Update が実行されるまで実際には変更されません。また、その変更は、コミットされないかぎり、永続化されません。

参照

- 「ULResultSet クラス」 415 ページ
- 「ULResultSet メンバ」 416 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「Schema プロパティ」 286 ページ
- 「GetFieldType メソッド」 296 ページ
- Int16
- 「Insert メソッド」 564 ページ
- 「Update メソッド」 441 ページ
- 「FieldCount プロパティ」 281 ページ

SetInt32 メソッド

指定されたカラムの値を、System.Int32 を使用して設定します。

構文

Visual Basic

```
Public Sub SetInt32( _
  ByVal columnID As Integer, _
  ByVal val As Integer _
)
```

C#

```
public void SetInt32(
  int columnID,
  int val
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。
- **val**　カラムの新しい値。

備考

ローのデータは、ULTable.Insert または Update が実行されるまで実際には変更されません。また、その変更は、コミットされないかぎり、永続化されません。

参照

- 「ULResultSet クラス」 415 ページ
- 「ULResultSet メンバ」 416 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「Schema プロパティ」 286 ページ
- 「GetFieldType メソッド」 296 ページ
- Int32
- 「Insert メソッド」 564 ページ
- 「Update メソッド」 441 ページ
- 「FieldCount プロパティ」 281 ページ

SetInt64 メソッド

指定されたカラムの値を、Int64 を使用して設定します。

構文

Visual Basic

```
Public Sub SetInt64( _
  ByVal columnID As Integer, _
  ByVal val As Long _
)
```

C#

```
public void SetInt64(
  int columnID,
  long val
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。
- **val**　カラムの新しい値。

備考

ローのデータは、ULTable.Insert または Update が実行されるまで実際には変更されません。また、その変更は、コミットされないかぎり、永続化されません。

参照

- 「ULResultSet クラス」 415 ページ
- 「ULResultSet メンバ」 416 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「Schema プロパティ」 286 ページ
- 「GetFieldType メソッド」 296 ページ
- Int64
- 「Insert メソッド」 564 ページ
- 「Update メソッド」 441 ページ
- 「FieldCount プロパティ」 281 ページ

SetString メソッド

指定されたカラムの値を、System.String を使用して設定します。

構文

Visual Basic

```
Public Sub SetString( _
  ByVal columnID As Integer, _
  ByVal val As String _
)
```

C#

```
public void SetString(
  int columnID,
  string  val
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。
- **val**　カラムの新しい値。

備考

ローのデータは、ULTable.Insert または Update が実行されるまで実際には変更されません。また、その変更は、コミットされないかぎり、永続化されません。

参照

- 「ULResultSet クラス」 415 ページ
- 「ULResultSet メンバ」 416 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「Schema プロパティ」 286 ページ
- 「GetFieldType メソッド」 296 ページ
- 「Insert メソッド」 564 ページ
- 「Update メソッド」 441 ページ
- 「FieldCount プロパティ」 281 ページ

SetTimeSpan メソッド

指定されたカラムの値を、System.TimeSpan を使用して設定します。

構文

Visual Basic
```
Public Sub SetTimeSpan( _
  ByVal columnID As Integer, _
  ByVal val As TimeSpan _
)
```

C#
```
public void SetTimeSpan(
  int columnID,
  TimeSpan val
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。
- **val**　カラムの新しい値。

備考

設定された値は、ミリ秒の精度であり、0 ～ 24 時までの間の負でない値に正規化されます。ローのデータは、ULTable.Insert または Update が実行されるまで実際には変更されません。また、その変更は、コミットされないかぎり、永続化されません。

参照

- 「ULResultSet クラス」 415 ページ
- 「ULResultSet メンバ」 416 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「Schema プロパティ」 286 ページ
- 「GetFieldType メソッド」 296 ページ
- TimeSpan
- 「Insert メソッド」 564 ページ
- 「Update メソッド」 441 ページ
- 「FieldCount プロパティ」 281 ページ

SetToDefault メソッド

指定されたカラムの値を、そのデフォルト値に設定します。

構文

Visual Basic
```
Public Sub SetToDefault( _
```

```
  ByVal columnID As Integer _
)

C#
public void SetToDefault(
  int columnID
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。

備考

ローのデータは、ULTable.Insert または Update が実行されるまで実際には変更されません。また、その変更は、コミットされないかぎり、永続化されません。

参照

- 「ULResultSet クラス」 415 ページ
- 「ULResultSet メンバ」 416 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「Schema プロパティ」 286 ページ
- 「GetFieldType メソッド」 296 ページ
- 「GetColumnDefaultValue メソッド」 576 ページ
- 「Insert メソッド」 564 ページ
- 「Update メソッド」 441 ページ
- 「FieldCount プロパティ」 281 ページ

SetUInt16 メソッド

指定されたカラムの値を、System.UInt16 を使用して設定します。

構文

```
Visual Basic
Public Sub SetUInt16( _
  ByVal columnID As Integer, _
  ByVal val As UInt16 _
)

C#
public void SetUInt16(
  int columnID,
  ushort val
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。

- **val**　カラムの新しい値。

備考

ローのデータは、ULTable.Insert または Update が実行されるまで実際には変更されません。また、その変更は、コミットされないかぎり、永続化されません。

参照

- 「ULResultSet クラス」 415 ページ
- 「ULResultSet メンバ」 416 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「GetFieldType メソッド」 296 ページ
- UInt16
- 「Insert メソッド」 564 ページ
- 「Update メソッド」 441 ページ
- 「FieldCount プロパティ」 281 ページ

SetUInt32 メソッド

指定されたカラムの値を、System.UInt32 を使用して設定します。

構文

Visual Basic

```
Public Sub SetUInt32( _
  ByVal columnID As Integer, _
  ByVal val As UInt32 _
)
```

C#

```
public void SetUInt32(
  int columnID,
  uint val
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。
- **val**　カラムの新しい値。

備考

ローのデータは、ULTable.Insert または Update が実行されるまで実際には変更されません。また、その変更は、コミットされないかぎり、永続化されません。

参照

- 「ULResultSet クラス」 415 ページ
- 「ULResultSet メンバ」 416 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「Schema プロパティ」 286 ページ
- 「GetFieldType メソッド」 296 ページ
- UInt32
- 「Insert メソッド」 564 ページ
- 「Update メソッド」 441 ページ
- 「FieldCount プロパティ」 281 ページ

SetUInt64 メソッド

指定されたカラムの値を、System.UInt64 を使用して設定します。

構文

Visual Basic

```
Public Sub SetUInt64( _
  ByVal columnID As Integer, _
  ByVal val As UInt64 _
)
```

C#

```
public void SetUInt64(
  int columnID,
  ulong val
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULDataReader.FieldCount-1] の範囲内であることが必要です。カーソルの先頭カラムの ID 値は 0 です。
- **val**　カラムの新しい値。

備考

ローのデータは、ULTable.Insert または Update が実行されるまで実際には変更されません。また、その変更は、コミットされないかぎり、永続化されません。

参照

- 「ULResultSet クラス」 415 ページ
- 「ULResultSet メンバ」 416 ページ
- 「GetOrdinal メソッド」 301 ページ
- 「Schema プロパティ」 286 ページ
- 「GetFieldType メソッド」 296 ページ
- UInt64
- 「Insert メソッド」 564 ページ
- 「Update メソッド」 441 ページ
- 「FieldCount プロパティ」 281 ページ

Update メソッド

現在のカラム値 (set メソッドを使用して指定されます) で新しいローを更新します。

構文

Visual Basic
Public Sub **Update()**

C#
public void **Update();**

参照

- 「ULResultSet クラス」 415 ページ
- 「ULResultSet メンバ」 416 ページ

ULResultSetSchema クラス

UL 拡張：Ultra Light の結果セットのスキーマを表します。このクラスは継承できません。

構文

Visual Basic
Public NotInheritable Class **ULResultSetSchema**
  Inherits ULCursorSchema

C#
public sealed class **ULResultSetSchema**: ULCursorSchema

備考

このクラスにはコンストラクタがありません。ULResultSetSchema オブジェクトは、その ULDataReader.Schema として結果セットにアタッチされます。

結果セットのスキーマが有効なのは、データ・リーダが開かれている間だけです。

継承：ULCursorSchema

参照

- 「ULResultSetSchema メンバ」 442 ページ
- 「ULCommand クラス」 91 ページ
- 「ULDataReader クラス」 276 ページ
- 「ULResultSetSchema クラス」 442 ページ
- 「Schema プロパティ」 286 ページ
- 「ULCursorSchema クラス」 233 ページ

ULResultSetSchema メンバ

パブリック・プロパティ

メンバ名	説明
「ColumnCount プロパティ」 234 ページ (ULCursorSchema から継承)	このカーソル内のカラム数を返します。
「IsOpen プロパティ」 235 ページ (ULCursorSchema から継承)	カーソルのスキーマが現在開いているかどうかを確認します。
「Name プロパティ」 444 ページ	カーソルの名前を返します。

パブリック・メソッド

メンバ名	説明
「GetColumnID メソッド」 235 ページ (ULCursorSchema から継承)	指定されたカラムのカラム ID を返します。
「GetColumnName メソッド」 236 ページ (ULCursorSchema から継承)	指定されたカラム ID で識別されたカラムの名前を返します。
「GetColumnPrecision メソッド」 237 ページ (ULCursorSchema から継承)	カラムが数値カラム (SQL NUMERIC 型) の場合は、指定されたカラム ID によって識別されたカラムの精度を返します。
「GetColumnSQLName メソッド」 238 ページ (ULCursorSchema から継承)	指定されたカラム ID で識別されたカラムの名前を返します。
「GetColumnScale メソッド」 239 ページ (ULCursorSchema から継承)	カラムが数値カラム (SQL NUMERIC 型) の場合は、指定されたカラム ID によって識別されたカラムの位取りを返します。
「GetColumnSize メソッド」 239 ページ (ULCursorSchema から継承)	カラムがサイズ指定されたカラム (SQL BINARY 型または CHAR 型) の場合は、指定されたカラム ID によって識別されたカラムのサイズを返します。
「GetColumnULDbType メソッド」 240 ページ (ULCursorSchema から継承)	指定されたカラム ID によって識別されたカラムの Ultra Light.NET データ型を返します。
「GetSchemaTable メソッド」 241 ページ (ULCursorSchema から継承)	ULDataReader のカラムのスキーマが記述された System.Data.DataTable を返します。

参照

- 「ULResultSetSchema クラス」 442 ページ
- 「ULCommand クラス」 91 ページ
- 「ULDataReader クラス」 276 ページ
- 「ULResultSetSchema クラス」 442 ページ
- 「Schema プロパティ」 286 ページ
- 「ULCursorSchema クラス」 233 ページ

Name プロパティ

カーソルの名前を返します。

構文

Visual Basic
Public Overrides Readonly Property **Name** As String

C#
public override string **Name** { get;}

プロパティ値

ULResultSetSchema を生成した SQL 文。

参照

- 「ULResultSetSchema クラス」 442 ページ
- 「ULResultSetSchema メンバ」 442 ページ

ULRowsCopiedEventArgs クラス

ULRowsCopiedEventHandler に渡される引数のセットを表します。このクラスは継承できません。

構文

Visual Basic
Public NotInheritable Class **ULRowsCopiedEventArgs**

C#
public sealed class **ULRowsCopiedEventArgs**

備考

制限：ULRowsCopiedEventArgs クラスは、.NET Compact Framework 2.0 では使用できません。

参照

- 「ULRowsCopiedEventArgs メンバ」 445 ページ
- 「ULRowsCopiedEventHandler デリゲート」 448 ページ

ULRowsCopiedEventArgs メンバ

パブリック・コンストラクタ

メンバ名	説明
「ULRowsCopiedEventArgs コンストラクタ」 445 ページ	ULRowsCopiedEventArgs オブジェクトの新しいインスタンスを作成します。

パブリック・プロパティ

メンバ名	説明
「Abort プロパティ」 446 ページ	バルク・コピー・オペレーションをアボートするかどうかを示す値を取得または設定します。
「RowsCopied プロパティ」 446 ページ	現在のバルク・コピー・オペレーションでコピーされるローの数を返します。

参照

- 「ULRowsCopiedEventArgs クラス」 445 ページ
- 「ULRowsCopiedEventHandler デリゲート」 448 ページ

ULRowsCopiedEventArgs コンストラクタ

ULRowsCopiedEventArgs オブジェクトの新しいインスタンスを作成します。

構文

Visual Basic
```
Public Sub New( _
  ByVal rowsCopied As Long _
)
```

C#
```
public  ULRowsCopiedEventArgs(
  long rowsCopied
);
```

パラメータ

- **rowsCopied**　現在のバルク・コピー・オペレーションでコピーされるローの数を示す 64 ビット整数値。

備考

制限：ULRowsCopiedEventArgs クラスは、.NET Compact Framework 2.0 では使用できません。

参照

- 「ULRowsCopiedEventArgs クラス」 445 ページ
- 「ULRowsCopiedEventArgs メンバ」 445 ページ

Abort プロパティ

バルク・コピー・オペレーションをアボートするかどうかを示す値を取得または設定します。

構文

Visual Basic
```
Public Property Abort As Boolean
```

C#
```
public bool Abort { get; set; }
```

備考

制限：ULRowsCopiedEventArgs クラスは、.NET Compact Framework 2.0 では使用できません。

参照

- 「ULRowsCopiedEventArgs クラス」 445 ページ
- 「ULRowsCopiedEventArgs メンバ」 445 ページ

RowsCopied プロパティ

現在のバルク・コピー・オペレーションでコピーされるローの数を返します。

構文

Visual Basic
Public Readonly Property **RowsCopied** As Long

C#
public long **RowsCopied** { get;}

プロパティ値

コピーされたロー数を表す long integer。

備考

制限：ULRowsCopiedEventArgs クラスは、.NET Compact Framework 2.0 では使用できません。

参照

- 「ULRowsCopiedEventArgs クラス」 445 ページ
- 「ULRowsCopiedEventArgs メンバ」 445 ページ

ULRowsCopiedEventHandler デリゲート

ULBulkCopy.ULRowsCopied イベントを処理するメソッドを表します。

構文

Visual Basic

```
Public Delegate Sub ULRowsCopiedEventHandler( _
  ByVal sender As Object, _
  ByVal rowsCopiedEventArgs As ULRowsCopiedEventArgs _
)
```

C#

```
public delegate void ULRowsCopiedEventHandler(
  object sender,
  ULRowsCopiedEventArgs rowsCopiedEventArgs
);
```

備考

制限：ULRowsCopiedEventHandler デリゲートは、.NET Compact Framework 2.0 では使用できません。

参照

- 「ULRowsCopied イベント」 73 ページ
- オブジェクト

ULRowUpdatedEventArgs クラス

ULDataAdapter.RowUpdated イベントのデータを提供します。このクラスは継承できません。

構文

Visual Basic
Public NotInheritable Class **ULRowUpdatedEventArgs**
  Inherits RowUpdatedEventArgs

C#
public sealed class **ULRowUpdatedEventArgs**: RowUpdatedEventArgs

備考

継承：System.Data.Common.RowUpdatedEventArgs

参照

- 「ULRowUpdatedEventArgs メンバ」 449 ページ
- 「RowUpdated イベント」 252 ページ
- RowUpdatedEventArgs

ULRowUpdatedEventArgs メンバ

パブリック・コンストラクタ

メンバ名	説明
「ULRowUpdatedEventArgs コンストラクタ」 450 ページ	ULRowUpdatedEventArgs クラスの新しいインスタンスを初期化します。

パブリック・プロパティ

メンバ名	説明
「Command プロパティ」 451 ページ	DbDataAdapter.Update(System.Data.DataRow[],System.Data.Common.DataTableMapping) が呼び出されたときに実行された ULCommand を返します。
Errors (RowUpdatedEventArgs から継承)	RowUpdatedEventArgs.Command の実行時に .NET Framework データ・プロバイダによって生成されるエラーを取得します。
「RecordsAffected プロパティ」 452 ページ	SQL 文の実行によって変更、挿入、または削除されたローの数を返します。SELECT 文の場合、この値は -1 です。

メンバ名	説明
Row (RowUpdatedEventArgs から継承)	DbDataAdapter.Update によって送信された DataRow を取得します。
RowCount (RowUpdatedEventArgs から継承)	更新済みレコードのバッチで処理されたローの数を取得します。
StatementType (RowUpdatedEventArgs から継承)	実行された SQL 文のタイプを取得します。
Status (RowUpdatedEventArgs から継承)	RowUpdatedEventArgs.Command の UpdateStatus を取得します。
TableMapping (RowUpdatedEventArgs から継承)	DbDataAdapter.Update によって送信された DataTableMapping を取得します。

パブリック・メソッド

メンバ名	説明
CopyToRows (RowUpdatedEventArgs から継承)	バッチ更新操作の実行中に処理されるローにアクセスできます。

参照

- 「ULRowUpdatedEventArgs クラス」 449 ページ
- 「RowUpdated イベント」 252 ページ
- RowUpdatedEventArgs

ULRowUpdatedEventArgs コンストラクタ

ULRowUpdatedEventArgs クラスの新しいインスタンスを初期化します。

構文

Visual Basic

```
Public Sub New( _
   ByVal row As DataRow, _
   ByVal command As IDbCommand, _
   ByVal statementType As StatementType, _
   ByVal tableMapping As DataTableMapping _
)
```

C#

```
public  ULRowUpdatedEventArgs(
   DataRow row,
   IDbCommand command,
```



```
  StatementType statementType,
  DataTableMapping tableMapping
);
```

パラメータ

- **row** DbDataAdapter.Update(System.Data.DataRow[],System.Data.Common.DataTableMapping) によって送信された System.Data.DataRow。
- **command** DbDataAdapter.Update(System.Data.DataRow[],System.Data.Common.DataTableMapping) が呼び出されたときに実行された System.Data.IDbCommand。
- **statementType** 実行されたクエリのタイプを指定する System.Data.StatementType 値の 1 つ。
- **tableMapping** DbDataAdapter.Update(System.Data.DataRow[],System.Data.Common.DataTableMapping) によって送信された System.Data.Common.DataTableMapping。

参照

- 「ULRowUpdatedEventArgs クラス」 449 ページ
- 「ULRowUpdatedEventArgs メンバ」 449 ページ
- DataRow
- IDbCommand
- StatementType
- DataTableMapping
- DbDataAdapter.Update

Command プロパティ

DbDataAdapter.Update(System.Data.DataRow[],System.Data.Common.DataTableMapping) が呼び出されたときに実行された ULCommand を返します。

構文

Visual Basic
Public Readonly Property **Command** As ULCommand

C#
public ULCommand **Command** { get;}

プロパティ値

更新で実行された ULCommand オブジェクト。

備考

これは、System.Data.Common.RowUpdatedEventArgs.Command が厳密に型指定されたものです。

参照

- 「ULRowUpdatedEventArgs クラス」 449 ページ
- 「ULRowUpdatedEventArgs メンバ」 449 ページ
- 「ULCommand クラス」 91 ページ
- DbDataAdapter.Update
- RowUpdatedEventArgs.Command

RecordsAffected プロパティ

SQL 文の実行によって変更、挿入、または削除されたローの数を返します。SELECT 文の場合、この値は -1 です。

構文

Visual Basic
Public Readonly Property **RecordsAffected** As Integer

C#
public int **RecordsAffected** { get;}

プロパティ値

変更、挿入、または削除されたローの数。文が失敗したときにローが影響されなかった場合は 0、SELECT 文の場合は -1。

参照

- 「ULRowUpdatedEventArgs クラス」 449 ページ
- 「ULRowUpdatedEventArgs メンバ」 449 ページ

ULRowUpdatedEventHandler デリゲート

ULDataAdapter.RowUpdated イベントを処理するメソッドを表します。

構文

Visual Basic

```
Public Delegate Sub ULRowUpdatedEventHandler( _
  ByVal sender As Object, _
  ByVal e As ULRowUpdatedEventArgs _
)
```

C#

```
public delegate void ULRowUpdatedEventHandler(
  object sender,
  ULRowUpdatedEventArgs e
);
```

参照

- 「RowUpdated イベント」 252 ページ
- 「ULRowUpdatedEventArgs クラス」 449 ページ

ULRowUpdatingEventArgs クラス

ULDataAdapter.RowUpdating イベントのデータを提供します。このクラスは継承できません。

構文

Visual Basic
Public NotInheritable Class **ULRowUpdatingEventArgs**
  Inherits RowUpdatingEventArgs

C#
public sealed class **ULRowUpdatingEventArgs**: RowUpdatingEventArgs

備考

継承： System.Data.Common.RowUpdatingEventArgs

参照

- 「ULRowUpdatingEventArgs メンバ」 454 ページ
- 「RowUpdating イベント」 253 ページ
- RowUpdatingEventArgs

ULRowUpdatingEventArgs メンバ

パブリック・コンストラクタ

メンバ名	説明
「ULRowUpdatingEventArgs コンストラクタ」 455 ページ	ULRowUpdatingEventArgs クラスの新しいインスタンスを初期化します。

パブリック・プロパティ

メンバ名	説明
「Command プロパティ」 456 ページ	DbDataAdapter.Update(System.Data.DataRow[],System.Data.Common.DataTableMapping) の実行時に実行される ULCommand を指定します。
Errors (RowUpdatingEventArgs から継承)	RowUpdatedEventArgs.Command の実行時に .NET Framework データ・プロバイダによって生成されるエラーを取得します。
Row (RowUpdatingEventArgs から継承)	挿入、更新、または削除操作の一部としてサーバに送信される DataRow を取得します。
StatementType (RowUpdatingEventArgs から継承)	実行する SQL 文のタイプを取得します。

メンバ名	説明
Status (RowUpdatingEventArgs から継承)	RowUpdatedEventArgs.Command の UpdateStatus を取得または設定します。
TableMapping (RowUpdatingEventArgs から継承)	DbDataAdapter.Update によって送信する DataTableMapping を取得します。

参照

- 「ULRowUpdatingEventArgs クラス」 454 ページ
- 「RowUpdating イベント」 253 ページ
- RowUpdatingEventArgs

ULRowUpdatingEventArgs コンストラクタ

ULRowUpdatingEventArgs クラスの新しいインスタンスを初期化します。

構文

Visual Basic

```
Public Sub New( _
  ByVal row As DataRow, _
  ByVal command As IDbCommand, _
  ByVal statementType As StatementType, _
  ByVal tableMapping As DataTableMapping _
)
```

C#

```
public  ULRowUpdatingEventArgs(
  DataRow row,
  IDbCommand command,
  StatementType statementType,
  DataTableMapping tableMapping
);
```

パラメータ

- **row**　更新する System.Data.DataRow。
- **command**　更新時に実行される System.Data.IDbCommand。
- **statementType**　実行されたクエリのタイプを指定する System.Data.StatementType 値の 1 つ。
- **tableMapping**　DbDataAdapter.Update(System.Data.DataRow[],System.Data.Common.DataTableMapping) によって送信された System.Data.Common.DataTableMapping。

参照

- 「ULRowUpdatingEventArgs クラス」 454 ページ
- 「ULRowUpdatingEventArgs メンバ」 454 ページ
- DataRow
- IDbCommand
- StatementType
- DataTableMapping
- DbDataAdapter.Update

Command プロパティ

DbDataAdapter.Update(System.Data.DataRow[],System.Data.Common.DataTableMapping) の実行時に実行される ULCommand を指定します。

構文

Visual Basic
Public Property **Command** As ULCommand

C#
public ULCommand **Command** { get; set; }

プロパティ値

更新時に実行される ULCommand オブジェクト。

備考

これは、System.Data.Common.RowUpdatingEventArgs.Command が厳密に型指定されたものです。

参照

- 「ULRowUpdatingEventArgs クラス」 454 ページ
- 「ULRowUpdatingEventArgs メンバ」 454 ページ
- 「ULCommand クラス」 91 ページ
- DbDataAdapter.Update
- RowUpdatingEventArgs.Command

ULRowUpdatingEventHandler デリゲート

ULDataAdapter.RowUpdating イベントを処理するメソッドを表します。

構文

Visual Basic

```
Public Delegate Sub ULRowUpdatingEventHandler( _
  ByVal sender As Object, _
  ByVal e As ULRowUpdatingEventArgs _
)
```

C#

```
public delegate void ULRowUpdatingEventHandler(
  object sender,
  ULRowUpdatingEventArgs e
);
```

参照

- 「RowUpdating イベント」 253 ページ
- 「ULRowUpdatingEventArgs クラス」 454 ページ

ULRuntimeType 列挙体

UL 拡張：Ultra Light.NET ランタイムのタイプを列挙します。

構文

Visual Basic
Public Enum **ULRuntimeType**

C#
public enum **ULRuntimeType**

メンバ

メンバ名	説明	値
STANDALONE_UL	スタンドアロンの Ultra Light.NET ランタイムを選択します。 スタンドアロンのランタイムは、データベースに直接アクセスします。データベースへのアクセスは高速になりますが、データベースを共有することはできません。	0
UL_ENGINE_CLIENT	Ultra Light エンジンのランタイムを選択します。 Ultra Light.NET エンジン・クライアントは、Ultra Light エンジンと通信してデータベースにアクセスします。つまり、別々のアプリケーションでデータベースを共有できます。	1

参照

- 「RuntimeType プロパティ」 256 ページ

ULServerSyncListener インタフェース

UL 拡張：サーバの同期メッセージを受信するリスナ・インタフェースです。

構文

Visual Basic
Public Interface **ULServerSyncListener**

C#
public interface **ULServerSyncListener**

参照

- 「ULServerSyncListener メンバ」 459 ページ

ULServerSyncListener メンバ

パブリック・メソッド

メンバ名	説明
「ServerSyncInvoked メソッド」 459 ページ	サーバ起動同期用の Mobile Link Listener がアプリケーションを呼び出して同期を実行するときに起動されます。

参照

- 「ULServerSyncListener インタフェース」 459 ページ

ServerSyncInvoked メソッド

サーバ起動同期用の Mobile Link Listener がアプリケーションを呼び出して同期を実行するときに起動されます。

構文

Visual Basic
```
Public Sub ServerSyncInvoked( _
  ByVal messageName As String _
)
```

C#
```
public void ServerSyncInvoked(
  string  messageName
);
```

パラメータ

- **messageName**　アプリケーションに送信されるメッセージの名前。

備考

このメソッドは、別のスレッドによって呼び出されます。マルチスレッドの問題を回避するには、このメソッドがユーザ・インタフェースにイベントを通知する必要があります。マルチスレッドを使用する場合は、スレッドごとに別々の接続を使用し、lock キーワードを使用して、アプリケーションの共有オブジェクトにアクセスすることをおすすめします。

例

次のコード・フラグメントは、サーバ同期要求の受信方法と UI スレッドでの同期の実行方法を示してします。

```
' Visual Basic
Imports iAnywhere.Data.UltraLite

Public Class MainWindow
  Inherits System.Windows.Forms.Form
  Implements ULServerSyncListener
  Private conn As ULConnection

  Public Sub New(ByVal args() As String)

    MyBase.New()

    'This call is required by the Windows Form Designer.
    InitializeComponent()

    'Add any initialization after the InitializeComponent() call
    ULConnection.DatabaseManager.SetServerSyncListener( _
        "myCompany.mymsg", "myCompany.myapp", Me _
      )
    'Create Connection
    ...
  End Sub

  Protected Overrides Sub OnClosing( _
      ByVal e As System.ComponentModel.CancelEventArgs _
    )
    ULConnection.DatabaseManager.SetServerSyncListener( _
      Nothing, Nothing, Nothing _
    )
    MyBase.OnClosing(e)
  End Sub

  Public Sub ServerSyncInvoked(ByVal messageName As String) _
     Implements ULServerSyncListener.ServerSyncInvoked
    Me.Invoke(New EventHandler(AddressOf Me.ServerSyncAction))
  End Sub

  Public Sub ServerSyncAction( _
      ByVal sender As Object, ByVal e As EventArgs _
    )
    ' Do Server sync
    conn.Synchronize()
  End Sub
End Class


// C#
using iAnywhere.Data.UltraLite;
public class Form1 : System.Windows.Forms.Form, ULServerSyncListener
```



```
{
  private System.Windows.Forms.MainMenu mainMenu1;
  private ULConnection conn;

  public Form1()
  {
    //
    // Required for Windows Form Designer support
    //
    InitializeComponent();

    //
    // TODO: Add any constructor code after
    // InitializeComponent call
    //
    ULConnection.DatabaseManager.SetServerSyncListener(
        "myCompnay.mymsg", "myCompany.myapp", this
      );
    // Create connection
    ...
  }

  protected override void Dispose( bool disposing )
  {
    base.Dispose( disposing );
  }

  protected override void OnClosing(
      System.ComponentModel.CancelEventArgs e
    )
  {
    ULConnection.DatabaseManager.SetServerSyncListener(
        null, null, null
      );
    base.OnClosing(e);
  }

  public void ServerSyncInvoked( string messageName )
  {
    this.Invoke( new EventHandler( ServerSyncHandler ) );
  }

  internal void ServerSyncHandler(object sender, EventArgs e)
  {
    conn.Synchronize();
  }
}
```

参照

- 「ULServerSyncListener インタフェース」 459 ページ
- 「ULServerSyncListener メンバ」 459 ページ

ULSQLCode 列挙体

UL 拡張： Ultra Light.NET によってレポートされる可能性のある SQL コードを列挙します。

構文

Visual Basic
Public Enum **ULSQLCode**

C#
public enum **ULSQLCode**

メンバ

メンバ名	説明	値
SQLE_AGGREGATES_NOT_ALLOWED	「集合関数の使用が無効です。」『エラー・メッセージ』を参照してください。	-150
SQLE_ALIAS_NOT_UNIQUE	「エイリアス '%1' がユニークではありません。」『エラー・メッセージ』を参照してください。	-830
SQLE_ALIAS_NOT_YET_DEFINED	「エイリアス '%1' の定義は、最初の参照前に記述する必要があります。」『エラー・メッセージ』を参照してください。	-831
SQLE_AMBIGUOUS_INDEX_NAME	「インデックス名 '%1' があいまいです。」『エラー・メッセージ』を参照してください。	-678
SQLE_ARGUMENT_CANNOT_BE_NULL	「プロシージャ '%2' の引数 %1 に NULL は指定できません。」『エラー・メッセージ』を参照してください。	-90
SQLE_BAD_ENCRYPTION_KEY	「暗号化キーが不正であるか、見つかりません。」『エラー・メッセージ』を参照してください。	-840
SQLE_BAD_PARAM_INDEX	「入力パラメータ・インデックスが範囲外です。」『エラー・メッセージ』を参照してください。	-689
SQLE_CANNOT_ACCESS_FILE	「ファイル '%1' にアクセスできません -- %2」『エラー・メッセージ』を参照してください。	-602
SQLE_CANNOT_ACCESS_FILESYSTEM	「デバイス上のファイルシステムにアクセスできません。」『エラー・メッセージ』を参照してください。	-1108

メンバ名	説明	値
SQLE_CANNOT_ACCESS_SCHEMA_FILE	「スキーマ・ファイル '%1' にアクセスできません。」『エラー・メッセージ』を参照してください。	-951
SQLE_CANNOT_CHANGE_ML_REMOTE_ID	「前回のアップロードのステータスが不明の場合、Mobile Link リモート ID は変更できません。」『エラー・メッセージ』を参照してください。	-1118
SQLE_CANNOT_EXECUTE_STMT	「文を実行することができませんでした。」『エラー・メッセージ』を参照してください。	111
SQLE_CANNOT_MODIFY	「テーブル '%2' のカラム '%1' を変更できません。」『エラー・メッセージ』を参照してください。	-191
SQLE_CANNOT_OPEN_LOG	「トランザクション・ログ・ファイルを開けません -- %1」『エラー・メッセージ』を参照してください。	-106
SQLE_CANNOT_REGISTER_LISTENER	「指定されたリスナを登録できませんでした。」『エラー・メッセージ』を参照してください。	-992
SQLE_CLIENT_OUT_OF_MEMORY	「クライアントでメモリが不足しています。」『エラー・メッセージ』を参照してください。	-876
SQLE_COLUMN_AMBIGUOUS	「カラム '%1' が複数のテーブルで見つかりました。相関名が必要です。」『エラー・メッセージ』を参照してください。	-144
SQLE_COLUMN_CANNOT_BE_NULL	「テーブル '%2' のカラム '%1' を NULL にすることはできません。」『エラー・メッセージ』を参照してください。	-195
SQLE_COLUMN_IN_INDEX	「インデックスのカラムを変更することはできません。」『エラー・メッセージ』を参照してください。	-127
SQLE_COLUMN_NOT_FOUND	「カラム '%1' が見つかりません。」『エラー・メッセージ』を参照してください。	-143
SQLE_COLUMN_NOT_INDEXED	「カラム '%1' は、それを含んでいるテーブルのどのインデックスにも属していません。」『エラー・メッセージ』を参照してください。	-1101

メンバ名	説明	値
SQLE_COLUMN_NOT_STREAMABLE	「操作が失敗しました。カラム '%1' のタイプがストリーミングをサポートしていません。」『エラー・メッセージ』を参照してください。	-1100
SQLE_COMMUNICATIONS_ERROR	「通信エラーが発生しました。」『エラー・メッセージ』を参照してください。	-85
SQLE_CONNECTION_ALREADY_EXISTS	「この接続はすでに存在します。」『エラー・メッセージ』を参照してください。	-955
SQLE_CONNECTION_NOT_FOUND	「接続が見つかりません。」『エラー・メッセージ』を参照してください。	-108
SQLE_CONNECTION_RESTORED	「Ultra Light の接続がリストアされました。」『エラー・メッセージ』を参照してください。	133
SQLE_CONSTRAINT_NOT_FOUND	「制約 '%1' が見つかりません。」『エラー・メッセージ』を参照してください。	-929
SQLE_CONVERSION_ERROR	「値 %1 をデータ型 %2 に変換できません。」『エラー・メッセージ』を参照してください。	-157
SQLE_CORRUPT_PAGE_READ_RETRY	「破損したページ (ページ '%1') の読み込みをリトライしています。」『エラー・メッセージ』を参照してください。	143
SQLE_CORRUPT_ULTRALITE_DATABASE	「データベース・ページの検証は次のコードで失敗しました : %1」『エラー・メッセージ』を参照してください。	-1186
SQLE_CORRUPT_ULTRALITE_INDEX	「テーブル %1、インデックス %2 のインデックスの検証は次のコードで失敗しました : %3」『エラー・メッセージ』を参照してください。	-1185
SQLE_COULD_NOT_FIND_FUNCTION	「ダイナミック・ライブラリ '%2' に '%1' が見つかりませんでした。」『エラー・メッセージ』を参照してください。	-621
SQLE_COULD_NOT_LOAD_LIBRARY	「ダイナミック・ライブラリ '%1' をロードできませんでした。」『エラー・メッセージ』を参照してください。	-620
SQLE_CURSOR_ALREADY_OPEN	「カーソルはすでに開いています。」『エラー・メッセージ』を参照してください。	-172

メンバ名	説明	値
SQLE_CURSOR_NOT_DECLARED	「カーソルが宣言されていません。」 『エラー・メッセージ』を参照してください。	-170
SQLE_CURSOR_NOT_OPEN	「カーソルが開きません。」 『エラー・メッセージ』を参照してください。	-180
SQLE_CURSOR_RESTORED	「Ultra Light のカーソル (あるいは結果セットまたはテーブル) がリストアされました。」 『エラー・メッセージ』を参照してください。	134
SQLE_CURSOROP_NOT_ALLOWED	「不正なカーソル処理をしようとしました。」 『エラー・メッセージ』を参照してください。	-187
SQLE_DATABASE_ERROR	「データベースの内部エラー %1 -- トランザクションはロールバックされました。」 『エラー・メッセージ』を参照してください。	-301
SQLE_DATABASE_NAME_REQUIRED	「サーバを起動するには、データベース名が必要です。」 『エラー・メッセージ』を参照してください。	-87
SQLE_DATABASE_NOT_CREATED	「データベースの作成に失敗しました : %1」 『エラー・メッセージ』を参照してください。	-645
SQLE_DATABASE_STATE_RESTORED	「Ultra Light データベース・ステータスがリストアされました。」 『エラー・メッセージ』を参照してください。	142
SQLE_DATABASE_UPGRADE_WARNING	「自動データベース・アップグレードが適用されました」 『エラー・メッセージ』を参照してください。	149
SQLE_DATATYPE_NOT_ALLOWED	「式にサポートされていないデータ型があります。」 『エラー・メッセージ』を参照してください。	-624
SQLE_DBSPACE_FULL	「DB 領域が最大ファイル・サイズに達しています。」 『エラー・メッセージ』を参照してください。	-604
SQLE_DESCRIBE_NONSELECT	「SELECT 文以外は記述できません。」 『エラー・メッセージ』を参照してください。	-160
SQLE_DEVICE_ERROR	「I/O エラーです。%1 -- トランザクションはロールバックされました。」 『エラー・メッセージ』を参照してください。	-305

メンバ名	説明	値
SQLE_DEVICE_IO_FAILED	「'%1' のファイル I/O に失敗しました。」 『エラー・メッセージ』を参照してください。	-974
SQLE_DIV_ZERO_ERROR	「ゼロで除算しようとしました。」 『エラー・メッセージ』を参照してください。	-628
SQLE_DOWNLOAD_CONFLICT	「既存のローと競合しているため、ダウンロードに失敗しました。」 『エラー・メッセージ』を参照してください。	-839
SQLE_DOWNLOAD_RESTART_FAILED	「ダウンロードをリトライできません。アップロードが完了していません。」 『エラー・メッセージ』を参照してください。	-1102
SQLE_DROP_DATABASE_FAILED	「データベース '%1' の削除に失敗しました。」 『エラー・メッセージ』を参照してください。	-651
SQLE_DUPLICATE_CURSOR_NAME	「カーソル名 '%1' はすでに存在します。」 『エラー・メッセージ』を参照してください。	-683
SQLE_DUPLICATE_FOREIGN_KEY	「テーブル '%2' の外部キー '%1' は、既存の外部キーと重複しています。」 『エラー・メッセージ』を参照してください。	-251
SQLE_DUPLICATE_OPTION	「オプション '%1' が複数回指定されています。」 『エラー・メッセージ』を参照してください。	139
SQLE_DUPLICATE_ROW_FOUND_IN_DOWNLOAD	「同じプライマリ・キーを持つ 2 つのローが、テーブル '%1' 用にダウンロードされています」 『エラー・メッセージ』を参照してください。	145
SQLE_DYNAMIC_MEMORY_EXHAUSTED	「動的メモリが足りません。」 『エラー・メッセージ』を参照してください。	-78
SQLE_ENCRYPTION_INITIALIZATION_FAILED	「暗号化 DLL を初期化できませんでした : '%1'」 『エラー・メッセージ』を参照してください。	-984
SQLE_ENCRYPTION_NOT_ENABLED	「暗号化が有効になっていません。」 『エラー・メッセージ』を参照してください。	-1143
SQLE_ENCRYPTION_NOT_ENABLED_WARNING	「暗号化が有効になっていません。」 『エラー・メッセージ』を参照してください。	140

メンバ名	説明	値
SQLE_ENGINE_ALREADY_RUNNING	「データベース・サーバはすでに起動しています。」『エラー・メッセージ』を参照してください。	-96
SQLE_ERROR	「実行時 SQL エラーです -- %1」『エラー・メッセージ』を参照してください。	-300
SQLE_ERROR_CALLING_FUNCTION	「外部関数の呼び出しのためのリソースを割り付けられませんでした。」『エラー・メッセージ』を参照してください。	-622
SQLE_ERROR_IN_ASSIGNMENT	「割り当てのエラー」『エラー・メッセージ』を参照してください。	-641
SQLE_EVENT_NOT_FOUND	「イベント '%1' が見つかりません。」『エラー・メッセージ』を参照してください。	-771
SQLE_EVENT_NOTIFICATION_QUEUE_FULL	「イベント通知キュー '%1' が満杯なため、通知は破棄されました」『エラー・メッセージ』を参照してください。	146
SQLE_EVENT_NOTIFICATION_QUEUE_NOT_FOUND	「イベント通知キュー '%1' が見つかりません」『エラー・メッセージ』を参照してください。	-1263
SQLE_EVENT_NOTIFICATION_QUEUE_NOT_FOUND_WARN	「イベント通知キュー '%1' に警告は見つかりません」『エラー・メッセージ』を参照してください。	148
SQLE_EVENT_NOTIFICATION_QUEUE_TIMEOUT	「タイムアウトするまでに、キュー '%1' で通知がありませんでした」『エラー・メッセージ』を参照してください。	-1266
SQLE_EVENT_NOTIFICATIONS_LOST	「キュー '%1' でイベント通知が失われました」『エラー・メッセージ』を参照してください。	147
SQLE_EVENT_OBJECT_ALREADY_EXISTS	「'%1' という名前のイベント・オブジェクトはすでに存在します」『エラー・メッセージ』を参照してください。	-1265
SQLE_EVENT_PARAMETER_NOT_FOUND	「イベント・パラメータ '%1' が見つかりません」『エラー・メッセージ』を参照してください。	-1267

メンバ名	説明	値
SQLE_EXPRESSION_ERROR	「'%1' 付近に無効な式があります。」 『エラー・メッセージ』を参照してください。	-156
SQLE_FEATURE_NOT_ENABLED	「呼び出そうとしたメソッドは、お使いのアプリケーションでは使用できません。」 『エラー・メッセージ』を参照してください。	-1092
SQLE_FILE_BAD_DB	「指定されたデータベースを開始できません。'%1' は有効なデータベース・ファイルではありません。」 『エラー・メッセージ』を参照してください。	-1006
SQLE_FILE_IN_USE	「指定されたデータベース・ファイルはすでに使用されています。」 『エラー・メッセージ』を参照してください。	-816
SQLE_FILE_NOT_DB	「指定されたデータベースを開始できません。'%1' はデータベースではありません。」 『エラー・メッセージ』を参照してください。	-1004
SQLE_FILE_VOLUME_NOT_FOUND	「データベース '%1' に対して指定したファイルシステム・ボリュームが見つかりません。」 『エラー・メッセージ』を参照してください。	-1112
SQLE_FILE_WRONG_VERSION	「指定されたデータベースを開始できません。'%1' は異なるバージョンのソフトウェアで作成されています。」 『エラー・メッセージ』を参照してください。	-1005
SQLE_FOREIGN_KEY_NAME_NOT_FOUND	「外部キー '%1' は見つかりません。」 『エラー・メッセージ』を参照してください。	-145
SQLE_IDENTIFIER_TOO_LONG	「識別子 '%1' が長すぎます。」 『エラー・メッセージ』を参照してください。	-250
SQLE_INCORRECT_VOLUME_ID	「'%1' のボリューム ID が不正です。」 『エラー・メッセージ』を参照してください。	-975
SQLE_INDEX_NAME_NOT_UNIQUE	「インデックス名 '%1' はユニークではありません。」 『エラー・メッセージ』を参照してください。	-111
SQLE_INDEX_NOT_FOUND	「インデックス '%1' が見つかりません。」 『エラー・メッセージ』を参照してください。	-183

メンバ名	説明	値
SQLE_INDEX_NOT_UNIQUE	「テーブル '%2' のインデックス '%1' はユニークでなければなりません。」『エラー・メッセージ』を参照してください。	-196
SQLE_INTERRUPTED	「文の実行がユーザによって中断させられました。」『エラー・メッセージ』を参照してください。	-299
SQLE_INVALID_CONSTRAINT_REF	「制約 '%1' への参照または操作が無効です。」『エラー・メッセージ』を参照してください。	-937
SQLE_INVALID_DESCRIPTOR_INDEX	「記述子のインデックスが正しくありません。」『エラー・メッセージ』を参照してください。	-640
SQLE_INVALID_DESCRIPTOR_NAME	「SQL 記述子名が正しくありません。」『エラー・メッセージ』を参照してください。	-642
SQLE_INVALID_DISTINCT_AGGREGATE	「グループ化されたクエリに、複数の異なる集合関数が含まれています。」『エラー・メッセージ』を参照してください。	-863
SQLE_INVALID_EVENT_OBJECT_NAME	「イベント・オブジェクト名 '%1' は無効です」『エラー・メッセージ』を参照してください。	-1264
SQLE_INVALID_FOREIGN_KEY	「テーブル '%2' の外部キー '%1' に対応するプライマリ・キーの値がありません。」『エラー・メッセージ』を参照してください。	-194
SQLE_INVALID_FOREIGN_KEY_DEF	「外部キーのカラム '%1' にプライマリ・キーと異なる定義があります。」『エラー・メッセージ』を参照してください。	-113
SQLE_INVALID_GROUP_SELECT	「'%1' に対する関数またはカラムの参照も GROUP BY 句に記述する必要があります。」『エラー・メッセージ』を参照してください。	-149
SQLE_INVALID_INDEX_TYPE	「'%1' のインデックスの型の指定は不正です。」『エラー・メッセージ』を参照してください。	-650
SQLE_INVALID_LOGON	「ユーザ ID またはパスワードが無効です。」『エラー・メッセージ』を参照してください。	-103

メンバ名	説明	値
SQLE_INVALID_OPTION	「オプション '%1' が無効です -- PUBLIC 設定がありません。」『エラー・メッセージ』を参照してください。	-200
SQLE_INVALID_OPTION_SETTING	「オプション '%1' の設定が無効です。」『エラー・メッセージ』を参照してください。	-201
SQLE_INVALID_OPTION_VALUE	「'%1' は '%2' に対して無効な値です。」『エラー・メッセージ』を参照してください。	-1053
SQLE_INVALID_ORDER	「ORDER BY 句の指定が不正です。」『エラー・メッセージ』を参照してください。	-152
SQLE_INVALID_PARAMETER	「不正なパラメータです。」『エラー・メッセージ』を参照してください。	-735
SQLE_INVALID_PARSE_PARAMETER	「解析エラー : %1」『エラー・メッセージ』を参照してください。	-95
SQLE_INVALID_SQL_IDENTIFIER	「SQL の識別子が無効です。」『エラー・メッセージ』を参照してください。	-760
SQLE_INVALID_STATEMENT	「文が無効です。」『エラー・メッセージ』を参照してください。	-130
SQLE_INVALID_UNION	「UNION、INTERSECT、または EXCEPT の select リストの長さが一致していません。」『エラー・メッセージ』を参照してください。	-153
SQLE_KEYLESS_ENCRYPTION	「このデータベースはキー不使用の暗号化を使用するため、要求された操作を実行できません。」『エラー・メッセージ』を参照してください。	-1109
SQLE_LOCKED	「'%2' のローは、ユーザ '%1' によってロックされています。」『エラー・メッセージ』を参照してください。	-210
SQLE_MAX_ROW_SIZE_EXCEEDED	「テーブル '%1' の最大ロー・サイズを超過します。」『エラー・メッセージ』を参照してください。	-1132
SQLE_MEMORY_ERROR	「メモリ・エラー -- トランザクションはロールバックされました。」『エラー・メッセージ』を参照してください。	-309

メンバ名	説明	値
SQLE_METHOD_CANNOT_BE_CALLED	「メソッド '%1' は現時点では呼び出せません。」『エラー・メッセージ』を参照してください。	-669
SQLE_MIRROR_FILE_MISMATCH	「ミラー '%1' がデータベース '%2' と一致しません。」『エラー・メッセージ』を参照してください。	-1138
SQLE_MIRROR_FILE_REQUIRES_CHECKSUMS	「ミラー・ファイルには、これより高い checksum_level が必要です。」『エラー・メッセージ』を参照してください。	144
SQLE_MOBILINK_COMMUNICATIONS_ERROR	「Mobile Link 通信エラー : コード : %1、パラメータ : %2、システム・コード %3」『エラー・メッセージ』を参照してください。	-1305
SQLE_NAME_NOT_UNIQUE	「アイテム '%1' はすでに存在しています。」『エラー・メッセージ』を参照してください。	-110
SQLE_NO_COLUMN_NAME	「派生テーブル '%1' にはカラム %2 に対する名前がありません。」『エラー・メッセージ』を参照してください。	-163
SQLE_NO_CURRENT_ROW	「カーソルの現在のローがありません。」『エラー・メッセージ』を参照してください。	-197
SQLE_NO_INDICATOR	「NULL に対して、インジケータ変数が用意されていません。」『エラー・メッセージ』を参照してください。	-181
SQLE_NO_MATCHING_SELECT_ITEM	「派生テーブル '%1' の select リストに '%2' と一致する式がありません。」『エラー・メッセージ』を参照してください。	-812
SQLE_NO_PRIMARY_KEY	「テーブル '%1' にはプライマリ・キーが定義されていません。」『エラー・メッセージ』を参照してください。	-118
SQLE_NOERROR	SQLE_NOERROR(0) – このコードは、エラーまたは警告がなかったことを示します。	0
SQLE_NON_UPDATEABLE_COLUMN	「式を更新できません。」『エラー・メッセージ』を参照してください。	-190

メンバ名	説明	値
SQLE_NON_UPDATEABLE_CURSOR	「FOR UPDATE が READ ONLY カーソルに誤って指定されました。」『エラー・メッセージ』を参照してください。	-813
SQLE_NOT_IMPLEMENTED	「'%1' の機能は実装されていません。」『エラー・メッセージ』を参照してください。	-134
SQLE_NOT_SUPPORTED_IN_ULTRALITE	「Ultra Light では使用できない機能です。」『エラー・メッセージ』を参照してください。	-749
SQLE_NOTFOUND	「ローが見つかりません。」『エラー・メッセージ』を参照してください。	100
SQLE_ONLY_ONE_TABLE	「カーソルの INSERT/DELETE は、1 つのテーブルしか変更できません。」『エラー・メッセージ』を参照してください。	-199
SQLE_OVERFLOW_ERROR	「値 %1 は、対象先にとって大きすぎます。」『エラー・メッセージ』を参照してください。	-158
SQLE_PAGE_SIZE_INVALID	「無効なデータベース・ページ・サイズです。」『エラー・メッセージ』を参照してください。	-644
SQLE_PARTIAL_DOWNLOAD_NOT_FOUND	「部分ダウンロードが見つかりませんでした。」『エラー・メッセージ』を参照してください。	-1103
SQLE_PASSTHROUGH_SQL_SCRIPT_FAILED_E	「パススルー SQL スクリプトに失敗しました」『エラー・メッセージ』を参照してください。	-1238
SQLE_PASSTHROUGH_SQL_SCRIPT_FAILED_W	「パススルー SQL スクリプトに失敗しました」『エラー・メッセージ』を参照してください。	141
SQLE_PERMISSION_DENIED	「パーミッションがありません : %1」『エラー・メッセージ』を参照してください。	-121
SQLE_PRIMARY_KEY_NOT_UNIQUE	「テーブル '%1' のプライマリ・キーがユニークではありません : プライマリ・キー値 ('%2')」『エラー・メッセージ』を参照してください。	-193

メンバ名	説明	値
SQLE_PRIMARY_KEY_TWICE	「テーブルに 2 つのプライマリ・キーを定義することはできません。」『エラー・メッセージ』を参照してください。	-126
SQLE_PRIMARY_KEY_VALUE_REF	「テーブル '%1' 内のローのプライマリ・キーがテーブル '%3' 内の外部キー '%2' によって参照されています。」『エラー・メッセージ』を参照してください。	-198
SQLE_PUBLICATION_NOT_FOUND	「パブリケーション '%1' が見つかりません。」『エラー・メッセージ』を参照してください。	-280
SQLE_PUBLICATION_PREDICATE_IGNORED	「パブリケーションの述部は評価されませんでした。」『エラー・メッセージ』を参照してください。	138
SQLE_RESOURCE_GOVERNOR_EXCEEDED	「%1 のリソース・ガバナーが制限を超えています。」『エラー・メッセージ』を参照してください。	-685
SQLE_ROW_DELETED_TO_MAINTAIN_REFERENTIAL_INTEGRITY	「参照整合性を保つためにテーブル %1 からローが削除されました。」『エラー・メッセージ』を参照してください。	137
SQLE_ROW_DROPPED_DURING_SCHEMA_UPGRADE	「ローが新しいスキーマ・フォーマットに変換されなかったため、ローは削除されました。」『エラー・メッセージ』を参照してください。	130
SQLE_ROW_EXCEEDS_PAGE_SIZE	「ローがデータベースのページ・サイズを超えているため、ローを格納できません。」『エラー・メッセージ』を参照してください。	-1117
SQLE_ROW_UPDATED_SINCE_READ	「最後に読み込まれた後で、ローは更新されています。操作はキャンセルされました。」『エラー・メッセージ』を参照してください。	-208
SQLE_SCHEMA_UPGRADE_NOT_ALLOWED	「スキーマのアップグレードは現在有効ではありません。」『エラー・メッセージ』を参照してください。	-953
SQLE_SERVER_SYNCHRONIZATION_ERROR	「サーバ %1 でエラーが発生したため、同期に失敗しました。」『エラー・メッセージ』を参照してください。	-857

メンバ名	説明	値
SQLE_START_STOP_DATABASE_DENIED	「データベースの起動／停止の要求は拒否されました。」『エラー・メッセージ』を参照してください。	-75
SQLE_STATEMENT_ERROR	「SQL 文にエラーがあります。」『エラー・メッセージ』を参照してください。	-132
SQLE_STRING_RIGHT_TRUNCATION	「文字列データの右側がトランケートされます。」『エラー・メッセージ』を参照してください。	-638
SQLE_SUBQUERY_RESULT_NOT_UNIQUE	「サブクエリは複数行を返すことはできません。」『エラー・メッセージ』を参照してください。	-186
SQLE_SUBQUERY_SELECT_LIST	「select リストの中にカラムが 2 つ以上指定されています。」『エラー・メッセージ』を参照してください。	-151
SQLE_SYNC_INFO_INVALID	「同期の情報が不完全か無効です。'%1' を確認してください。」『エラー・メッセージ』を参照してください。	-956
SQLE_SYNC_INFO_REQUIRED	「同期の情報が指定されていません。」『エラー・メッセージ』を参照してください。	-1111
SQLE_SYNC_NOT_REENTRANT	「同期処理に戻ることができませんでした。」『エラー・メッセージ』を参照してください。	-1110
SQLE_SYNC_PROFILE_INVALID	「同期プロファイル '%1' に無効なパラメータ '%2' があります」『エラー・メッセージ』を参照してください。	-1224
SQLE_SYNC_PROFILE_NOT_FOUND	「同期プロファイル '%1' が見つかりません」『エラー・メッセージ』を参照してください。	-1217
SQLE_SYNC_STATUS_UNKNOWN	「最後の同期アップロードのステータスは不明です。」『エラー・メッセージ』を参照してください。	-952
SQLE_SYNTAX_ERROR	「'%1' %2 の近くに構文エラーがあります。」『エラー・メッセージ』を参照してください。	-131
SQLE_TABLE_ALREADY_INCLUDED	「テーブル '%1' はすでにインクルードされています。」『エラー・メッセージ』を参照してください。	-822

メンバ名	説明	値
SQLE_TABLE_HAS_PUBLICATIONS	「テーブル '%1' はパブリケーションを持っています。」『エラー・メッセージ』を参照してください。	-281
SQLE_TABLE_IN_USE	「テーブルは使用されています。」『エラー・メッセージ』を参照してください。	-214
SQLE_TABLE_NOT_FOUND	「テーブル '%1' が見つかりません。」『エラー・メッセージ』を参照してください。	-141
SQLE_TOO_MANY_BLOB_REFS	「BLOB への参照が多すぎます。」『エラー・メッセージ』を参照してください。	-1107
SQLE_TOO_MANY_CONNECTIONS	「データベース・サーバに接続できる限界数を超えています。」『エラー・メッセージ』を参照してください。	-102
SQLE_TOO_MANY_CURSORS	「オープン・カーソルが多すぎます」『エラー・メッセージ』を参照してください。	-1230
SQLE_TOO_MANY_PUBLICATIONS	「オペレーションに指定されているパブリケーションが多すぎます」『エラー・メッセージ』を参照してください。	-1106
SQLE_TOO_MANY_TEMP_TABLES	「接続しているテンポラリ・テーブルが多すぎます。」『エラー・メッセージ』を参照してください。	-817
SQLE_TOO_MANY_USERS	「データベースのユーザが多すぎます。」『エラー・メッセージ』を参照してください。	-1104
SQLE_TRUNCATED	「値がトランケートされました。」『エラー・メッセージ』を参照してください。	101
SQLE_ULTRALITE_DATABASE_NOT_FOUND	「データベース '%1' が見つかりませんでした。」『エラー・メッセージ』を参照してください。	-954
SQLE_ULTRALITE_OBJ_CLOSED	「閉じられたオブジェクトに対する操作は無効です。」『エラー・メッセージ』を参照してください。	-908
SQLE_UNABLE_TO_CONNECT	「データベースが起動できません -- %1」『エラー・メッセージ』を参照してください。	-105

メンバ名	説明	値
SQLE_UNABLE_TO_START_DATABASE	「指定されたデータベースを起動できません : %1」『エラー・メッセージ』を参照してください。	-82
SQLE_UNABLE_TO_START_DATABASE_VER_NEWER	「指定されたデータベースを起動できません : データベース %1 を起動するにはサーバをアップグレードする必要があります。」『エラー・メッセージ』を参照してください。	-934
SQLE_UNKNOWN_FUNC	「関数 '%1' はありません。」『エラー・メッセージ』を参照してください。	-148
SQLE_UNKNOWN_OPTION	「'%1' は認識できないオプションです。」『エラー・メッセージ』を参照してください。	120
SQLE_UNKNOWN_USERID	「'%1' というユーザ ID はありません。」『エラー・メッセージ』を参照してください。	-140
SQLE_UNRECOGNIZED_OPTION	「オプション '%1' が認識されません。」『エラー・メッセージ』を参照してください。	-1002
SQLE_UPLOAD_FAILED_AT_SERVER	「同期サーバがアップロードのコミットに失敗しました。」『エラー・メッセージ』を参照してください。	-794
SQLE_VALUE_IS_NULL	「要求されたデータ型として NULL の結果を返すことができません。」『エラー・メッセージ』を参照してください。	-1050
SQLE_VARIABLE_INVALID	「無効なホスト変数です。」『エラー・メッセージ』を参照してください。	-155
SQLE_WRONG_NUM_OF_INSERT_COLS	「INSERT 文に指定した値の数が正しくありません。」『エラー・メッセージ』を参照してください。	-207
SQLE_WRONG_PARAMETER_COUNT	「関数 '%1' のパラメータ数が誤りです。」『エラー・メッセージ』を参照してください。	-154

ULSqlPassthroughProgressListener インタフェース

UL 拡張 : SQL パススルー・スクリプトのプログレス・イベントを受信するリスナ・インタフェースです。

構文

Visual Basic
Public Interface **ULSqlPassthroughProgressListener**

C#
public interface **ULSqlPassthroughProgressListener**

参照

- 「ULSqlPassthroughProgressListener メンバ」 477 ページ

ULSqlPassthroughProgressListener メンバ

パブリック・メソッド

メンバ名	説明
「ScriptProgressed メソッド」 477 ページ	SQL パススルー・スクリプトが実行されると呼び出され、進行状況をアプリケーションに通知します。このメソッドは、スクリプトの実行をキャンセルする場合は true を、続行する場合は false を返す必要があります。

参照

- 「ULSqlPassthroughProgressListener インタフェース」 477 ページ

ScriptProgressed メソッド

SQL パススルー・スクリプトが実行されると呼び出され、進行状況をアプリケーションに通知します。このメソッドは、スクリプトの実行をキャンセルする場合は true を、続行する場合は false を返す必要があります。

構文

Visual Basic
```
Public Function ScriptProgressed( _
  ByVal data As ULSqlProgressData _
) As Boolean
```

C#
```
public bool ScriptProgressed(
  ULSqlProgressData data
);
```

パラメータ

- **data**　現在のスクリプトに関する情報を保持している ULSqlProgressData オブジェクト。

戻り値

このメソッドは、スクリプト処理をキャンセルする場合は true を、続行する場合は false を返す必要があります。

備考

ScriptProgressed の呼び出し中に、Ultra Light.NET API のメソッドを呼び出さないでください。

参照

- 「ULSqlPassthroughProgressListener インタフェース」 477 ページ
- 「ULSqlPassthroughProgressListener メンバ」 477 ページ
- 「ULSqlProgressData クラス」 479 ページ

ULSqlProgressData クラス

UL 拡張：SQL パススルー・スクリプトの進行状況のモニタリング・データを返します。

構文

Visual Basic
Public Class **ULSqlProgressData**

C#
public class **ULSqlProgressData**

参照

- 「ULSqlProgressData メンバ」 479 ページ
- 「ULSqlPassthroughProgressListener インタフェース」 477 ページ

ULSqlProgressData メンバ

パブリック・プロパティ

メンバ名	説明
「CurrentScript プロパティ」 479 ページ	これまでに実行されたスクリプトのインデックス。
「ScriptCount プロパティ」 480 ページ	実行中のスクリプトの数を返します。
「State プロパティ」 480 ページ	現在の進行状況のステータスを返します。

参照

- 「ULSqlProgressData クラス」 479 ページ
- 「ULSqlPassthroughProgressListener インタフェース」 477 ページ

CurrentScript プロパティ

これまでに実行されたスクリプトのインデックス。

構文

Visual Basic
Public Readonly Property **CurrentScript** As Long

C#
public long **CurrentScript** { get;}

プロパティ値

実行中のスクリプトの現在のインデックス。

参照

- 「ULSqlProgressData クラス」 479 ページ
- 「ULSqlProgressData メンバ」 479 ページ

ScriptCount プロパティ

実行中のスクリプトの数を返します。

構文

Visual Basic
Public Readonly Property **ScriptCount** As Long

C#
public long **ScriptCount** { get;}

プロパティ値

実行中のスクリプトの数。

参照

- 「ULSqlProgressData クラス」 479 ページ
- 「ULSqlProgressData メンバ」 479 ページ

State プロパティ

現在の進行状況のステータスを返します。

構文

Visual Basic
Public Readonly Property **State** As ULSqlProgressState

C#
public ULSqlProgressState **State** { get;}

プロパティ値

現在の SQL パススルーのコールバック・ステータスを指定する ULSqlProgressState 値の 1 つ。

参照

- 「ULSqlProgressData クラス」 479 ページ
- 「ULSqlProgressData メンバ」 479 ページ
- 「ULSyncProgressState 列挙体」 540 ページ

ULSqlProgressState 列挙体

UL 拡張：SQL パススルー・スクリプトの実行中に発生する可能性のあるすべてのステータスを列挙します。

構文

Visual Basic
Public Enum **ULSqlProgressState**

C#
public enum **ULSqlProgressState**

メンバ

メンバ名	説明	値
STATE_DONE	スクリプトは正常に完了しました。	2
STATE_ERROR	スクリプトは完了しましたが、エラーが発生しました。	3
STATE_RUNNING_SCRIPT	SQL パススルー・スクリプトを現在実行中です。	1
STATE_STARTING	実行されたスクリプトはまだありません。	0

参照

- 「ULSqlProgressData クラス」 479 ページ

ULStreamErrorCode 列挙体

UL 拡張：同期中にストリームによってレポートされる可能性のあるエラー・コードを列挙します。

構文

Visual Basic
Public Enum **ULStreamErrorCode**

C#
public enum **ULStreamErrorCode**

メンバ

メンバ名	説明	値
ACTSYNC_NO_PORT	ActiveSync 同期を開始できるのは、ActiveSync だけです。開始するには、クレードル内にデバイスを置くか、ActiveSync マネージャの [同期] を選択します。アプリケーションから同期を開始するには、TCP/IP ソケットの同期ストリームを使用してください。	75
ACTSYNC_NOT_INSTALLED	ActiveSync プロバイダがインストールされていません。mlasinst を実行して ActiveSync プロバイダをインストールしてください (詳細についてはマニュアルを参照してください)。	76
AUTHENTICATION_FAILED	クライアントが Mobile Link への認証に失敗しました。	249
CONNECT_TIMEOUT	接続試行がタイムアウトになりました。サーバが指定のホストとポートで実行されていないか、タイムアウト値を大きくして接続の試行に使用できる時間を増やす必要があります。	241
COULD_NOT_OPEN_FILE	指定されたファイルを開けませんでした。	264
COULD_NOT_OPEN_FILE_FOR_WRITE	指定されたファイルを書き込み用に開くことができませんでした。このファイルが正しいファイルであること、またファイルが他のアプリケーションで使用されていないことを確認してください。	230

メンバ名	説明	値
CREATE_RANDOM_OBJECT	セキュア・ネットワーク・レイヤが、乱数生成オブジェクトを作成できませんでした。システム・リソースを解放してから、もう一度接続と操作を行ってください。	17
DEQUEUING_CONNECTION	キューイングされている接続 (同期) 要求を取得しようとしているときに、Mobile Link サーバでエラーが発生しました。システム・リソースを解放してください。問題が解決しなければ、Mobile Link サーバを再起動してください。	19
DUN_DIAL_FAILED	指定されたダイヤルアップ・ネットワークに自動ダイヤルアップで接続を確立できませんでした。	204
DUN_NOT_SUPPORTED	システム・サポートが十分でないため、ダイヤルアップに失敗しました。PocketPC では cellcore.dll を使用し、Windows では IE 4.0 以降の wininet.dll を使用する必要があります。他のプラットフォームではダイヤルアップはサポートされていません。	203
E2EE_DECODING_PRIVATE_KEY_FILE	ファイルが見つかり、内容を読み込みましたが、ファイルの復号化でエラーが発生しました。弊社製品の保守契約を結んでいるサポート・センタに連絡して、エラー・コードを報告してください。	257
E2EE_INIT_ECC	ECC を初期化しようとしてエラーが発生しました。ECC オプションがインストールされていることを確認してください。	262
E2EE_INVALID_TYPE	無効な e2ee_type が指定されました。有効な値を指定してください。	261
E2EE_MISMATCHED_KEYS	リモートでのエンドツーエンド暗号化に使用された e2ee_public_key がサーバでの e2ee_private_key に一致しないため、クライアントとサーバが通信できません。	253

メンバ名	説明	値
E2EE_MISSING_PRIVATE_KEY	別のエンドツーエンド暗号化オプションが指定されましたが、e2ee_private_key オプションではありません。すべてのエンドツーエンド暗号化オプションを指定するか、削除してください。必要なエンドツーエンド暗号化オプションには、e2ee_type、e2ee_private_key、e2ee_private_key_password があります。	260
E2EE_MISSING_PRIVATE_KEY_PASSWORD	e2ee_private_key ファイルは、e2ee_private_key_password がないと読み込めません。e2ee_private_key_password を入力してください。	259
E2EE_NO_PRIVATE_KEY_IN_FILE	指定されたファイル名にプライベート・キーが含まれていません。	256
E2EE_PUBLIC_KEY	エンドツーエンド暗号化のパブリック・キーを読み込もうとして、エラーが発生しました。	263
E2EE_READING_PRIVATE_KEY	e2ee_private_key ファイルの読み込み中にエラーが発生しました。弊社製品の保守契約を結んでいるサポート・センタに連絡して、エラー・コードを報告してください。	255
E2EE_READING_PRIVATE_KEY_FILE	指定されたファイルが読み込めませんでした。弊社製品の保守契約を結んでいるサポート・センタに連絡して、エラー・コードを報告してください。	258
E2EE_UNEXPECTED_PRIVATE_KEY_TYPE	e2ee_private_key ファイルで見つかったプライベート・キー・タイプは、e2ee_type で指定されたタイプと一致しません。	254
E2EE_UNEXPECTED_PUBLIC_KEY_ENC_TYPE	クライアントは、サーバで指定された e2ee_type とは異なる e2ee_type 値を送信しました。両者が同じであることを確認してください。	252

メンバ名	説明	値
E2EE_UNKNOWN_PUBLIC_KEY_ENC_TYPE	クライアントは、サーバで認識されない e2ee_type 値を送信しました。サーバのバージョンがリモートのバージョンと同じかそれ以上であることを確認してください。	251
END_READ	ネットワークからの一連の読み込みを完了できません。参照：READ	11
END_WRITE	ネットワークへの一連の書き込みを完了できません。参照：WRITE	10
GENERATE_RANDOM	セキュア・ネットワーク・レイヤでは乱数が必要ですが、それを生成できませんでした。システム・リソースを解放してから、もう一度接続と操作を行ってください。	14
HTTP_AUTHENTICATION_FAILED	入力したユーザ ID とパスワードは拒否されました。正しく入力したかどうか確認してください。入力した場合は、正しいアクセス権があることをシステム管理者に確認してください。	211
HTTP_AUTHENTICATION_REQUIRED	HTTP サーバまたはゲートウェイが HTTP 認証を要求しました。HTTP 同期パラメータ http_userid と http_password を使用してユーザ ID とパスワードを指定してください。	209
HTTP_BAD_STATUS_CODE	ステータス行を調べて失敗の原因を判別してください。	86
HTTP_BUFFER_SIZE_OUT_OF_RANGE	HTTP バッファ・サイズを修正してください。有効なバッファ・サイズは正の値で、ホスト・プラットフォームにとって大きすぎないように指定してください。	79
HTTP_CHUNK_LEN_BAD_CHARACTER	固定長の HTTP 本文を使用してください。	85
HTTP_CHUNK_LEN_ENCODED_MISSING	固定長の HTTP 本文を使用してください。	84

メンバ名	説明	値
HTTP_CLIENT_ID_NOT_SET	クライアント ID が HTTP クライアント・コードに渡されませんでした。修正プログラムについては、弊社製品の保守契約を結んでいるサポート・センタにお問い合わせください。	78
HTTP_CONTENT_TYPE_NOT_SPECIFIED	未知の内容タイプが指定されました。マニュアルを参照し、サポートされる内容タイプに変更してください。	77
HTTP_CRLF_ENCODED_MISSING	使用しているプロキシが Mobile Link と互換性がない可能性があります。設定を確認してください。	81
HTTP_CRLF_MISSING	使用しているプロキシが Mobile Link と互換性がない可能性があります。設定を確認してください。	82
HTTP_EXPECTED_POST	使用しているプロキシが Mobile Link と互換性がない可能性があります。設定を確認してください。	89
HTTP_EXTRA_DATA_END_READ	HTTP 本文に余分なデータが追加されています。このデータは、プロキシ・エージェントによって追加された可能性があります。プロキシを削除してください。	80
HTTP_HEADER_PARSE_ERROR	HTTP ヘッダを解析しようとしてエラーが発生しました。ヘッダの形式が間違っている可能性があります。	216
HTTP_INTERNAL_HEADER_STATE	HTTP ヘッダを復号化できませんでした。これは通常は起こるはずのない内部エラーです。弊社製品の保守契約を結んでいるサポート・センタに問い合わせてください。	236
HTTP_INTERNAL_REQUEST_TYPE	HTTP 要求タイプを判別できませんでした。これは通常は起こるはずのない内部エラーです。弊社製品の保守契約を結んでいるサポート・センタに問い合わせてください。	237

メンバ名	説明	値
HTTP_INVALID_CHARACTER	HTTP ヘッダで予期しない文字が読み込まれました。ヘッダの形式が間違っているか、通信先から HTTP が送信されていない可能性があります。	219
HTTP_INVALID_SESSION_KEY	未知のセッション・キー・タイプが指定されました。マニュアルを参照し、サポートされるセッション・キー・タイプに変更してください。	244
HTTP_MALFORMED_SESSION_COOKIE	同期セッションの管理用の HTTP cookie が破損しています。cookie が破損している箇所を特定してください。クライアント・エラーか、HTTP の中間が正しく動作していないことが主な原因と考えられます。	235
HTTP_NO_CONTD_CONNECTION	リモート・サイトから送信される次の HTTP 要求の待機中に、サーバがタイムアウトになりました。この要求がサーバに届かなかった原因を判別するか、永続的な接続を試みてください。	83
HTTP_NO_PASSWORD	HTTP 認証にユーザ ID が入力されましたが、パスワードが入力されていません。認証には両方が必要です。	214
HTTP_NO_USERID	HTTP 認証にパスワードが入力されましたが、ユーザ ID が入力されていません。認証には両方が必要です。	213
HTTP_PROXY_AUTHENTICATION_FAILED	入力したユーザ ID とパスワードがプロキシ・サーバで拒否されました。正しく入力したかどうか確認してください。入力した場合は、正しいアクセス権があることをシステム管理者に確認してください。	212
HTTP_PROXY_AUTHENTICATION_REQUIRED	HTTP プロキシが HTTP 認証を要求しました。HTTP 同期パラメータ http_proxy_userid と http_proxy_password を使用してユーザ ID とパスワードを指定してください。	210

メンバ名	説明	値
HTTP_SERVER_AUTH_FAILED	サーバから送信された認証情報ヘッダに不正な値が含まれているため、認証に失敗しました。正規の HTTP サーバに接続していることを確認してください。	217
HTTP_UNABLE_TO_PARSE_COOKIE	破損している set cookie ヘッダを判別してください。	88
HTTP_UNKNOWN_TRANSFER_ENCODING	未知の転送エンコードがどのように生成されているかを判別してください。	87
HTTP_UNSUPPORTED_AUTH_ALGORITHM	サーバが要求した HTTP ダイジェスト認証アルゴリズムは、サポートされていません。サポートされているのは「MD5」と「MD5-sess」だけです。	215
HTTP_VERSION	要求された HTTP バージョンはサポートされていません。マニュアルを調べて、サポート対象の HTTP バージョンを指定してください。パブリケーション時のサポート対象 HTTP バージョンは 1.0 と 1.1 です。	54
INCONSISTENT_FIPS	Mobile Link サーバのコマンド・ラインで -fips スイッチを使用するには、すべてのセキュア・ストリームが FIPS に準拠している必要があります。セキュア・ストリームに fips オプションが設定されていない場合は、自動的に FIPS 準拠 (fips=y など) になります。セキュア・ストリームから fips オプションを削除するか、fips=y で有効にしてください。	242
INIT_RANDOM	セキュア・ネットワーク・レイヤが、乱数ジェネレータを初期化できませんでした。システム・リソースを解放してから、もう一度接続と操作を行ってください。	15
INTERNAL	ネットワーク・レイヤで内部エラーが発生しました。弊社製品の保守契約を結んでいるサポート・センタに問い合わせてください。	220

メンバ名	説明	値
INTERNAL_API	ネットワーク・レイヤで内部エラーが発生しました。弊社製品の保守契約を結んでいるサポート・センタに問い合わせてください。	238
INTERNAL_PROTOCOL_NOT_LOADED	同期プロトコルをロードできませんでした。Ultra Light を使用している場合は、適切な ULEnable メソッドを呼び出していることを確認してください。	227
INTERRUPTED	現在の操作は呼び出し元によって中断されました。	218
INVALID_COMPRESSION_TYPE	指定された圧縮型が認識されませんでした。	232
INVALID_LOCAL_PATH	ダウンロードされたファイルのローカル・パスが無効です。詳細については、マニュアルを参照してください。	243
INVALID_NETWORK_LIBRARY	指定されたネットワーク・インタフェース DLL または共有オブジェクトがロードできませんでした。不正か破損している可能性があります。	245
INVALID_SYNC_PROTOCOL	指定されたプロトコルは有効な同期プロトコルではありません。	226
LIBRARY_ENTRY_POINT_NOT_FOUND	指定されたライブラリのエントリ・ポイントが見つかりませんでした。	225
LOAD_LIBRARY_FAILURE	パスでダイナミック・ライブラリを見つけることができませんでした。同期に TLS 暗号化を使用しようとしている場合は、適切なライセンスを取得していることを確認してください。	224

メンバ名	説明	値
LOAD_NETWORK_LIBRARY	ネットワーク・インタフェース・ライブラリを検出してロードできませんでした (またはそのいずれか)。次の項目を確認してください。 1) ソケット・レイヤが適切にインストールされている。適切なネットワーク・インタフェース・ライブラリ (または DLL か共有オブジェクト) が存在しており、アクセス可能になっていなければなりません。 2) 使用可能なシステム・リソースが十分にある。不足していれば、システム・リソースを解放してください。	73
MEMORY_ALLOCATION	ネットワーク・レイヤで十分なバイト数の記憶領域を割り付けることができませんでした。システム・メモリを解放してもう一度操作を行ってください。システム・メモリを解放する方法は、オペレーティング・システムとその設定方法によって異なります。最も簡単な方法は、アクティブな処理の数を減らすことです。詳細については、使用しているオペレーティング・システムのマニュアルを参照してください。	6
MISSING_PARAMETER	必要なパラメータが指定されていません。	229
NETWORK_LIBRARY_VERSION_MISMATCH	バージョンが間違っているため、ネットワーク・インタフェース DLL または共有オブジェクトがロードできませんでした。	248
NO_ECC_FIPS	この圧縮処理を行うことができませんでした。弊社製品の保守契約を結んでいるサポート・センタに問い合わせてください。	239
NONE	このコードは、ネットワーク・エラーが発生しなかったか、未知のネットワーク・エラーが発生したことを示します。	0

メンバ名	説明	値
NOT_IMPLEMENTED	実装されていない内部機能が要求されました。弊社製品の保守契約を結んでいるサポート・センタに問い合わせてください。	12
PARAMETER	ネットワーク・パラメータの形式は、"name=value;[name2=value2[;...]]" です。このコードは、不正なパラメータ値を示しています。該当するパラメータ名について説明したマニュアルを参照して、パラメータ値を修正してください。	1
PARAMETER_NOT_BOOLEAN	ネットワーク・パラメータの形式は、"name=value;[name2=value2[;...]]" です。このパラメータ値はブール値ではありません。パラメータ指定の違反箇所を探して、パラメータ値を 0 (off または False に対応) または 1 (on または True に対応) に変更してください。	4
PARAMETER_NOT_HEX	ネットワーク・パラメータの形式は、"name=value;[name2=value2[;...]]" です。このパラメータ値は、16 進数値 (基数 16) ではありません。パラメータ指定の違反箇所を探して、パラメータ値を 16 進数値に変更してください。	5
PARAMETER_NOT_UINT32	ネットワーク・パラメータの形式は、"name=value;[name2=value2[;...]]" です。このパラメータ値は、符号なし整数ではありません。パラメータ指定の違反箇所を探して、パラメータ値を符号なしの整数に変更してください。	2
PARAMETER_NOT_UINT32_RANGE	ネットワーク・パラメータの形式は、"name=value;[name2=value2[;...]]" です。このパラメータ値は、符号なし整数値または範囲のいずれでもありません。パラメータ指定の違反箇所を探して、パラメータ値を符号なしの整数値か範囲に変更してください。符号なし範囲の形式は NNN-NNN です。	3

メンバ名	説明	値
PARSE	ネットワーク・パラメータの形式は、"name=value;[name2=value2[;...]]" です。オプションで、カッコ内にパラメータの全リストを入力できます。入力された文字列はこの表記規則に従っていません。文字列を検査して、フォーマット上の問題を修正してからもう一度操作を行ってください。	7
PROTOCOL_ERROR	予期しない値またはトークンが読み込まれました。	231

メンバ名	説明	値
READ	ネットワーク・レイヤによって指定されたバイト数を読み込めませんでした。読み込みは、より規模の大きいネットワーク操作の一部として行われます。たとえば、ネットワーク・レイヤによっては、サブ・レイヤを設定して、ここで上位レイヤの基本操作の一部として読み込みと書き込みを行います。通常、読み込みエラーは次のような事柄が原因で発生します。 1) ネットワークに、読み込みを失敗させる問題が生じた。もう一度接続と操作を行ってください。 2) 接続がタイムアウトした。もう一度接続と操作を行ってください。 3) 反対側の接続が完全に終了した。クライアントかサーバ (またはその両方) のログで、接続が削除された理由を示すエラーを確認してください。出力ログのエラーを調べて原因を解明し、もう一度操作を行ってください。 4) 接続の反対側で処理がアボートされた。クライアントかサーバ (またはその両方) のメッセージ・ログで、処理がアボートされた理由を示すエラーを確認してください。通常の方法以外のやり方で処理が停止された場合は、メッセージ・ログにエラーは記録されません。もう一度接続と操作を行ってください。 5) システム・リソースが不足しており、読み込みを実行できない。システム・リソースを解放してから、もう一度接続と操作を行ってください。再度行ってもうまくいかない場合は、ネットワーク管理者に問い合わせてください。	8
READ_TIMEOUT	ネットワーク・レイヤによって指定されたバイト数を指定時間内に読み込めませんでした。ネットワークが正常に機能しており、送信側アプリケーションがまだ動作中であることを確認してください。	201

メンバ名	説明	値
SACI_ERROR_CLIENT	SACI 実装によってエラーがレポートされました。	246
SACI_ERROR_SERVER	SACI ネットワーク・ライブラリがエラーをレポートしています。SACI ネットワーク・ライブラリのプロバイダを参照して、問題を解決してください。	247
SACI_IMPLEMENTATION_MISMATCH	SACI 実装に互換性のない実装 ID が設定されていたため、ロードできませんでした。	250
SECURE_ADD_CERTIFICATE	セキュア・ネットワーク・レイヤが、証明書チェーンに証明書を追加できませんでした。システム・リソースを解放して、もう一度操作を行ってください。	39
SECURE_ADD_TRUSTED_CERTIFICATE	セキュア・ネットワーク・レイヤが、証明書チェーンに信頼できる証明書を追加できませんでした。システム・リソースの不足が主な原因と考えられます。システム・リソースを解放して、もう一度操作を行ってください。	48
SECURE_CERTIFICATE_COMMON_NAME	指定された通称は証明書チェーンに存在しません。次の項目を確認してください。 1) 通称が正しく入力されている。 2) 適切な証明書ファイルが指定されている。 3) 証明書チェーンに通称が存在している。viewcert ユーティリティを使用して上記を確認できます。	52

メンバ名	説明	値
SECURE_CERTIFICATE_COMPANY_NAME	指定された組織名は証明書チェーンに存在しません。次の項目を確認してください。 1) 組織名が正しく入力されている。 2) 適切な証明書ファイルが指定されている。 3) 証明書チェーンに組織名が存在している。viewcert ユーティリティを使用して上記を確認できます。	21
SECURE_CERTIFICATE_COMPANY_UNIT	指定された組織単位は証明書チェーンに存在しません。次の項目を確認してください。 1) 会社名が正しく入力されている。 2) 適切な証明書ファイルが指定されている。 3) 証明書チェーンに会社名が存在している。viewcert ユーティリティを使用して上記を確認できます。	51
SECURE_CERTIFICATE_COUNT	指定されたファイルに証明書が入っていません。次の項目を確認してください。 1) 証明書ファイル名が適切に指定されている。 2) 証明書ファイルに証明書が 1 つ以上入っている。 3) 証明書ファイルに適切な証明書が入っている。	42
SECURE_CERTIFICATE_EXPIRED	証明書チェーンにある証明書の有効期限が切れています。有効期限が切れていない新しい証明書を入手して、再度操作を行ってください。	50

メンバ名	説明	値
SECURE_CERTIFICATE_EXPIRY_DATE	証明書の有効期日を読み込むことができませんでした。次の項目を確認してください。 1) パスワードが正しく入力されている。 2) 証明書ファイルに証明書が 1 つ以上入っている。 3) 証明書ファイルに適切な証明書が入っている。 4) 証明書ファイルは破損していない。	37
SECURE_CERTIFICATE_FILE_NOT_FOUND	証明書ファイルを開けませんでした。次の項目を確認してください。 1) 証明書ファイル名が適切に指定されている。 2) 証明書ファイルが存在する。 3) 証明書ファイルに証明書が 1 つ以上入っている。 4) 証明書ファイルに適切な証明書が入っている。 5) 証明書ファイルを開こうとしているプログラムにファイルを読み込む権限がある。この項目は、ユーザ・パーミッションかファイル・パーミッション (またはその両方) を持つオペレーティング・システムだけに適用されます。	33
SECURE_CERTIFICATE_NOT_TRUSTED	サーバの証明書は信頼できる認証局によって署名されていません。次の項目を確認してください。 1) 証明書ファイル名が適切に指定されている。 2) 証明書ファイルに証明書が 1 つ以上入っている。 3) 証明書ファイルに適切な証明書が入っている。 4) 信頼できるルート証明書のクライアント側のリストに、サーバ側のルート証明書が含まれている。	24

メンバ名	説明	値
SECURE_CERTIFICATE_ROOT	チェーンのルート証明書が不正です。このエラーは、パブリケーション時に定義されましたが使用されませんでした。	20
SECURE_CREATE_CERTIFICATE	セキュア・ネットワーク・レイヤが、証明書用の記憶領域を割り付けることができませんでした。システム・リソースを解放して、もう一度操作を行ってください。	43
SECURE_CREATE_PRIVATE_KEY_OBJECT	プライベート・キーを読み込む前に、セキュア・ネットワーク・レイヤがプライベート・キーのオブジェクトを作成できませんでした。システム・リソースの不足が主な原因と考えられます。システム・リソースを解放して、もう一度操作を行ってください。	49
SECURE_DUPLICATE_CONTEXT	セキュア・ネットワーク・レイヤが、セキュリティ・コンテキストを複製できませんでした。システム・リソースを解放して、もう一度操作を行ってください。	25
SECURE_EXPORT_CERTIFICATE	セキュア・ネットワーク・レイヤが、証明書をコピーできませんでした。システム・リソースを解放して、もう一度操作を行ってください。	38
SECURE_HANDSHAKE	安全なハンドシェイクに失敗しました。次の項目を確認してください。 1) クライアントでは、正しいホスト・マシンとポート番号が指定されている。 2) サーバでは、正しいポート番号が指定されている。 3) クライアントでは、正しい信頼できる証明書が指定されていて、サーバでは、正しい ID ファイルが指定されている。	53
SECURE_IMPORT_CERT_FROM_SYSTEM_STORE	システム証明書ストアから証明書をインポートできませんでした。	222

メンバ名	説明	値
SECURE_IMPORT_CERTIFICATE	セキュア・ネットワーク・レイヤが、証明書をインポートできませんでした。次の項目を確認してください。 1) 証明書ファイル名が適切に指定されている。 2) 証明書ファイルが存在する。 3) 証明書ファイルに証明書が 1 つ以上入っている。 4) 証明書ファイルに適切な証明書が入っている。	44
SECURE_NO_CERTS_IN_SYS_STORE	システム証明書ストアで証明書が見つかりませんでした。	223
SECURE_NO_SERVER_CERTIFICATE	サーバの証明書が指定されていません。通信のセキュリティを保護するには、サーバの証明書が必要です。指定するファイルには、サーバの証明書とプライベート・キーが含まれている必要があります。	205
SECURE_NO_SERVER_CERTIFICATE_PASSWORD	サーバ証明書のパスワードが入力されていません。サーバの暗号化されたプライベート・キーを復号化するには、パスワードが必要です。	206
SECURE_NO_TRUSTED_ROOTS	信頼できるルート証明書が指定されていません。通信のセキュリティを保護するには、信頼できるルート証明書が少なくとも 1 つ必要です。	207
SECURE_OPEN_SYSTEM_CERT_STORE	システム証明書ストアを開けませんでした。	221

メンバ名	説明	値
SECURE_READ_CERTIFICATE	証明書ファイルが読み込めませんでした。次の項目を確認してください。 1) パスワードが正しく入力されている。 2) 証明書ファイルに証明書が 1 つ以上入っている。 3) 証明書ファイルに適切な証明書が入っている。 4) 証明書ファイルは破損していない。	34
SECURE_READ_PRIVATE_KEY	証明書ファイルからプライベート・キーを読み込めませんでした。次の項目を確認してください。 1) パスワードが正しく入力されている。 2) 証明書ファイルに証明書が 1 つ以上入っている。 3) 証明書ファイルに適切な証明書が入っている。 4) 証明書ファイルは破損していない。	35
SECURE_REDUNDANT_SERVER_CERTIFICATE_PASSWORD	サーバのプライベート・キーがパスワードで暗号化されていないのに、パスワードが指定されました。	208
SECURE_SET_IO	セキュア・ネットワーク・レイヤをネットワーク・レイヤに付加できませんでした。システム・リソースを解放して、もう一度操作を行ってください。	26
SECURE_SET_PRIVATE_KEY	プライベート・キーを使用できませんでした。次の項目を確認してください。 1) パスワードが正しく入力されている。 2) 証明書ファイルに証明書が 1 つ以上入っている。 3) 証明書ファイルに適切な証明書が入っている。 4) 証明書ファイルは破損していない。	36

メンバ名	説明	値
SECURE_TRUSTED_CERTIFICATE_FILE_NOT_FOUND	証明書ファイルが見つかりませんでした。次の項目を確認してください。 1) 証明書ファイル名が適切に指定されている。 2) 証明書ファイルが存在する。 3) 証明書ファイルに証明書が 1 つ以上入っている。 4) 証明書ファイルに適切な証明書が入っている。 5) 証明書ファイルを開こうとしているプログラムにファイル参照の権限がある。この項目は、ユーザ・パーミッションかファイル・パーミッション (またはその両方) を持つオペレーティング・システムだけに適用されます。	40
SECURE_TRUSTED_CERTIFICATE_READ	セキュア・ネットワーク・レイヤが、信頼できる証明書ファイルを読み込むことができませんでした。次の項目を確認してください。 1) 証明書ファイル名が適切に指定されている。 2) 証明書ファイルが存在する。 3) 証明書ファイルに証明書が 1 つ以上入っている。 4) 証明書ファイルに適切な証明書が入っている。 5) 証明書ファイルを開こうとしているプログラムにファイル参照の権限がある。この項目は、ユーザ・パーミッションかファイル・パーミッション (またはその両方) を持つオペレーティング・システムだけに適用されます。	41
SEED_RANDOM	セキュア・ネットワーク・レイヤが、乱数ジェネレータにシードを設定できませんでした。システム・リソースを解放してから、もう一度接続と操作を行ってください。	16

メンバ名	説明	値
SERVER_ERROR	サーバによってエラーがレポートされました。詳細については、Mobile Link の管理者に問い合わせてください。	228
SHUTTING_DOWN	シャットダウン中、Mobile Link サーバがネットワーク・レイヤでエラーを検出しました。シャットダウン時に保留になっているネットワーク操作が影響を受けた可能性があります。	18
SOCKET_BIND	ネットワーク・レイヤが、指定ポートにソケットをバインドできませんでした。次の項目を確認してください。 1) (サーバのみ) ポートがまだ使用されていないことを確認する。ポートが使用中の場合は、そのポートで受信しているアプリケーションを停止するか、別のポートを指定してください。 2) (サーバのみ) ポートの使用に関するファイアウォールの制限事項がないことを確認する。 3) (クライアントのみ) client_port オプションが使用された場合は、指定したポートがまだ使用されていないことを確認する。クライアント・ポートを 1 つだけ指定した場合は、NNN-NNN などの範囲を使用することを検討してください。範囲が指定されていた場合は、その範囲を広げるか、別の範囲にすることを検討してください。 4) (クライアントのみ) client_port オプションが使用された場合は、ポートの使用に関するファイアウォールの制限事項がないことを確認する。	60
SOCKET_CLEANUP	ネットワーク・レイヤがソケット・レイヤをクリーンアップできませんでした。これは、すべての接続が終了した後にだけ発生するエラーです。したがって現在の接続には影響しません。	61

メンバ名	説明	値
SOCKET_CLOSE	ネットワーク・レイヤがソケットを閉じることができませんでした。フラッシュされなかった保留中の書き込みが原因で、ネットワーク・セッションが不完全なまま終了した可能性があります (または、終了していない可能性があります)。次の項目を確認してください。 1) ネットワーク接続のどちらか一方の側でエラーが発生した。 2) 接続の反対側は正常に動作している。 3) マシンがネットワークに接続されたままになっており、ネットワークからの応答がある。	62
SOCKET_CONNECT	ネットワーク・レイヤがソケットを接続できませんでした。次の項目を確認してください。 1) マシンがネットワークに接続されている。 2) ソケット・レイヤは適切に初期化されている。 3) 適切なホスト・マシンとポートが指定されている。 4) ホスト・サーバが正常に稼働しており、適切なポートで受信している。 5) ホスト・マシンが、適切なソケット・タイプ (TCP/IP と UDP) で受信を行っている。 6) client_port オプションが使用された場合は、ポートの使用に関するファイアウォールの制限事項がないことを確認する。 7) オープンしているソケット数について、デバイスに制限が設けられている場合は、その値に達していないことを確認する。 8) 使用可能なシステム・リソースが十分にある。不足していれば、システム・リソースを解放してください。	63

メンバ名	説明	値
SOCKET_CREATE_TCPIP	ネットワーク・レイヤが TCP/IP ソケットを作成できませんでした。次の項目を確認してください。 1) マシンがネットワークに接続されている。 2) ソケット・レイヤは適切に初期化されている。 5) オープンしているソケット数について、デバイスに制限が設けられている場合は、その値に達していないことを確認する。 6) 使用可能なシステム・リソースが十分にある。不足していれば、システム・リソースを解放してください。	58

メンバ名	説明	値
SOCKET_CREATE_UDP	ネットワーク・レイヤが UDP ソケットを作成できませんでした。次の項目を確認してください。 1) マシンがネットワークに接続されている。 2) ソケット・レイヤは適切に初期化されている。 3) client_port オプションが使用された場合は、指定したポートがまだ使用されていないことを確認する。クライアント・ポートを 1 つだけ指定した場合は、NNN-NNN などの範囲を使用することを検討してください。範囲が指定されていた場合は、その範囲を広げるか、別の範囲にすることを検討してください。 4) client_port オプションが使用された場合は、ポートの使用に関するファイアウォールの制限事項がないことを確認する。 5) オープンしているソケット数について、デバイスに制限が設けられている場合は、その値に達していないことを確認する。 6) 使用可能なシステム・リソースが十分にある。不足していれば、システム・リソースを解放してください。	59
SOCKET_GET_HOST_BY_ADDR	ネットワーク・レイヤが、IP アドレスを使用してホスト名を取得できませんでした。このエラーは、パブリケーション時に定義されましたが使用されませんでした。	72

メンバ名	説明	値
SOCKET_GET_NAME	ネットワーク・レイヤがソケットのローカル名を判別できませんでした。TCP/IP 接続には、各接続の両端にポート専用に付加されたソケットがあります。ソケットのローカル名にはこのポート番号が含まれており、これは、接続時にネットワークによって割り当てられます。次の項目を確認してください。 1) マシンがネットワークに接続されたままになっており、ネットワークからの応答がある。 2) 接続の反対側は正常に動作している。 3) 使用可能なシステム・リソースが十分にある。不足していれば、システム・リソースを解放してください。	64
SOCKET_GET_OPTION	ネットワーク・レイヤがソケット・オプションを取得できませんでした。このエラーは、接続が失われたことを示す最初の徴候であると考えられます。次の項目を確認してください。 1) マシンがネットワークに接続されたままになっており、ネットワークからの応答がある。 2) 接続の反対側は正常に動作している。 3) 使用可能なシステム・リソースが十分にある。不足していれば、システム・リソースを解放してください。	65

メンバ名	説明	値
SOCKET_HOST_NAME_NOT_FOUND	指定されたホスト名が見つかりませんでした。次の項目を確認してください。 1) ホスト名が正しく指定されている。 2) ホストがアクセス可能である。多くのシステムには、名前を指定したホストへのアクセスを検証できる ping ユーティリティが組み込まれています。 3) ドメイン・ネーム・サーバ (DNS) またはそれに相当するものが使用できる。DNS を使用できない場合は、ホスト名の代わりにホストの IP アドレス (例：NNN.NNN.NNN.NNN) を指定してみてください。 4) HOSTS ファイルに、ホスト名を IP アドレスにマッピングするエントリが入っている。	57
SOCKET_LISTEN	サーバがソケットで受信できません。バックログとは、ある一定期間に保留中になっている可能性がある、キューイングされた接続要求の最大数のことです。次の項目を確認してください。 1) マシンがネットワークに接続されたままになっており、ネットワークからの応答がある。 2) 現在のマシンでソケット・リスナが動作するのを妨げるファイアウォールやその他の制限事項がない。 3) マシンに制限がある場合、バックログ設定が制限内に収まっている。 4) 使用可能なシステム・リソースが十分にある。不足していれば、システム・リソースを解放してください。	67
SOCKET_LIVENESS_OUT_OF_RANGE	無効な活性タイムアウト値が指定されました。活性タイムアウト値は、0 ～ 65535 の間の整数にしてください。	200

メンバ名	説明	値
SOCKET_LOCALHOST_NAME_NOT_FOUND	ネットワーク・レイヤが "localhost" の IP アドレスを判別できませんでした。次の項目を確認してください。 1) ドメイン・ネーム・サーバ (DNS) またはそれに相当するものが使用できる。DNS が使用できない場合には、代わりに localhost の IP アドレスを明示的に指定してみます (通常は 127.0.0.1)。 2) HOSTS ファイルに、"localhost" 名を IP アドレスにマッピングするエントリが入っている。 3) 使用可能なシステム・リソースが十分にある。不足していれば、システム・リソースを解放してください。	71
SOCKET_PORT_OUT_OF_RANGE	不正なポート番号を指定しています。ポート番号は、0 ～ 65535 の間の整数にしてください。	74
SOCKET_SELECT	ネットワーク・レイヤで、読み込みまたは書き込み可能状態のソケットに対して待ち状態に入ろうとするエラーが発生しました。次の項目を確認してください。 1) ネットワークにマシンが接続されており、ネットワークからの応答がある。 2) 接続の反対側は正常に動作している。 3) 使用可能なシステム・リソースが十分にある。不足していれば、システム・リソースを解放してください。	69

メンバ名	説明	値
SOCKET_SET_OPTION	ネットワーク・レイヤがソケット・オプションを設定できませんでした。このエラーは、接続が失われたことを示す最初の徴候であると考えられます。次の項目を確認してください。 1) マシンがネットワークに接続されたままになっており、ネットワークからの応答がある。 2) 接続の反対側は正常に動作している。 3) 使用可能なシステム・リソースが十分にある。不足していれば、システム・リソースを解放してください。	66
SOCKET_SHUTDOWN	ネットワーク・レイヤがソケットをシャトダウンできませんでした。次の項目を確認してください。 1) ネットワークにマシンが接続されており、ネットワークからの応答がある。 2) 接続の反対側は正常に動作している。 3) 使用可能なシステム・リソースが十分にある。不足していれば、システム・リソースを解放してください。	68
SOCKET_STARTUP	ネットワーク・レイヤがソケット・レイヤを初期化できませんでした。次の項目を確認してください。 1) ソケット・レイヤが適切にインストールされている。正しいネットワーク・インタフェース・ライブラリが存在し、アクセス可能になっていなければなりません。 2) ネットワークにマシンが接続されており、ネットワークからの応答がある。 3) 使用可能なシステム・リソースが十分にある。不足していれば、システム・リソースを解放してください。	70

メンバ名	説明	値
UNEXPECTED_HTTP_REQUEST_TYPE	この HTTP 要求タイプはこの時点では予期されていませんでした。Mobile Link クライアントではない HTTP クライアントが主な原因と考えられます。	234
UNRECOGNIZED_TLS_TYPE	TLS タイプが無効です。有効なタイプについては、マニュアルを参照してください。	240
VALUE_OUT_OF_RANGE	指定された値が、パラメータに許可される値の範囲内ではありませんでした。マニュアルでパラメータの説明を参照して、値に許可される範囲を確認してください。	233
WOULD_BLOCK	ブロッキングが不要または予期されないところで、要求された操作がブロックした可能性があります。	13

メンバ名	説明	値
WRITE	指定されたバイト数をネットワーク・レイヤに書き込むことができません。書き込みは、より規模の大きいネットワーク操作の一部として行われます。たとえば、ネットワーク・レイヤによっては、サブ・レイヤを設定して、ここで上位レイヤの基本操作の一部として読み込みと書き込みを行います。通常、書き込みエラーは次のような事柄が原因で発生します。 1) ネットワークに、書き込みを失敗させる問題が生じた。もう一度接続と操作を行ってください。 2) 接続がタイムアウトした。もう一度接続と操作を行ってください。 3) 反対側の接続が完全に終了した。クライアントかサーバ (またはその両方) のログで、接続が削除された理由を示すエラーを確認してください。出力ログのエラーを調べて原因を解明し、もう一度操作を行ってください。 4) 接続の反対側で処理がアボートされた。クライアントかサーバ (またはその両方) のメッセージ・ログで、処理がアボートされた理由を示すエラーを確認してください。通常の方法以外のやり方で処理が停止された場合は、メッセージ・ログにエラーは記録されません。もう一度接続と操作を行ってください。 5) システム・リソースが不足しており、書き込みを実行できない。システム・リソースを解放してから、もう一度接続と操作を行ってください。再度行ってもうまくいかない場合は、ネットワーク管理者に問い合わせてください。	9
WRITE_TIMEOUT	ネットワーク・レイヤによって指定されたバイト数を指定時間内に書き込めませんでした。ネットワークが正常に機能しており、受信側アプリケーションがまだ動作中であることを確認してください。	202

参照

- 「StreamErrorCode プロパティ」 545 ページ

ULStreamType 列挙体

UL 拡張：同期に使用する Mobile Link 同期ストリームのタイプを列挙します。

構文

Visual Basic
Public Enum **ULStreamType**

C#
public enum **ULStreamType**

備考

特定のストリームのタイプの設定方法については、「Ultra Light 同期ストリームのネットワーク・プロトコルのオプション」『Ultra Light データベース管理とリファレンス』を参照してください。

別途ライセンスが必要な必須コンポーネント
ECC 暗号化と FIPS 認定の暗号化には、別途ライセンスが必要です。強力な暗号化テクノロジはすべて、輸出規制対象品目です。

「別途ライセンスが必要なコンポーネント」『SQL Anywhere 11 - 紹介』を参照してください。

メンバ

メンバ名	説明	値
HTTP	HTTP を介して同期します。 HTTP ストリームは基本となるトランスポートとして TCP/IP を使用します。Ultra Light アプリケーションは Web ブラウザとして機能し、Mobile Link サーバは Web サーバとして機能します。Ultra Light アプリケーションは、サーバへのデータ送信のために POST 要求を送り、サーバからのデータの読み込みのために GET 要求を送ります。	1
HTTPS	HTTPS を介して同期します (トランスポート・レイヤ・セキュリティを適用した HTTP)。	2
TCPIP	TCP/IP を介して同期します。	0
TLS	TLS (TCP/IP を介して同期)	3

HTTPS や TLS のストリームで暗号化された同期を使用する場合は、適切な DLL ファイルをデバイスに配備する必要があります。次の表に、配備する DLL ファイルを暗号化プロトコル別に示します。DLL ファイルは、SQL Anywhere インストール環境の *UltraLite¥CE* ディレクトリにあります。

プロトコル	DLL
ECC	*mlcecc10.dll*
RSA	*mlcrsa10.dll*
FIPS	*mlcrsafips10.dll*

参照

- 「Stream プロパティ」 524 ページ

ULSyncParms クラス

UL 拡張： Ultra Light データベースの同期方法を定義する同期パラメータを表します。このクラスは継承できません。

構文

Visual Basic
Public NotInheritable Class **ULSyncParms**

C#
public sealed class **ULSyncParms**

備考

このクラスにはコンストラクタがありません。各接続には、ULConnection.SyncParms としてアタッチされた、固有の ULSyncParms インスタンスがあります。

同期コマンド (ULSyncParms.DownloadOnly、ULSyncParms.PingOnly、ULSyncParms.ResumePartialDownload、ULSyncParms.UploadOnly) は、一度に 1 つしか指定できません。これらの複数のパラメータが true に設定されていると、ULSQLCode.SQLE_SYNC_INFO_INVALID SQLException が ULConnection.Synchronize() によってスローされます。

その他の ULSQLCode.SQLE_SYNC_INFO_INVALID エラーの原因には、ULSyncParms.Stream 値または ULSyncParms.Version 値を指定していないことが含まれます。

参照

- 「ULSyncParms メンバ」 515 ページ
- 「ULConnection クラス」 139 ページ
- 「SyncParms プロパティ」 152 ページ
- 「Synchronize() メソッド」 182 ページ
- 「DownloadOnly プロパティ」 517 ページ
- 「PingOnly プロパティ」 520 ページ
- 「ResumePartialDownload プロパティ」 522 ページ
- 「UploadOnly プロパティ」 525 ページ
- 「ULSQLCode 列挙体」 462 ページ
- 「Synchronize() メソッド」 182 ページ
- 「Stream プロパティ」 524 ページ
- 「Version プロパティ」 526 ページ

ULSyncParms メンバ

パブリック・プロパティ

メンバ名	説明
「AdditionalParms プロパティ」 516 ページ	「名前=値」のペアをセミコロンで区切ったリストで、追加のパラメータを指定します。ここで指定されるのは、あまり一般的に使用されないパラメータです。
「AuthenticationParms プロパティ」 517 ページ	カスタム・ユーザ認証スクリプト (Mobile Link authenticate_parameters 接続イベント) のパラメータを指定します。
「DownloadOnly プロパティ」 517 ページ	同期時のアップロードを無効にするか、有効にするかを指定します。
「KeepPartialDownload プロパティ」 518 ページ	同期時の部分的なダウンロードを無効にするか、有効にするかを指定します。
「NewPassword プロパティ」 519 ページ	UserName で指定されたユーザの新しい Mobile Link パスワードを指定します。
「Password プロパティ」 520 ページ	UserName で指定されたユーザの Mobile Link パスワードです。
「PingOnly プロパティ」 520 ページ	実際に同期を行う代わりに、クライアントが Mobile Link サーバに ping のみを行うかどうかを指定します。
「Publications プロパティ」 521 ページ	同期させるパブリケーションを指定します。
「ResumePartialDownload プロパティ」 522 ページ	前の部分的なダウンロードを再開するか、破棄するかを指定します。
「SendColumnNames プロパティ」 523 ページ	同期中に、クライアントが Mobile Link サーバにカラム名を送信するかどうかを指定します。
「SendDownloadAck プロパティ」 523 ページ	同期中に、クライアントが Mobile Link サーバにダウンロード確認を送信するかどうかを指定します。ダウンロード確認は、ダウンロードがリモートで完全に適用されてコミットされた後 (正の確認)、またはダウンロードに失敗した後 (負の確認) に送信されます。
「Stream プロパティ」 524 ページ	同期に使用する Mobile Link 同期ストリームを指定します。

メンバ名	説明
「StreamParms プロパティ」 524 ページ	同期ストリームの設定パラメータを指定します。
「UploadOnly プロパティ」 525 ページ	同期時のダウンロードを無効にするか、有効にするかを指定します。
「UserName プロパティ」 526 ページ	Mobile Link サーバが Mobile Link クライアントをユニークに識別するユーザ名です。
「Version プロパティ」 526 ページ	使用する同期スクリプトを指定します。

パブリック・メソッド

メンバ名	説明
「CopyFrom メソッド」 527 ページ	指定された ULSyncParms オブジェクトのプロパティを、この ULSyncParms オブジェクトにコピーします。

参照

- 「ULSyncParms クラス」 514 ページ
- 「ULConnection クラス」 139 ページ
- 「SyncParms プロパティ」 152 ページ
- 「Synchronize() メソッド」 182 ページ
- 「DownloadOnly プロパティ」 517 ページ
- 「PingOnly プロパティ」 520 ページ
- 「ResumePartialDownload プロパティ」 522 ページ
- 「UploadOnly プロパティ」 525 ページ
- 「ULSQLCode 列挙体」 462 ページ
- 「Synchronize() メソッド」 182 ページ
- 「Stream プロパティ」 524 ページ
- 「Version プロパティ」 526 ページ

AdditionalParms プロパティ

「キーワード=値」のペアの文字列を指定します。通常、このペアのリストには、使用頻度の低い同期パラメータが入ります。

構文

Visual Basic
Public Property **AdditionalParms** As String

C#
public string **AdditionalParms** { get; set; }

プロパティ値

「キーワード=値」をセミコロンで区切ったリストによる追加のパラメータ。

参照

- 「ULSyncParms クラス」 514 ページ
- 「ULSyncParms メンバ」 515 ページ

AuthenticationParms プロパティ

カスタム・ユーザ認証スクリプト (Mobile Link authenticate_parameters 接続イベント) のパラメータを指定します。

構文

Visual Basic
Public Property **AuthenticationParms** As String()

C#
public string[] **AuthenticationParms** { get; set; }

プロパティ値

それぞれに認証パラメータが格納された、文字列の配列 (配列のエントリが NULL であると、同期エラーになります)。デフォルト値は NULL 参照 (Visual Basic の Nothing) で、認証パラメータは指定されません。

備考

最初の 255 文字のみが使用されます。また、各文字列は 128 文字以下であることが必要です (長すぎる文字列は、Mobile Link サーバに送信されるときにトランケートされます)。

参照

- 「ULSyncParms クラス」 514 ページ
- 「ULSyncParms メンバ」 515 ページ

DownloadOnly プロパティ

同期時のアップロードを無効にするか、有効にするかを指定します。

構文

Visual Basic
Public Property **DownloadOnly** As Boolean

C#
public bool **DownloadOnly** { get; set; }

プロパティ値

同期時のアップロードを無効にする場合は true、有効にする場合は false。デフォルトは false です。

備考

同期コマンド (ULSyncParms.DownloadOnly、ULSyncParms.PingOnly、ULSyncParms.ResumePartialDownload、ULSyncParms.UploadOnly) は、一度に 1 つしか指定できません。これらの複数のパラメータが true に設定されていると、ULSQLCode.SQLE_SYNC_INFO_INVALID SQLException が ULConnection.Synchronize() によってスローされます。

参照

- 「ULSyncParms クラス」 514 ページ
- 「ULSyncParms メンバ」 515 ページ
- 「UploadOnly プロパティ」 525 ページ
- 「DownloadOnly プロパティ」 517 ページ
- 「PingOnly プロパティ」 520 ページ
- 「ResumePartialDownload プロパティ」 522 ページ
- 「UploadOnly プロパティ」 525 ページ
- 「ULSQLCode 列挙体」 462 ページ
- 「Synchronize() メソッド」 182 ページ

KeepPartialDownload プロパティ

同期時の部分的なダウンロードを無効にするか、有効にするかを指定します。

構文

Visual Basic
Public Property **KeepPartialDownload** As Boolean

C#
public bool **KeepPartialDownload** { get; set; }

プロパティ値

同期時の部分的なダウンロードを有効にする場合は true、無効にする場合は false。デフォルトは false です。

備考

Ultra Light.NET では、通信エラーや、ユーザによる ULSyncProgressListener からのアボートが原因で失敗したダウンロードを再起動できます。Ultra Light.NET は、ダウンロードを受信しながら処理します。ダウンロードが中断した場合は、部分的なダウンロード・トランザクションがデータベース内に残るため、次の同期中に再開できます。

Ultra Light.NET で部分的なダウンロードを保存する必要があることを示すには、connection.SyncParms.KeepPartialDownload=true と指定します。指定しないと、エラーが発生した場合にダウンロードがロールバックされます。

部分的なダウンロードが保持された場合、connection.Synchronize() の終了時に、出力フィールド connection.SyncResult.ULSyncResult.PartialDownloadRetained が true に設定されます。

PartialDownloadRetained が設定されている場合は、ダウンロードを再開できます。再開するには、true に設定された connection.SyncParms.ULSyncParms.ResumePartialDownload を使用して connection.Synchronize() を呼び出します。別の通信エラーの発生に備えて、KeepPartialDownload も true に設定しておくことをおすすめします。ダウンロードが省略された場合は、アップロードは行われません。

再開したダウンロードで受信するダウンロードは、最初にダウンロードを開始したときと同じものです。最新のデータが必要な場合は、再開された特別なダウンロードが完了した直後に、もう一度ダウンロードを行うことができます。

ダウンロードを再開する場合、ULSyncParms フィールドの多くは関係ありません。たとえば、Publications フィールドは使用されません。受信するパブリケーションは、最初のダウンロード時に要求したものです。設定する必要があるフィールドは、ResumePartialDownload と UserName だけです。KeepPartialDownload フィールドと DisableConcurrency フィールドを必要に応じて設定すると、正常に機能します。

部分的なダウンロードが存在するが、このダウンロードが必要ではなくなった場合は、ULConnection.RollbackPartialDownload() を呼び出して、失敗したダウンロード・トランザクションをロールバックできます。また、同期をもう一度実行したときに ResumePartialDownload を指定しなかった場合は、部分的なダウンロードがロールバックされてから、次の同期が開始されます。

詳細については、「失敗したダウンロードの再開」 『Mobile Link - サーバ管理』を参照してください。

参照

- 「ULSyncParms クラス」 514 ページ
- 「ULSyncParms メンバ」 515 ページ
- 「PartialDownloadRetained プロパティ」 544 ページ
- 「ResumePartialDownload プロパティ」 522 ページ
- 「RollbackPartialDownload メソッド」 180 ページ
- 「ResumePartialDownload プロパティ」 522 ページ
- 「UserName プロパティ」 526 ページ
- 「AdditionalParms プロパティ」 516 ページ
- 「RollbackPartialDownload メソッド」 180 ページ

NewPassword プロパティ

UserName で指定されたユーザの新しい Mobile Link パスワードを指定します。

構文

Visual Basic
Public Property **NewPassword** As String

C#
public string **NewPassword** { get; set; }

プロパティ値

新しい Mobile Link パスワードを指定する文字列。デフォルト値は NULL 参照 (Visual Basic の Nothing) で、パスワードは変更されません。

備考

新しいパスワードが有効になるのは、次の同期の後です。

参照

- 「ULSyncParms クラス」 514 ページ
- 「ULSyncParms メンバ」 515 ページ
- 「UserName プロパティ」 526 ページ

Password プロパティ

UserName で指定されたユーザの Mobile Link パスワードです。

構文

Visual Basic
Public Property **Password** As String

C#
public string **Password** { get; set; }

プロパティ値

Mobile Link パスワードを指定する文字列。デフォルト値は NULL 参照 (Visual Basic の Nothing) で、パスワードは指定されません。

備考

Mobile Link ユーザ名とパスワードは他のデータベース・ユーザ ID やパスワードとは別のもので、アプリケーションを Mobile Link サーバに対して識別し、認証するために使用されます。

参照

- 「ULSyncParms クラス」 514 ページ
- 「ULSyncParms メンバ」 515 ページ
- 「NewPassword プロパティ」 519 ページ
- 「UserName プロパティ」 526 ページ

PingOnly プロパティ

実際に同期を行う代わりに、クライアントが Mobile Link サーバに ping のみを行うかどうかを指定します。

構文

Visual Basic
Public Property **PingOnly** As Boolean

C#
public bool **PingOnly** { get; set; }

プロパティ値

クライアントが Mobile Link サーバに ping のみを行うように指定する場合は true、クライアントが実際に同期を行うように指定する場合は false。デフォルトは false です。

備考

同期コマンド (ULSyncParms.DownloadOnly、ULSyncParms.PingOnly、ULSyncParms.ResumePartialDownload、ULSyncParms.UploadOnly) は、一度に 1 つしか指定できません。これらの複数のパラメータが true に設定されていると、ULSQLCode.SQLE_SYNC_INFO_INVALID SQLException が ULConnection.Synchronize() によってスローされます。

参照

- 「ULSyncParms クラス」 514 ページ
- 「ULSyncParms メンバ」 515 ページ
- 「DownloadOnly プロパティ」 517 ページ
- 「PingOnly プロパティ」 520 ページ
- 「ResumePartialDownload プロパティ」 522 ページ
- 「UploadOnly プロパティ」 525 ページ
- 「ULSQLCode 列挙体」 462 ページ
- 「Synchronize() メソッド」 182 ページ

Publications プロパティ

同期させるパブリケーションを指定します。

構文

Visual Basic
Public Property **Publications** As String

C#
public string **Publications** { get; set; }

プロパティ値

カンマ (,) で区切られたパブリケーション名のリストが含まれた文字列、特別値 ULPublicationSchema.SYNC_ALL_PUBS、または特別値 ULPublicationSchema.SYNC_ALL_DB。デフォルトは ULPublicationSchema.SYNC_ALL_DB です。詳細については、「ULPublicationSchema クラス」 412 ページを参照してください。

参照

- 「ULSyncParms クラス」 514 ページ
- 「ULSyncParms メンバ」 515 ページ
- 「SYNC_ALL_PUBS フィールド」 413 ページ
- 「SYNC_ALL_DB フィールド」 413 ページ
- 「ULPublicationSchema クラス」 412 ページ

ResumePartialDownload プロパティ

前の部分的なダウンロードを再開するか、破棄するかを指定します。

構文

Visual Basic
Public Property **ResumePartialDownload** As Boolean

C#
public bool **ResumePartialDownload** { get; set; }

プロパティ値

前の部分的なダウンロードを再開する場合は true、破棄する場合は false。デフォルトは false です。

備考

同期コマンド (ULSyncParms.DownloadOnly、ULSyncParms.PingOnly、ULSyncParms.ResumePartialDownload、ULSyncParms.UploadOnly) は、一度に 1 つしか指定できません。これらの複数のパラメータが true に設定されていると、ULSQLCode.SQLE_SYNC_INFO_INVALID SQLException が ULConnection.Synchronize() によってスローされます。

部分的なダウンロードの詳細については、「KeepPartialDownload プロパティ」 518 ページを参照してください。

参照

- 「ULSyncParms クラス」 514 ページ
- 「ULSyncParms メンバ」 515 ページ
- 「DownloadOnly プロパティ」 517 ページ
- 「PingOnly プロパティ」 520 ページ
- 「ResumePartialDownload プロパティ」 522 ページ
- 「UploadOnly プロパティ」 525 ページ
- 「ULSQLCode 列挙体」 462 ページ
- 「Synchronize() メソッド」 182 ページ
- 「PartialDownloadRetained プロパティ」 544 ページ

SendColumnNames プロパティ

同期中に、クライアントが Mobile Link サーバにカラム名を送信するかどうかを指定します。

構文

Visual Basic
Public Property **SendColumnNames** As Boolean

C#
public bool **SendColumnNames** { get; set; }

プロパティ値

クライアントがカラム名を Mobile Link サーバに送信する必要があると指定する場合は true、カラム名を送信しないように指定する場合は false。デフォルトは false です。

備考

カラム名は、Mobile Link サーバによってダイレクト・ロー・ハンドリングで使用されます。Mobile Link サーバがロー・ハンドリング API を使用して、インデックスではなく、名前でカラムを参照する場合は、このオプションを設定してください。このオプションで送信されるカラム名は、このような場合にのみ使用されます。

参照

- 「ULSyncParms クラス」 514 ページ
- 「ULSyncParms メンバ」 515 ページ

SendDownloadAck プロパティ

同期中に、クライアントが Mobile Link サーバにダウンロード確認を送信するかどうかを指定します。ダウンロード確認は、ダウンロードがリモートで完全に適用されてコミットされた後 (正の確認)、またはダウンロードに失敗した後 (負の確認) に送信されます。

構文

Visual Basic
Public Property **SendDownloadAck** As Boolean

C#
public bool **SendDownloadAck** { get; set; }

プロパティ値

True に設定すると、クライアントが Mobile Link サーバにダウンロード確認を送信することを指定します。ダウンロード確認を送信しないように指定する場合は False。デフォルト値は False です。

備考

クライアントがダウンロード確認を送信する場合、Mobile Link サーバのデータベース・ワーカ・スレッドは、クライアントがダウンロードを適用してコミットするまで待機します。クライアントがダウンロード確認を送信しない場合、Mobile Link サーバは、次の同期のため、より早く解放されます。

参照

- 「ULSyncParms クラス」 514 ページ
- 「ULSyncParms メンバ」 515 ページ

Stream プロパティ

同期に使用する Mobile Link 同期ストリームを指定します。

構文

Visual Basic
Public Property **Stream** As ULStreamType

C#
public ULStreamType **Stream** { get; set; }

プロパティ値

使用する同期ストリームのタイプを指定する ULStreamType 値の 1 つ。デフォルトは ULStreamType.TCPIP です。

備考

ほとんどの同期ストリームでは、Mobile Link サーバのアドレスを識別したり、その他の動作を制御したりするパラメータが必要です。これらのパラメータは、ULSyncParms.StreamParms で指定します。

ストリーム・タイプが、プラットフォームに適さない無効な値に設定されていると、ストリーム・タイプは ULStreamType.TCPIP に設定されます。

参照

- 「ULSyncParms クラス」 514 ページ
- 「ULSyncParms メンバ」 515 ページ
- 「ULStreamType 列挙体」 512 ページ
- 「StreamParms プロパティ」 524 ページ
- 「ULStreamType 列挙体」 512 ページ

StreamParms プロパティ

同期ストリームの設定パラメータを指定します。

構文

Visual Basic
Public Property **StreamParms** As String

C#
public string **StreamParms** { get; set; }

プロパティ値

キーワードと値の組み合わせがセミコロンで区切られたリスト形式の、ストリームのパラメータを指定する文字列。デフォルト値は NULL 参照 (Visual Basic の Nothing) です。

備考

特定のストリームのタイプの設定方法については、次を参照してください。

「Ultra Light 同期ストリームのネットワーク・プロトコルのオプション」『Ultra Light データベース管理とリファレンス』

StreamParms は、同期ストリームに使用されるすべてのパラメータが含まれている文字列です。パラメータは、「名前=値」のペアをセミコロンで区切ったリスト ("param1=value1;param2=value2") で指定します。

参照

- 「ULSyncParms クラス」 514 ページ
- 「ULSyncParms メンバ」 515 ページ
- 「Stream プロパティ」 524 ページ
- 「ULStreamType 列挙体」 512 ページ

UploadOnly プロパティ

同期時のダウンロードを無効にするか、有効にするかを指定します。

構文

Visual Basic
Public Property **UploadOnly** As Boolean

C#
public bool **UploadOnly** { get; set; }

プロパティ値

ダウンロードを無効にする場合は true、有効にする場合は false。デフォルトは false です。

備考

同期コマンド (ULSyncParms.DownloadOnly、ULSyncParms.PingOnly、ULSyncParms.ResumePartialDownload、ULSyncParms.UploadOnly) は、一度に 1 つしか指定できません。これらの複数のパラメータが true に設定されていると、ULSQLCode.SQLE_SYNC_INFO_INVALID SQLException が ULConnection.Synchronize() によってスローされます。

参照

- 「ULSyncParms クラス」 514 ページ
- 「ULSyncParms メンバ」 515 ページ
- 「DownloadOnly プロパティ」 517 ページ
- 「PingOnly プロパティ」 520 ページ
- 「ResumePartialDownload プロパティ」 522 ページ
- 「UploadOnly プロパティ」 525 ページ
- 「ULSQLCode 列挙体」 462 ページ
- 「Synchronize() メソッド」 182 ページ

UserName プロパティ

Mobile Link サーバが Mobile Link クライアントをユニークに識別するユーザ名です。

構文

Visual Basic
Public Property **UserName** As String

C#
public string **UserName** { get; set; }

プロパティ値

ユーザ名を指定する文字列。このパラメータにはデフォルト値がないので、明示的に設定してください。

備考

Mobile Link サーバでは、この値を使用して、ダウンロードする内容の決定、同期ステータスの記録、同期中の割り込みからの復帰を行います。このユーザ名とパスワードは他のデータベース・ユーザ ID やパスワードとは別のもので、アプリケーションを Mobile Link サーバに対して識別し、認証するために使用されます。

参照

- 「ULSyncParms クラス」 514 ページ
- 「ULSyncParms メンバ」 515 ページ
- 「Password プロパティ」 520 ページ

Version プロパティ

使用する同期スクリプトを指定します。

構文

Visual Basic
Public Property **Version** As String

C#
public string **Version** { get; set; }

プロパティ値

使用する同期スクリプトのバージョンを指定する文字列。このパラメータにはデフォルト値がないので、明示的に設定してください。

備考

統合データベースの同期スクリプトは、それぞれバージョン文字列でマーク付けされます。たとえば、異なる文字列バージョンによって特定される 2 種類の download_cursor スクリプトがあります。Ultra Light アプリケーションは、バージョン文字列により、同期スクリプトのセットから選択できます。

参照

- 「ULSyncParms クラス」 514 ページ
- 「ULSyncParms メンバ」 515 ページ

CopyFrom メソッド

指定された ULSyncParms オブジェクトのプロパティを、この ULSyncParms オブジェクトにコピーします。

構文

Visual Basic
```
Public Sub CopyFrom( _
  ByVal src As ULSyncParms _
)
```

C#
```
public void CopyFrom(
  ULSyncParms src
);
```

パラメータ

- **src** コピー元のオブジェクト。

参照

- 「ULSyncParms クラス」 514 ページ
- 「ULSyncParms メンバ」 515 ページ
- 「ULSyncParms クラス」 514 ページ

ULSyncProgressData クラス

UL 拡張：同期の進行状況のモニタリング・データを返します。

構文

Visual Basic
Public Class **ULSyncProgressData**

C#
public class **ULSyncProgressData**

参照

- 「ULSyncProgressData メンバ」 528 ページ
- 「ULSyncProgressListener インタフェース」 538 ページ

ULSyncProgressData メンバ

パブリック・フィールド

メンバ名	説明
「FLAG_IS_BLOCKING フィールド」 529 ページ	Mobile Link サーバからの応答の待機中、同期はブロックされていることを示すフラグです。このフィールドは定数で、読み込み専用です。

パブリック・プロパティ

メンバ名	説明
「Flags プロパティ」 530 ページ	現在の状態に関連する追加情報を示す、現在の同期フラグを返します。
「ReceivedBytes プロパティ」 530 ページ	現在までに受信したバイト数を返します。この情報は、すべてのステータスで更新されます。
「ReceivedDeletes プロパティ」 531 ページ	現在までに受信した削除済みのローの数を返します。
「ReceivedInserts プロパティ」 531 ページ	現在までに受信した挿入済みのローの数を返します。
「ReceivedUpdates プロパティ」 531 ページ	現在までに受信した更新済みのローの数を返します。
「SentBytes プロパティ」 532 ページ	現在までに送信されたバイト数を返します。この情報は、すべてのステータスで更新されます。

メンバ名	説明
「SentDeletes プロパティ」 532 ページ	現在までに送信された削除済みのローの数を返します。
「SentInserts プロパティ」 533 ページ	現在までに送信された挿入済みのローの数を返します。
「SentUpdates プロパティ」 533 ページ	現在までに送信された更新済みのローの数を返します。
「State プロパティ」 534 ページ	現在の同期のステータスを返します。
「SyncTableCount プロパティ」 534 ページ	同期中のテーブルの数を返します。
「SyncTableIndex プロパティ」 535 ページ	現在同期中のテーブルのインデックスを返します (テーブルには 1 ～ DatabaseSchema.TableCount までの番号が付けられています)。
「TableID プロパティ」 535 ページ	現在同期中のテーブルのデータベース・インデックスを返します。
「TableName プロパティ」 536 ページ	アップロード中またはダウンロード中の現在のテーブルの名前を返します。

パブリック・メソッド

メンバ名	説明
「setValue メソッド」 536 ページ	現在の同期値をこのオブジェクトにロードします。

参照

- 「ULSyncProgressData クラス」 528 ページ
- 「ULSyncProgressListener インタフェース」 538 ページ

FLAG_IS_BLOCKING フィールド

Mobile Link サーバからの応答の待機中、同期はブロックされていることを示すフラグです。このフィールドは定数で、読み込み専用です。

構文

Visual Basic
Public Shared **FLAG_IS_BLOCKING** As Integer

C#
public const int **FLAG_IS_BLOCKING**;

参照

- 「ULSyncProgressData クラス」 528 ページ
- 「ULSyncProgressData メンバ」 528 ページ

Flags プロパティ

現在の状態に関連する追加情報を示す、現在の同期フラグを返します。

構文

Visual Basic
Public Readonly Property **Flags** As Integer

C#
public int **Flags** { get;}

プロパティ値

フラグの組み合わせを保持する整数。

参照

- 「ULSyncProgressData クラス」 528 ページ
- 「ULSyncProgressData メンバ」 528 ページ
- 「FLAG_IS_BLOCKING フィールド」 529 ページ

ReceivedBytes プロパティ

現在までに受信したバイト数を返します。この情報は、すべてのステータスで更新されます。

構文

Visual Basic
Public Readonly Property **ReceivedBytes** As Long

C#
public long **ReceivedBytes** { get;}

プロパティ値

現在までに受信したバイト数。

参照

- 「ULSyncProgressData クラス」 528 ページ
- 「ULSyncProgressData メンバ」 528 ページ
- 「ULSyncProgressState 列挙体」 540 ページ
- 「ULSyncProgressState 列挙体」 540 ページ

ReceivedDeletes プロパティ

現在までに受信した削除済みのローの数を返します。

構文

Visual Basic
Public Readonly Property **ReceivedDeletes** As Integer

C#
public int **ReceivedDeletes** { get;}

プロパティ値

現在までに受信した削除済みのローの数。

参照

- 「ULSyncProgressData クラス」 528 ページ
- 「ULSyncProgressData メンバ」 528 ページ
- 「ULSyncProgressState 列挙体」 540 ページ
- 「ULSyncProgressState 列挙体」 540 ページ

ReceivedInserts プロパティ

現在までに受信した挿入済みのローの数を返します。

構文

Visual Basic
Public Readonly Property **ReceivedInserts** As Integer

C#
public int **ReceivedInserts** { get;}

プロパティ値

現在までに受信した挿入済みのローの数。

参照

- 「ULSyncProgressData クラス」 528 ページ
- 「ULSyncProgressData メンバ」 528 ページ
- 「ULSyncProgressState 列挙体」 540 ページ
- 「ULSyncProgressState 列挙体」 540 ページ

ReceivedUpdates プロパティ

現在までに受信した更新済みのローの数を返します。

構文

Visual Basic
Public Readonly Property **ReceivedUpdates** As Integer

C#
public int **ReceivedUpdates** { get;}

プロパティ値

現在までに受信した更新済みのローの数。

参照

- 「ULSyncProgressData クラス」 528 ページ
- 「ULSyncProgressData メンバ」 528 ページ
- 「ULSyncProgressState 列挙体」 540 ページ
- 「ULSyncProgressState 列挙体」 540 ページ

SentBytes プロパティ

現在までに送信されたバイト数を返します。この情報は、すべてのステータスで更新されます。

構文

Visual Basic
Public Readonly Property **SentBytes** As Long

C#
public long **SentBytes** { get;}

プロパティ値

現在までに送信されたバイト数。

参照

- 「ULSyncProgressData クラス」 528 ページ
- 「ULSyncProgressData メンバ」 528 ページ
- 「ULSyncProgressState 列挙体」 540 ページ
- 「ULSyncProgressState 列挙体」 540 ページ

SentDeletes プロパティ

現在までに送信された削除済みのローの数を返します。

構文

Visual Basic
Public Readonly Property **SentDeletes** As Integer

C#
public int **SentDeletes** { get;}

プロパティ値

現在までに送信された削除済みのローの数。

参照

- 「ULSyncProgressData クラス」 528 ページ
- 「ULSyncProgressData メンバ」 528 ページ
- 「ULSyncProgressState 列挙体」 540 ページ
- 「ULSyncProgressState 列挙体」 540 ページ

SentInserts プロパティ

現在までに送信された挿入済みのローの数を返します。

構文

Visual Basic
Public Readonly Property **SentInserts** As Integer

C#
public int **SentInserts** { get;}

プロパティ値

現在までに送信された挿入済みのローの数。

参照

- 「ULSyncProgressData クラス」 528 ページ
- 「ULSyncProgressData メンバ」 528 ページ
- 「ULSyncProgressState 列挙体」 540 ページ
- 「ULSyncProgressState 列挙体」 540 ページ

SentUpdates プロパティ

現在までに送信された更新済みのローの数を返します。

構文

Visual Basic
Public Readonly Property **SentUpdates** As Integer

C#
public int **SentUpdates** { get;}

プロパティ値

現在までに送信された更新済みのローの数。

参照

- 「ULSyncProgressData クラス」 528 ページ
- 「ULSyncProgressData メンバ」 528 ページ
- 「ULSyncProgressState 列挙体」 540 ページ
- 「ULSyncProgressState 列挙体」 540 ページ

State プロパティ

現在の同期のステータスを返します。

構文

Visual Basic
Public Readonly Property **State** As ULSyncProgressState

C#
public ULSyncProgressState **State** { get;}

プロパティ値

現在の同期のステータスを指定する ULSyncProgressState 値の 1 つ。

参照

- 「ULSyncProgressData クラス」 528 ページ
- 「ULSyncProgressData メンバ」 528 ページ
- 「ULSyncProgressState 列挙体」 540 ページ

SyncTableCount プロパティ

同期中のテーブルの数を返します。

構文

Visual Basic
Public Readonly Property **SyncTableCount** As Integer

C#
public int **SyncTableCount** { get;}

プロパティ値

同期中のテーブルの数。テーブルごとに送信と受信のフェーズがあります。したがって、この数は同期されるテーブルの数より多い場合があります。

参照

- 「ULSyncProgressData クラス」 528 ページ
- 「ULSyncProgressData メンバ」 528 ページ
- 「ULSyncProgressState 列挙体」 540 ページ
- 「ULSyncProgressState 列挙体」 540 ページ

SyncTableIndex プロパティ

現在同期中のテーブルのインデックスを返します (テーブルには 1 ～ DatabaseSchema.TableCount までの番号が付けられています)。

構文

Visual Basic
Public Readonly Property **SyncTableIndex** As Integer

C#
public int **SyncTableIndex** { get;}

プロパティ値

現在同期中のテーブルのインデックス (1 ～ SyncTableCount の値)

参照

- 「ULSyncProgressData クラス」 528 ページ
- 「ULSyncProgressData メンバ」 528 ページ
- 「ULSyncProgressState 列挙体」 540 ページ
- 「ULSyncProgressState 列挙体」 540 ページ

TableID プロパティ

現在同期中のテーブルのデータベース・インデックスを返します。

構文

Visual Basic
Public Readonly Property **TableID** As Integer

C#
public int **TableID** { get;}

プロパティ値

データベース・インデックス (1 ～ ULDatabaseSchema.TableCount の値)

参照

- 「ULSyncProgressData クラス」 528 ページ
- 「ULSyncProgressData メンバ」 528 ページ
- 「ULSyncProgressState 列挙体」 540 ページ
- 「ULSyncProgressState 列挙体」 540 ページ
- 「TableCount プロパティ」 268 ページ

TableName プロパティ

現在アップロード中またはダウンロード中のテーブルの名前を返します。

構文

Visual Basic
Public Property **TableName** As String

C#
public string **TableName** { get; set; }

プロパティ値

アップロード中またはダウンロード中のテーブルの名前。該当しない場合は NULL。

setValue メソッド

現在の同期値をこのオブジェクトにロードします。

構文

Visual Basic
Public Sub **SetValues(_**
ByVal *sync_state* As ULSyncProgressState, _
ByVal *table_id* As Integer, _
ByVal *table_count* As Integer, _
ByVal *table_index* As Integer, _
ByVal *sent_bytes* As Long, _
ByVal *sent_inserts* As Integer, _
ByVal *sent_updates* As Integer, _
ByVal *sent_deletes* As Integer, _
ByVal *recv_bytes* As Long, _
ByVal *recv_inserts* As Integer, _
ByVal *recv_updates* As Integer, _
ByVal *recv_deletes* As Integer, _
ByVal *flag_s* As Integer _
)

C#
public void **SetValues(**
ULSyncProgressState *sync_state*,
int *table_id*,
int *table_count*,


```
    int table_index,
    long sent_bytes,
    int sent_inserts,
    int sent_updates,
    int sent_deletes,
    long recv_bytes,
    int recv_inserts,
    int recv_updates,
    int recv_deletes,
    int flag_s
);
```

参照

- 「ULSyncProgressData クラス」 528 ページ
- 「ULSyncProgressData メンバ」 528 ページ

ULSyncProgressListener インタフェース

UL 拡張：同期プログレス・イベントを受信するリスナ・インタフェースです。

構文

Visual Basic
Public Interface **ULSyncProgressListener**

C#
public interface **ULSyncProgressListener**

参照

- 「ULSyncProgressListener メンバ」 538 ページ
- 「Synchronize(ULSyncProgressListener) メソッド」 183 ページ

ULSyncProgressListener メンバ

パブリック・メソッド

メンバ名	説明
「SyncProgressed メソッド」 538 ページ	ユーザに進行状況を通知するために、同期処理中に呼び出されます。このメソッドは、同期をキャンセルする場合は true を、続行する場合は false を返す必要があります。

参照

- 「ULSyncProgressListener インタフェース」 538 ページ
- 「Synchronize(ULSyncProgressListener) メソッド」 183 ページ

SyncProgressed メソッド

ユーザに進行状況を通知するために、同期処理中に呼び出されます。このメソッドは、同期をキャンセルする場合は true を、続行する場合は false を返す必要があります。

構文

Visual Basic
```
Public Function SyncProgressed( _
  ByVal data As ULSyncProgressData _
) As Boolean
```

C#
```
public bool SyncProgressed(
  ULSyncProgressData data
);
```


パラメータ

- **data**　最新の同期のプログレス・データを保持している ULSyncProgressData オブジェクト。

戻り値

このメソッドは、同期をキャンセルする場合は true を、続行する場合は false を返す必要があります。

備考

SyncProgressed の呼び出し中に、Ultra Light.NET API のメソッドを呼び出さないでください。

参照

- 「ULSyncProgressListener インタフェース」 538 ページ
- 「ULSyncProgressListener メンバ」 538 ページ
- 「ULSyncProgressData クラス」 528 ページ

ULSyncProgressState 列挙体

UL 拡張：同期中に発生する可能性のあるすべてのステータスを列挙します。

構文

Visual Basic
Public Enum **ULSyncProgressState**

C#
public enum **ULSyncProgressState**

メンバ

メンバ名	説明	値
STATE_canceled	同期がキャンセルされました。	15
STATE_COMMITTING_DOWNLOAD	ダウンロードをコミットしています。受信された最終的なロー数が、このイベントに含まれます。	9
STATE_CONNECTING	同期ストリームは構築されていますが、まだ開かれていません。	1
STATE_DISCONNECTING	同期ストリームが閉じられようとしています。	11
STATE_DONE	同期は正常に完了しました。	12
STATE_ERROR	同期は完了しましたが、エラーが発生しました。	13
STATE_FINISHING_UPLOAD	アップロードの完了処理中です。送信された最終的なロー数が、このイベントに含まれます。	5
STATE_RECEIVING_DATA	現在のテーブルのデータを受信しています。ULSyncProgressData.ReceivedBytes、ULSyncProgressData.ReceivedInserts、ULSyncProgressData.ReceivedUpdates、および ULSyncProgressData.ReceivedDeletes は更新されました。	8
STATE_RECEIVING_TABLE	テーブルを受信しています。進行状況をモニタするには、ULSyncProgressData.SyncTableIndex と ULSyncProgressData.SyncTableCount を使用します。	7

メンバ名	説明	値
STATE_RECEIVING_UPLOAD_ACK	アップロード完了の確認を受信しています。	6
STATE_ROLLING_BACK_DOWNLOAD	ダウンロード中にエラーが発生したため、同期によってダウンロードがロールバックされています。後続の STATE_ERROR 進行状況レポートにエラーがレポートされます。	14
STATE_SENDING_DATA	現在のテーブルのデータが送信されています。ULSyncProgressData.SentBytes、ULSyncProgressData.SentInserts、ULSyncProgressData.SentUpdates、および ULSyncProgressData.SentDeletes は更新されました。	4
STATE_SENDING_DOWNLOAD_ACK	ダウンロード完了の確認が送信されています。	10
STATE_SENDING_HEADER	同期ストリームが開かれ、ヘッダが送信されようとしています。	2
STATE_SENDING_TABLE	テーブルが送信されています。進行状況をモニタするには、ULSyncProgressData.SyncTableIndex と ULSyncProgressData.SyncTableCount を使用します。	3
STATE_STARTING	同期処理はまだ行われていません。	0

参照

- 「ULSyncProgressData クラス」 528 ページ

ULSyncResult クラス

UL 拡張：前回の同期のステータスを表します。

構文

Visual Basic
Public Class **ULSyncResult**

C#
public class **ULSyncResult**

備考

このクラスにはコンストラクタがありません。各接続には、ULConnection.SyncResult としてアタッチされた、固有の ULSyncResult インスタンスがあります。ULSyncResult インスタンスが有効なのは、接続が開かれている間だけです。

参照

- 「ULSyncResult メンバ」 542 ページ
- 「SyncResult プロパティ」 153 ページ
- 「Synchronize() メソッド」 182 ページ

ULSyncResult メンバ

パブリック・プロパティ

メンバ名	説明
「AuthStatus プロパティ」 543 ページ	前回試行された同期の認証ステータス・コードを返します。
「AuthValue プロパティ」 543 ページ	カスタム・ユーザ認証同期スクリプトからの戻り値を返します。
「IgnoredRows プロパティ」 544 ページ	前回行われた同期で、アップロードされたローが無視されたかどうかを確認します。
「PartialDownloadRetained プロパティ」 544 ページ	前回行われた同期で、部分的なダウンロードが保持されたかどうかを確認します。
「StreamErrorCode プロパティ」 545 ページ	ストリーム自体によってレポートされるエラーを返します。
「StreamErrorParameters プロパティ」 545 ページ	ストリーム・エラー・パラメータをカンマで区切ったリストを返します。

メンバ名	説明
「StreamErrorSystem プロパティ」 546 ページ	ストリーム・エラー・システム固有のコードを返します。
「Timestamp プロパティ」 546 ページ	前回の同期のタイムスタンプを返します。
「UploadOK プロパティ」 547 ページ	前回のアップロード同期が成功したかどうかをチェックします。

参照

- 「ULSyncResult クラス」 542 ページ
- 「SyncResult プロパティ」 153 ページ
- 「Synchronize() メソッド」 182 ページ

AuthStatus プロパティ

前回試行された同期の認証ステータス・コードを返します。

構文

Visual Basic
Public Readonly Property **AuthStatus** As ULAuthStatusCode

C#
public ULAuthStatusCode **AuthStatus** { get;}

プロパティ値

前回行われた同期の認証ステータスを示す ULAuthStatusCode 値の 1 つ。

参照

- 「ULSyncResult クラス」 542 ページ
- 「ULSyncResult メンバ」 542 ページ
- 「ULAuthStatusCode 列挙体」 61 ページ

AuthValue プロパティ

カスタム・ユーザ認証同期スクリプトからの戻り値を返します。

構文

Visual Basic
Public Readonly Property **AuthValue** As Long

C#
public long **AuthValue** { get;}

プロパティ値

カスタム・ユーザ認証同期スクリプトから返された long integer。

参照

- 「ULSyncResult クラス」 542 ページ
- 「ULSyncResult メンバ」 542 ページ

IgnoredRows プロパティ

前回行われた同期で、アップロードされたローが無視されたかどうかを確認します。

構文

Visual Basic
Public Readonly Property **IgnoredRows** As Boolean

C#
public bool **IgnoredRows** { get;}

プロパティ値

前回の同期中にアップロードされたローが無視された場合は true、ローが無視されなかった場合は false。

参照

- 「ULSyncResult クラス」 542 ページ
- 「ULSyncResult メンバ」 542 ページ
- 「DownloadOnly プロパティ」 517 ページ

PartialDownloadRetained プロパティ

前回行われた同期で、部分的なダウンロードが保持されたかどうかを確認します。

構文

Visual Basic
Public Readonly Property **PartialDownloadRetained** As Boolean

C#
public bool **PartialDownloadRetained** { get;}

プロパティ値

ダウンロードが中断され、部分的なダウンロードが保持された場合は true、ダウンロードが中断されなかった場合または部分的なダウンロードがロールバックされた場合は false。

参照

- 「ULSyncResult クラス」 542 ページ
- 「ULSyncResult メンバ」 542 ページ
- 「KeepPartialDownload プロパティ」 518 ページ

StreamErrorCode プロパティ

ストリーム自体によってレポートされるエラーを返します。

構文

Visual Basic
Public Readonly Property **StreamErrorCode** As ULStreamErrorCode

C#
public ULStreamErrorCode **StreamErrorCode** { get;}

プロパティ値

ストリーム自体によってレポートされるエラーを示す ULStreamErrorCode 値の 1 つ。ただし、エラーが発生しなかった場合は ULStreamErrorCode.NONE。

参照

- 「ULSyncResult クラス」 542 ページ
- 「ULSyncResult メンバ」 542 ページ
- 「ULStreamErrorCode 列挙体」 482 ページ
- 「ULStreamErrorCode 列挙体」 482 ページ

StreamErrorParameters プロパティ

ストリーム・エラー・パラメータをカンマで区切ったリストを返します。

構文

Visual Basic
Public Readonly Property **StreamErrorParameters** As String

C#
public string **StreamErrorParameters** { get;}

プロパティ値

「StreamErrorCode プロパティ」 545 ページでレポートされるストリーム・エラー・コードのエラー・パラメータをカンマで区切ったリストが含まれます。エラーにパラメータがない場合、またエラーが設定されていない場合は、空の文字列になります。

参照

- 「ULSyncResult クラス」 542 ページ
- 「ULSyncResult メンバ」 542 ページ
- 「ULStreamErrorCode 列挙体」 482 ページ

StreamErrorSystem プロパティ

ストリーム・エラー・システム固有のコードを返します。

構文

Visual Basic
Public Readonly Property **StreamErrorSystem** As Integer

C#
public int **StreamErrorSystem** { get;}

プロパティ値

ストリーム・エラー・システム固有のコードを示す整数。

参照

- 「ULSyncResult クラス」 542 ページ
- 「ULSyncResult メンバ」 542 ページ

Timestamp プロパティ

前回の同期のタイムスタンプを返します。

構文

Visual Basic
Public Readonly Property **Timestamp** As Date

C#
public DateTime **Timestamp** { get;}

プロパティ値

前回の同期のタイムスタンプを指定する System.DateTime。

参照

- 「ULSyncResult クラス」 542 ページ
- 「ULSyncResult メンバ」 542 ページ
- DateTime

UploadOK プロパティ

前回のアップロード同期が成功したかどうかをチェックします。

構文

Visual Basic
Public Readonly Property **UploadOK** As Boolean

C#
public bool **UploadOK** { get;}

プロパティ値

前回のアップロード同期が成功であった場合は true、不成功であった場合は false。

参照

- 「ULSyncResult クラス」 542 ページ
- 「ULSyncResult メンバ」 542 ページ

ULTable クラス

UL 拡張：Ultra Light データベース内のテーブルを表します。

構文

Visual Basic
Public Class **ULTable**
  Inherits ULResultSet

C#
public class **ULTable**: ULResultSet

備考

このクラスにはコンストラクタがありません。テーブルは、ULCommand の ULCommand.ExecuteTable() を使用して作成されます。

継承：ULResultSet

実装：System.Data.IDataReader、System.Data.IDataRecord、System.IDisposable

参照

- 「ULTable メンバ」 548 ページ
- 「ExecuteTable() メソッド」 125 ページ
- 「ULCommand クラス」 91 ページ
- 「ULResultSet クラス」 415 ページ
- IDataReader
- IDataRecord
- IDisposable

ULTable メンバ

パブリック・プロパティ

メンバ名	説明
「Depth プロパティ」 281 ページ (ULDataReader から継承)	現在のローのネストの深さを返します。最も外側のテーブルの深さは 0 です。
「FieldCount プロパティ」 281 ページ (ULDataReader から継承)	このカーソル内のカラム数を返します。
「HasRows プロパティ」 282 ページ (ULDataReader から継承)	ULDataReader に 1 つまたは複数のローがあるかどうかをチェックします。
「IsBOF プロパティ」 282 ページ (ULDataReader から継承)	**UL 拡張：**現在のローの位置が最初のローの前かどうかをチェックします。

メンバ名	説明
「IsClosed プロパティ」 283 ページ (ULDataReader から継承)	カーソルが現在開いているかどうかを確認します。
「IsEOF プロパティ」 283 ページ (ULDataReader から継承)	**UL 拡張：**現在のローの位置が最後のローの後かどうかをチェックします。
「Item プロパティ」 283 ページ (ULDataReader から継承)	指定されたカラムの値を Object のインスタンスとして取得します。
「RecordsAffected プロパティ」 285 ページ (ULDataReader から継承)	SQL 文の実行によって変更、挿入、または削除されたローの数を返します。SELECT 文または CommandType.TableDirect テーブルの場合、この値は -1 です。
「RowCount プロパティ」 286 ページ (ULDataReader から継承)	**UL 拡張：**カーソル内のローの数を返します。
「Schema プロパティ」 556 ページ	テーブル・スキーマを保持します。このプロパティが有効なのは、接続が開かれている間だけです。
VisibleFieldCount (DbDataReader から継承)	DbDataReader の非表示でないフィールドの数を取得します。

パブリック・メソッド

メンバ名	説明
「AppendBytes メソッド」 422 ページ (ULResultSet から継承)	指定された System.Bytes 配列の指定されたサブセットを、指定された ULDbType.LongBinary カラムの新しい値に追加します。
「AppendChars メソッド」 423 ページ (ULResultSet から継承)	指定された System.Chars 配列の指定されたサブセットを、指定された ULDbType.LongVarchar カラムの新しい値に追加します。
「Close メソッド」 287 ページ (ULDataReader から継承)	カーソルを閉じます。
「Delete メソッド」 425 ページ (ULResultSet から継承)	現在の行を削除します。
「DeleteAllRows メソッド」 557 ページ	テーブルのすべてのローを削除します。

メンバ名	説明
Dispose (DbDataReader から継承)	DbDataReader の現在のインスタンスで使用しているすべてのリソースを解放します。
「FindBegin メソッド」 557 ページ	テーブルで新規に検索を実行する準備を行います。
「FindFirst メソッド」 558 ページ	テーブルを先頭から順方向に移動しながら、現在のインデックスの値かそのセット全体に完全に一致するローを検索します。
「FindLast メソッド」 559 ページ	テーブルを最後から逆方向に移動しながら、現在のインデックスの値またはそのセット全体に完全に一致するローを検索します。
「FindNext メソッド」 561 ページ	現在の位置からテーブルを順方向に移動しながら、次のローが現在のインデックスの値かそのセット全体に完全に一致するかどうかを調べて、ULTable.FindFirst() 検索を続行します。
「FindPrevious メソッド」 563 ページ	現在の位置からテーブルを逆方向に移動しながら、前のローが現在のインデックスの値かそのセット全体に完全に一致するかどうかを調べて、ULTable.FindLast() 検索を続行します。
「GetBoolean メソッド」 287 ページ (ULDataReader から継承)	指定されたカラムの値を System.Boolean として返します。
「GetByte メソッド」 288 ページ (ULDataReader から継承)	指定されたカラムの値を符号なし 8 ビット値 (System.Byte) として返します。
「GetBytes メソッド」 289 ページ (ULDataReader から継承)	**UL 拡張：**指定されたカラムの値を System.Bytes 配列として返します。ULDbType.Binary 型、ULDbType.LongBinary 型、ULDbType.UniqueIdentifier 型のカラムの場合にのみ有効です。
「GetChar メソッド」 291 ページ (ULDataReader から継承)	このメソッドは Ultra Light.NET ではサポートされていません。
「GetChars メソッド」 292 ページ (ULDataReader から継承)	指定されたオフセットで始まる、指定された ULDbType.LongVarchar カラムの値のサブセットを、コピー先の System.Char 配列の指定されたオフセットにコピーします。
GetData (DbDataReader から継承)	要求されたカラムの順序の DbDataReader オブジェクトを返します。

メンバ名	説明
「GetDataTypeName メソッド」 293 ページ (ULDataReader から継承)	指定されたカラムのプロバイダのデータ型の名前を返します。
「GetDateTime メソッド」 294 ページ (ULDataReader から継承)	指定されたカラムの値を、ミリ秒の精度の System.DateTime として返します。
「GetDecimal メソッド」 294 ページ (ULDataReader から継承)	指定されたカラムの値を System.Decimal として返します。
「GetDouble メソッド」 295 ページ (ULDataReader から継承)	指定されたカラムの値を System.Double として返します。
「GetEnumerator メソッド」 296 ページ (ULDataReader から継承)	ULDataReader の反復処理を実行する System.Collections.IEnumerator を返します。
「GetFieldType メソッド」 296 ページ (ULDataReader から継承)	指定されたカラムに最適な System.Type を返します。
「GetFloat メソッド」 297 ページ (ULDataReader から継承)	指定されたカラムの値を System.Single として返します。
「GetGuid メソッド」 298 ページ (ULDataReader から継承)	指定されたカラムの値を UUID (System.Guid) として返します。
「GetInt16 メソッド」 298 ページ (ULDataReader から継承)	指定されたカラムの値を System.Int16 として返します。
「GetInt32 メソッド」 299 ページ (ULDataReader から継承)	指定されたカラムの値を Int32 として返します。
「GetInt64 メソッド」 300 ページ (ULDataReader から継承)	指定されたカラムの値を Int64 として返します。

メンバ名	説明
「GetName メソッド」 300 ページ (ULDataReader から継承)	指定されたカラムの名前を返します。
「GetOrdinal メソッド」 301 ページ (ULDataReader から継承)	指定されたカラムのカラム ID を返します。
GetProviderSpecificFieldType (DbDataReader から継承)	指定されたカラムのプロバイダ固有のフィールド・タイプを返します。
GetProviderSpecificValue (DbDataReader から継承)	指定されたカラムの値を Object のインスタンスとして取得します。
GetProviderSpecificValues (DbDataReader から継承)	現在のローのコレクション内のプロバイダ固有のすべての属性カラムを取得します。
「GetSchemaTable メソッド」 303 ページ (ULDataReader から継承)	ULDataReader のカラムのメタデータが記述された System.Data.DataTable を返します。
「GetString メソッド」 305 ページ (ULDataReader から継承)	指定されたカラムの値を System.String として返します。
「GetTimeSpan メソッド」 306 ページ (ULDataReader から継承)	指定されたカラムの値を、ミリ秒の精度の System.TimeSpan として返します。
「GetUInt16 メソッド」 306 ページ (ULDataReader から継承)	指定されたカラムの値を System.UInt16 として返します。
「GetUInt32 メソッド」 307 ページ (ULDataReader から継承)	指定されたカラムの値を UInt32 として返します。
「GetUInt64 メソッド」 308 ページ (ULDataReader から継承)	指定されたカラムの値を System.UInt64 として返します。
「GetValue メソッド」 308 ページ (ULDataReader から継承)	指定されたカラムの値をネイティブ・フォーマットで返します。

メンバ名	説明
「GetValues メソッド」 309 ページ (ULDataReader から継承)	現在のローのすべてのカラム値を返します。
「Insert メソッド」 564 ページ	現在のカラム値 (set メソッドを使用して指定されます) で新しいローを挿入します。 挿入を行う前に ULTable.InsertBegin を呼び出してください。
「InsertBegin メソッド」 565 ページ	現在のすべてのカラムをデフォルト値に設定して、テーブルに新しいローを挿入する準備を行います。
「IsDBNull メソッド」 310 ページ (ULDataReader から継承)	指定されたカラムの値が NULL かどうかをチェックします。
「LookupBackward メソッド」 565 ページ	テーブルを最後から逆方向に移動しながら、現在のインデックスの値またはそのセット全体に一致するか、その値より小さい値を持つローを検索します。
「LookupBegin メソッド」 567 ページ	テーブルで新規に検索を実行する準備を行います。検索する値は、テーブルを開いたインデックス内のカラムで適切な setType メソッドを呼び出して指定します。
「LookupForward メソッド」 568 ページ	テーブルを最初から順方向に移動しながら、現在のインデックスの値またはそのセット全体に一致するか、その値より大きい値を持つローを検索します。
「MoveAfterLast メソッド」 311 ページ (ULDataReader から継承)	**UL 拡張：**カーソルの最後のローの後に、カーソルを配置します。
「MoveBeforeFirst メソッド」 311 ページ (ULDataReader から継承)	**UL 拡張：**カーソルの最初のローの前に、カーソルを配置します。
「MoveFirst メソッド」 311 ページ (ULDataReader から継承)	**UL 拡張：**カーソルの最初のローに、カーソルを配置します。
「MoveLast メソッド」 312 ページ (ULDataReader から継承)	**UL 拡張：**カーソルの最後のローに、カーソルを配置します。

メンバ名	説明
「MoveNext メソッド」 312 ページ (ULDataReader から継承)	**UL 拡張：**カーソルを次のローに配置します。ただし、カーソルがすでに最後のローにある場合は最後のローの後に配置します。
「MovePrevious メソッド」 313 ページ (ULDataReader から継承)	**UL 拡張：**カーソルを前のローに配置するか、最初のローの前に配置します。
「MoveRelative メソッド」 313 ページ (ULDataReader から継承)	**UL 拡張：**現在のローを基準としてカーソルを配置します。
「NextResult メソッド」 314 ページ (ULDataReader から継承)	バッチ SQL 文の結果を読み込むときに ULDataReader を次の結果に進めます。
「Read メソッド」 314 ページ (ULDataReader から継承)	カーソルを次のローに配置します。ただし、カーソルがすでに最後のローにある場合は最後のローの後に配置します。
「SetBoolean メソッド」 425 ページ (ULResultSet から継承)	指定されたカラムの値を、System.Boolean を使用して設定します。
「SetByte メソッド」 426 ページ (ULResultSet から継承)	指定されたカラムの値を、System.Byte (符号なし 8 ビット整数) を使用して設定します。
「SetBytes メソッド」 427 ページ (ULResultSet から継承)	指定されたカラムの値を、System.Bytes を使用して設定します。
「SetDBNull メソッド」 428 ページ (ULResultSet から継承)	カラムを NULL に設定します。
「SetDateTime メソッド」 429 ページ (ULResultSet から継承)	指定されたカラムの値を、System.DateTime を使用して設定します。
「SetDecimal メソッド」 429 ページ (ULResultSet から継承)	指定されたカラムの値を、System.Decimal を使用して設定します。
「SetDouble メソッド」 430 ページ (ULResultSet から継承)	指定されたカラムの値を、System.Double を使用して設定します。

メンバ名	説明
「SetFloat メソッド」 431 ページ (ULResultSet から継承)	指定されたカラムの値を、System.Single を使用して設定します。
「SetGuid メソッド」 432 ページ (ULResultSet から継承)	指定されたカラムの値を、System.Guid を使用して設定します。
「SetInt16 メソッド」 433 ページ (ULResultSet から継承)	指定されたカラムの値を、System.Int16 を使用して設定します。
「SetInt32 メソッド」 434 ページ (ULResultSet から継承)	指定されたカラムの値を、System.Int32 を使用して設定します。
「SetInt64 メソッド」 435 ページ (ULResultSet から継承)	指定されたカラムの値を、Int64 を使用して設定します。
「SetString メソッド」 436 ページ (ULResultSet から継承)	指定されたカラムの値を、System.String を使用して設定します。
「SetTimeSpan メソッド」 437 ページ (ULResultSet から継承)	指定されたカラムの値を、System.TimeSpan を使用して設定します。
「SetToDefault メソッド」 437 ページ (ULResultSet から継承)	指定されたカラムの値を、そのデフォルト値に設定します。
「SetUInt16 メソッド」 438 ページ (ULResultSet から継承)	指定されたカラムの値を、System.UInt16 を使用して設定します。
「SetUInt32 メソッド」 439 ページ (ULResultSet から継承)	指定されたカラムの値を、System.UInt32 を使用して設定します。
「SetUInt64 メソッド」 440 ページ (ULResultSet から継承)	指定されたカラムの値を、System.UInt64 を使用して設定します。

メンバ名	説明
「Truncate メソッド」 569 ページ	テーブル内のすべてのローを削除し、STOP SYNCHRONIZATION DELETE を一時的にアクティブにします。
「Update メソッド」 441 ページ (ULResultSet から継承)	現在のカラム値 (set メソッドを使用して指定されます) で新しいローを更新します。
「UpdateBegin メソッド」 570 ページ	テーブルの現在のローを更新する準備を行います。

参照

- 「ULTable クラス」 548 ページ
- 「ExecuteTable() メソッド」 125 ページ
- 「ULCommand クラス」 91 ページ
- 「ULResultSet クラス」 415 ページ
- IDataReader
- IDataRecord
- IDisposable

Schema プロパティ

テーブル・スキーマを保持します。このプロパティが有効なのは、接続が開かれている間だけです。

構文

Visual Basic
Public Readonly Property **Schema** As ULTableSchema

C#
public ULTableSchema **Schema** { get;}

プロパティ値

テーブル・スキーマを表す ULTableSchema オブジェクト。

備考

このプロパティは、ULDataReader.GetSchemaTable からの結果に示されない Ultra Light.NET の詳細情報を含め、テーブルの完全なスキーマを表します。

参照

- 「ULTable クラス」 548 ページ
- 「ULTable メンバ」 548 ページ
- 「ULTableSchema クラス」 571 ページ

DeleteAllRows メソッド

テーブルのすべてのローを削除します。

構文

Visual Basic
Public Sub **DeleteAllRows()**

C#
public void **DeleteAllRows();**

備考

アプリケーションによっては、テーブル内のローをすべて削除してから、新しいデータ・セットをテーブルにダウンロードする方が便利なことがあります。
ULConnection.StopSynchronizationDelete を使用すると、統合データベースからは削除しないで Ultra Light データベースからローを削除できます。

参照

- 「ULTable クラス」 548 ページ
- 「ULTable メンバ」 548 ページ
- 「Truncate メソッド」 569 ページ
- 「StopSynchronizationDelete メソッド」 182 ページ

FindBegin メソッド

テーブルで新規に検索を実行する準備を行います。

構文

Visual Basic
Public Sub **FindBegin()**

C#
public void **FindBegin();**

備考

検索する値は、テーブルを開いたインデックス内のカラムで適切な setType メソッドを呼び出して指定します。

参照

- 「ULTable クラス」 548 ページ
- 「ULTable メンバ」 548 ページ
- 「FindFirst() メソッド」 558 ページ
- 「FindFirst(Int16) メソッド」 558 ページ
- 「FindLast() メソッド」 559 ページ
- 「FindLast(Int16) メソッド」 560 ページ

FindFirst メソッド

テーブルを先頭から順方向に移動しながら、現在のインデックスの値かそのセット全体に完全に一致するローを検索します。

FindFirst() メソッド

テーブルを先頭から順方向に移動しながら、現在のインデックスの値かそのセット全体に完全に一致するローを検索します。

構文

Visual Basic
Public Function **FindFirst()** As Boolean

C#
public bool **FindFirst();**

戻り値

成功した場合は true、失敗した場合は false。

備考

検索する値を指定するには、インデックスの各カラムに値を設定します。カーソルは、インデックスの値と完全に一致した最初のローで停止します。失敗すると、カーソル位置は最後のローの後ろ (ULDataReader.IsEOF) になります。

検索を行う前に FindBegin を呼び出してください。

参照

- 「ULTable クラス」 548 ページ
- 「ULTable メンバ」 548 ページ
- 「FindFirst メソッド」 558 ページ
- 「FindBegin メソッド」 557 ページ
- 「FindNext() メソッド」 561 ページ
- 「FindPrevious() メソッド」 563 ページ
- 「FindFirst(Int16) メソッド」 558 ページ
- 「IsEOF プロパティ」 283 ページ
- 「FindBegin メソッド」 557 ページ

FindFirst(Int16) メソッド

テーブルを先頭から順方向に移動しながら、現在のインデックスの値かそのセットの一部に完全に一致するローを検索します。

構文

Visual Basic
```
Public Function FindFirst( _
  ByVal numColumns As Short _
) As Boolean
```

C#
```
public bool FindFirst(
  short numColumns
);
```

パラメータ

- **numColumns**　複合インデックスのための、検索で使用するカラムの数。たとえば、3 つのカラムのインデックスがあり、最初のカラムにのみ基づいて一致する値を検索する場合は、最初のカラムに値を設定してから、値を 1 に指定します。

戻り値

成功した場合は true、失敗した場合は false。

備考

検索する値を指定するには、インデックスの各カラムに値を設定します。カーソルは、インデックスの値と完全に一致した最初のローで停止します。失敗すると、カーソル位置は最後のローの後ろ (ULDataReader.IsEOF) になります。

検索を行う前に FindBegin を呼び出してください。

参照

- 「ULTable クラス」 548 ページ
- 「ULTable メンバ」 548 ページ
- 「FindFirst メソッド」 558 ページ
- 「FindBegin メソッド」 557 ページ
- 「FindNext(Int16) メソッド」 562 ページ
- 「FindPrevious(Int16) メソッド」 564 ページ
- 「FindFirst() メソッド」 558 ページ
- 「IsEOF プロパティ」 283 ページ
- 「FindBegin メソッド」 557 ページ

FindLast メソッド

テーブルを最後から逆方向に移動しながら、現在のインデックスの値またはそのセット全体に完全に一致するローを検索します。

FindLast() メソッド

テーブルを最後から逆方向に移動しながら、現在のインデックスの値またはそのセット全体に完全に一致するローを検索します。

構文

Visual Basic
Public Function **FindLast()** As Boolean

C#
public bool **FindLast();**

戻り値

成功した場合は true、失敗した場合は false。

備考

検索する値を指定するには、インデックスの各カラムに値を設定します。カーソルは、インデックスの値と完全に一致した最初のローで停止します。失敗すると、カーソル位置は最初のローの前 (ULDataReader.IsBOF) になります。

検索を行う前に FindBegin を呼び出してください。

参照

- 「ULTable クラス」 548 ページ
- 「ULTable メンバ」 548 ページ
- 「FindLast メソッド」 559 ページ
- 「FindBegin メソッド」 557 ページ
- 「FindNext() メソッド」 561 ページ
- 「FindPrevious() メソッド」 563 ページ
- 「FindLast(Int16) メソッド」 560 ページ
- 「IsBOF プロパティ」 282 ページ
- 「FindBegin メソッド」 557 ページ

FindLast(Int16) メソッド

テーブルを最後から逆方向に移動しながら、現在のインデックスの値またはそのセットの一部に完全に一致するローを検索します。

構文

Visual Basic
```
Public Function FindLast( _
   ByVal numColumns As Short _
) As Boolean
```

C#
```
public bool FindLast(
   short numColumns
);
```


パラメータ

- **numColumns**　複合インデックスのための、検索で使用するカラムの数。たとえば、3 つのカラムのインデックスがあり、最初のカラムにのみ基づいて一致する値を検索する場合は、最初のカラムに値を設定してから、値を 1 に指定します。

戻り値

成功した場合は true、失敗した場合は false。

備考

検索する値を指定するには、インデックスの各カラムに値を設定します。カーソルは、インデックスの値と完全に一致した最初のローで停止します。失敗すると、カーソル位置は最初のローの前 (ULDataReader.IsBOF) になります。

検索を行う前に FindBegin を呼び出してください。

参照

- 「ULTable クラス」 548 ページ
- 「ULTable メンバ」 548 ページ
- 「FindLast メソッド」 559 ページ
- 「FindBegin メソッド」 557 ページ
- 「FindNext(Int16) メソッド」 562 ページ
- 「FindPrevious(Int16) メソッド」 564 ページ
- 「FindLast() メソッド」 559 ページ
- 「IsBOF プロパティ」 282 ページ
- 「FindBegin メソッド」 557 ページ

FindNext メソッド

現在の位置からテーブルを順方向に移動しながら、次のローが現在のインデックスの値かそのセット全体に完全に一致するかどうかを調べて、ULTable.FindFirst() 検索を続行します。

FindNext() メソッド

現在の位置からテーブルを順方向に移動しながら、次のローが現在のインデックスの値かそのセット全体に完全に一致するかどうかを調べて、ULTable.FindFirst() 検索を続行します。

構文

Visual Basic
Public Function **FindNext()** As Boolean

C#
public bool **FindNext();**

戻り値

成功した場合は true、失敗した場合は false。

備考

インデックスの値と完全に一致すると、カーソルは次のローで停止します。失敗すると、カーソル位置は最後のローの後ろ (ULDataReader.IsEOF) になります。

検索されるカラムの値がローの更新において修正された場合の FindNext の動作は不確定です。

参照

- 「ULTable クラス」 548 ページ
- 「ULTable メンバ」 548 ページ
- 「FindNext メソッド」 561 ページ
- 「FindFirst() メソッド」 558 ページ
- 「FindNext(Int16) メソッド」 562 ページ
- 「IsEOF プロパティ」 283 ページ

FindNext(Int16) メソッド

現在の位置からテーブルを順方向に移動しながら、次のローが現在のインデックスの値かそのセットの一部に完全に一致するかどうかを調べて、ULTable.FindFirst() 検索を続行します。

構文

Visual Basic

```
Public Function FindNext( _
  ByVal numColumns As Short _
) As Boolean
```

C#

```
public bool FindNext(
  short numColumns
);
```

パラメータ

- **numColumns** 複合インデックスのための、検索で使用するカラムの数。たとえば、3 つのカラムのインデックスがあり、最初のカラムにのみ基づいて一致する値を検索する場合は、最初のカラムに値を設定してから、値を 1 に指定します。

戻り値

成功した場合は true、失敗した場合は false。

備考

インデックスの値と完全に一致すると、カーソルは次のローで停止します。失敗すると、カーソル位置は最後のローの後ろ (ULDataReader.IsEOF) になります。

検索されるカラムの値がローの更新において修正された場合の FindNext の動作は不確定です。

参照

- 「ULTable クラス」 548 ページ
- 「ULTable メンバ」 548 ページ
- 「FindNext メソッド」 561 ページ
- 「FindFirst(Int16) メソッド」 558 ページ
- 「FindNext() メソッド」 561 ページ
- 「FindFirst() メソッド」 558 ページ
- 「IsEOF プロパティ」 283 ページ

FindPrevious メソッド

現在の位置からテーブルを逆方向に移動しながら、前のローが現在のインデックスの値かそのセット全体に完全に一致するかどうかを調べて、ULTable.FindLast() 検索を続行します。

FindPrevious() メソッド

現在の位置からテーブルを逆方向に移動しながら、前のローが現在のインデックスの値かそのセット全体に完全に一致するかどうかを調べて、ULTable.FindLast() 検索を続行します。

構文

Visual Basic
Public Function **FindPrevious()** As Boolean

C#
public bool **FindPrevious();**

戻り値

成功した場合は true、失敗した場合は false。

備考

インデックスの値と完全に一致すると、カーソルは前のローで停止します。失敗すると、カーソル位置は最初のローの前 (ULDataReader.IsBOF) になります。

検索されるカラムの値がローの更新において修正された場合の FindPrevious の動作は不確定です。

参照

- 「ULTable クラス」 548 ページ
- 「ULTable メンバ」 548 ページ
- 「FindPrevious メソッド」 563 ページ
- 「FindLast() メソッド」 559 ページ
- 「FindPrevious(Int16) メソッド」 564 ページ
- 「IsBOF プロパティ」 282 ページ

FindPrevious(Int16) メソッド

現在の位置からテーブルを逆方向に移動しながら、前のローが現在のインデックスの値かそのセットの一部に完全に一致するかどうかを調べて、ULTable.FindLast() 検索を続行します。

構文

Visual Basic

```
Public Function FindPrevious( _
  ByVal numColumns As Short _
) As Boolean
```

C#

```
public bool FindPrevious(
  short numColumns
);
```

パラメータ

- **numColumns** 複合インデックスのための、検索で使用するカラムの数。たとえば、3 つのカラムのインデックスがあり、最初のカラムにのみ基づいて一致する値を検索する場合は、最初のカラムに値を設定してから、値を 1 に指定します。

戻り値

成功した場合は true、失敗した場合は false。

備考

インデックスの値と完全に一致すると、カーソルは前のローで停止します。失敗すると、カーソル位置は最初のローの前 (ULDataReader.IsBOF) になります。

検索されるカラムの値がローの更新において修正された場合の FindPrevious の動作は不確定です。

参照

- 「ULTable クラス」 548 ページ
- 「ULTable メンバ」 548 ページ
- 「FindPrevious メソッド」 563 ページ
- 「FindLast() メソッド」 559 ページ
- 「FindLast(Int16) メソッド」 560 ページ
- 「FindPrevious() メソッド」 563 ページ
- 「IsBOF プロパティ」 282 ページ

Insert メソッド

現在のカラム値 (set メソッドを使用して指定されます) で新しいローを挿入します。

挿入を行う前に ULTable.InsertBegin を呼び出してください。

構文

Visual Basic
Public Sub **Insert()**

C#
public void **Insert();**

参照

- 「ULTable クラス」 548 ページ
- 「ULTable メンバ」 548 ページ
- 「InsertBegin メソッド」 565 ページ

InsertBegin メソッド

現在のすべてのカラムをデフォルト値に設定して、テーブルに新しいローを挿入する準備を行います。

構文

Visual Basic
Public Sub **InsertBegin()**

C#
public void **InsertBegin();**

備考

適切な SetType メソッドまたは AppendType メソッドを呼び出して、挿入するデフォルト以外の値を指定します。

insert メソッドが実行されないと、ローが実際に挿入されることも、ロー内のデータが実際に変更されることもありません。また、その変更がコミットされないかぎり、永続化されません。

参照

- 「ULTable クラス」 548 ページ
- 「ULTable メンバ」 548 ページ
- 「Insert メソッド」 564 ページ
- 「Insert メソッド」 564 ページ

LookupBackward メソッド

テーブルを最後から逆方向に移動しながら、現在のインデックスの値またはそのセット全体に一致するか、その値より小さい値を持つローを検索します。

LookupBackward() メソッド

テーブルを最後から逆方向に移動しながら、現在のインデックスの値またはそのセット全体に一致するか、その値より小さい値を持つローを検索します。

構文

Visual Basic
Public Function **LookupBackward()** As Boolean

C#
public bool **LookupBackward();**

戻り値

成功した場合は true、失敗した場合は false。

備考

検索する値を指定するには、インデックスの各カラムに値を設定します。カーソルは、インデックスの値に一致する最初のローか、それより少ない値の最初のローで停止します。失敗した場合 (検索する値より小さい値のローがない場合)、カーソル位置は最初のローの前 (ULDataReader.IsBOF) になります。

検索を行う前に LookupBegin を呼び出してください。

参照

- 「ULTable クラス」 548 ページ
- 「ULTable メンバ」 548 ページ
- 「LookupBackward メソッド」 565 ページ
- 「LookupBegin メソッド」 567 ページ
- 「LookupBackward(Int16) メソッド」 566 ページ
- 「IsBOF プロパティ」 282 ページ

LookupBackward(Int16) メソッド

テーブルを最初から逆方向に移動しながら、現在のインデックスの値またはそのセットの一部に一致するか、その値より小さい値を持つローを検索します。

構文

Visual Basic
```
Public Function LookupBackward( _
  ByVal numColumns As Short _
) As Boolean
```

C#
```
public bool LookupBackward(
  short numColumns
);
```


パラメータ

- **numColumns**　複合インデックスのための、ルックアップで使用するカラムの数。たとえば、3 つのカラムのインデックスがあり、最初のカラムにのみ基づいて一致する値を検索する場合は、最初のカラムに値を設定してから、値を 1 に指定します。

戻り値

成功した場合は true、失敗した場合は false。

備考

検索する値を指定するには、インデックスの各カラムに値を設定します。カーソルは、インデックスの値に一致する最初のローか、それより少ない値の最初のローで停止します。失敗した場合 (検索する値より小さい値のローがない場合)、カーソル位置は最初のローの前 (ULDataReader.IsBOF) になります。

検索を行う前に LookupBegin を呼び出してください。

参照

- 「ULTable クラス」 548 ページ
- 「ULTable メンバ」 548 ページ
- 「LookupBackward メソッド」 565 ページ
- 「LookupBegin メソッド」 567 ページ
- 「LookupBackward(Int16) メソッド」 566 ページ
- 「IsBOF プロパティ」 282 ページ

LookupBegin メソッド

テーブルで新規に検索を実行する準備を行います。検索する値は、テーブルを開いたインデックス内のカラムで適切な setType メソッドを呼び出して指定します。

構文

Visual Basic
Public Sub **LookupBegin()**

C#
public void **LookupBegin();**

参照

- 「ULTable クラス」 548 ページ
- 「ULTable メンバ」 548 ページ
- 「LookupForward() メソッド」 568 ページ
- 「LookupForward(Int16) メソッド」 568 ページ
- 「LookupBackward() メソッド」 566 ページ
- 「LookupBackward(Int16) メソッド」 566 ページ

LookupForward メソッド

テーブルを最初から順方向に移動しながら、現在のインデックスの値またはそのセット全体に一致するか、その値より大きい値を持つローを検索します。

LookupForward() メソッド

テーブルを最初から順方向に移動しながら、現在のインデックスの値またはそのセット全体に一致するか、その値より大きい値を持つローを検索します。

構文

Visual Basic
Public Function **LookupForward()** As Boolean

C#
public bool **LookupForward();**

戻り値

成功した場合は true、失敗した場合は false。

備考

検索する値を指定するには、インデックスの各カラムに値を設定します。カーソルは、インデックスの値に一致するか、それより大きい値の最初のローで停止します。失敗した場合 (検索する値より大きい値のローがない場合)、カーソル位置は最後のローの後ろ (ULDataReader.IsEOF) になります。

検索を行う前に ULTable.LookupBegin() を呼び出してください。

参照

- 「ULTable クラス」 548 ページ
- 「ULTable メンバ」 548 ページ
- 「LookupForward メソッド」 568 ページ
- 「LookupBegin メソッド」 567 ページ
- 「LookupForward(Int16) メソッド」 568 ページ
- 「IsEOF プロパティ」 283 ページ

LookupForward(Int16) メソッド

テーブルを最初から順方向に移動しながら、現在のインデックスの値またはそのセットの一部に一致するか、その値より大きい値を持つローを検索します。

構文

Visual Basic
```
Public Function LookupForward( _
  ByVal numColumns As Short _
) As Boolean
```


C#
public bool **LookupForward(**
  short *numColumns*
);

パラメータ

- **numColumns**　複合インデックスのための、ルックアップで使用するカラムの数。たとえば、3 つのカラムのインデックスがあり、最初のカラムにのみ基づいて一致する値を検索する場合は、最初のカラムに値を設定してから、値を 1 に指定します。

戻り値

成功した場合は true、失敗した場合は false。

備考

検索する値を指定するには、インデックスの各カラムに値を設定します。カーソルは、インデックスの値に一致するか、それより大きい値の最初のローで停止します。失敗した場合 (検索する値より大きい値のローがない場合)、カーソル位置は最後のローの後ろ (ULDataReader.IsEOF) になります。

検索を行う前に LookupBegin を呼び出してください。

参照

- 「ULTable クラス」 548 ページ
- 「ULTable メンバ」 548 ページ
- 「LookupForward メソッド」 568 ページ
- 「LookupBegin メソッド」 567 ページ
- 「LookupForward() メソッド」 568 ページ
- 「IsEOF プロパティ」 283 ページ
- 「LookupBegin メソッド」 567 ページ

Truncate メソッド

テーブル内のすべてのローを削除し、STOP SYNCHRONIZATION DELETE を一時的にアクティブにします。

構文

Visual Basic
Public Sub **Truncate()**

C#
public void **Truncate();**

参照

- 「ULTable クラス」 548 ページ
- 「ULTable メンバ」 548 ページ
- 「DeleteAllRows メソッド」 557 ページ

UpdateBegin メソッド

テーブルの現在のローを更新する準備を行います。

構文

Visual Basic
Public Sub **UpdateBegin()**

C#
public void **UpdateBegin();**

備考

カラム値は、適切な setType メソッドまたは AppendType メソッドを呼び出すことによって修正します。カラムに対する最初の追加では、現在のカラム値がクリアされてから新しい値が追加されます。

ローのデータは、ULResultSet.Update() が呼び出されるまで実際には変更されません。また、その変更は、コミットされないかぎり、永続化されません。

テーブルを開くのに使用されるインデックス内のカラムを修正すると、アクティブな検索処理に予期しない影響を及ぼします。テーブルのプライマリ・キー内のカラムは更新できません。

参照

- 「ULTable クラス」 548 ページ
- 「ULTable メンバ」 548 ページ
- 「Update メソッド」 441 ページ

ULTableSchema クラス

UL 拡張：Ultra Light のテーブルのスキーマを表します。このクラスは継承できません。

構文

Visual Basic
Public NotInheritable Class **ULTableSchema**
  Inherits ULCursorSchema

C#
public sealed class **ULTableSchema**: ULCursorSchema

備考

このクラスにはコンストラクタがありません。ULTableSchema オブジェクトは、その ULTable.Schema としてテーブルにアタッチされます。

継承：ULCursorSchema

参照

- 「ULTableSchema メンバ」 571 ページ
- 「ULTableSchema クラス」 571 ページ
- 「Schema プロパティ」 556 ページ
- 「ULCursorSchema クラス」 233 ページ

ULTableSchema メンバ

パブリック・プロパティ

メンバ名	説明
「ColumnCount プロパティ」 234 ページ (ULCursorSchema から継承)	このカーソル内のカラム数を返します。
「IndexCount プロパティ」 574 ページ	テーブルのインデックス数を返します。
「IsNeverSynchronized プロパティ」 574 ページ	テーブルが、まったく同期されないようにマーク付けされているかどうかをチェックします。
「IsOpen プロパティ」 235 ページ (ULCursorSchema から継承)	カーソルのスキーマが現在開いているかどうかを確認します。
「Name プロパティ」 575 ページ	テーブルの名前を返します。

メンバ名	説明
「PrimaryKey プロパティ」 575 ページ	テーブルのプライマリ・キーのインデックス・スキーマを返します。
「UploadUnchangedRows プロパティ」 576 ページ	データベースが、変更されていないローをアップロードするかどうかをチェックします。

パブリック・メソッド

メンバ名	説明
「GetColumnDefaultValue メソッド」 576 ページ	指定されたカラムのデフォルト値を返します。
「GetColumnID メソッド」 235 ページ (ULCursorSchema から継承)	指定されたカラムのカラム ID を返します。
「GetColumnName メソッド」 236 ページ (ULCursorSchema から継承)	指定されたカラム ID で識別されたカラムの名前を返します。
「GetColumnPartitionSize メソッド」 577 ページ	指定されたカラムに割り当てられているグローバル・オートインクリメントの分割サイズを返します。
「GetColumnPrecision メソッド」 237 ページ (ULCursorSchema から継承)	カラムが数値カラム (SQL NUMERIC 型) の場合は、指定されたカラム ID によって識別されたカラムの精度を返します。
「GetColumnSQLName メソッド」 238 ページ (ULCursorSchema から継承)	指定されたカラム ID で識別されたカラムの名前を返します。
「GetColumnScale メソッド」 239 ページ (ULCursorSchema から継承)	カラムが数値カラム (SQL NUMERIC 型) の場合は、指定されたカラム ID によって識別されたカラムの位取りを返します。
「GetColumnSize メソッド」 239 ページ (ULCursorSchema から継承)	カラムがサイズ指定されたカラム (SQL BINARY 型または CHAR 型) の場合は、指定されたカラム ID によって識別されたカラムのサイズを返します。
「GetColumnULDbType メソッド」 240 ページ (ULCursorSchema から継承)	指定されたカラム ID によって識別されたカラムの Ultra Light.NET データ型を返します。
「GetIndex メソッド」 577 ページ	指定されたインデックスのインデックス・スキーマを返します。

メンバ名	説明
「GetIndexName メソッド」 578 ページ	指定されたインデックス ID で識別されたインデックスの名前を返します。
「GetOptimalIndex メソッド」 579 ページ	指定されたカラムを使用してテーブルを検索するために最適なインデックスです。
「GetPublicationPredicate メソッド」 579 ページ	指定されたパブリケーションに含まれているテーブルのパブリケーション述部を返します。
「GetSchemaTable メソッド」 241 ページ (ULCursorSchema から継承)	ULDataReader のカラムのスキーマが記述された System.Data.DataTable を返します。
「IsColumnAutoIncrement メソッド」 580 ページ	指定されたカラムのデフォルトがオートインクリメントに設定されているかどうかをチェックします。
「IsColumnCurrentDate メソッド」 581 ページ	指定されたカラムのデフォルトが、現在の日付 (ULDbType.Date) に設定されているかどうかをチェックします。
「IsColumnCurrentTime メソッド」 581 ページ	指定されたカラムのデフォルトが、現在の時刻 (ULDbType.Time) に設定されているかどうかをチェックします。
「IsColumnCurrentTimestamp メソッド」 582 ページ	指定されたカラムのデフォルトが、現在のタイムスタンプ (ULDbType.TimeStamp) に設定されているかどうかをチェックします。
「IsColumnGlobalAutoIncrement メソッド」 583 ページ	指定されたカラムのデフォルトがグローバル・オートインクリメントに設定されているかどうかをチェックします。
「IsColumnNewUUID メソッド」 583 ページ	指定されたカラムのデフォルトが新しい UUID (System.Guid) に設定されているかどうかをチェックします。
「IsColumnNullable メソッド」 584 ページ	指定されたカラムが NULL 入力可であるかどうかをチェックします。
「IsInPublication メソッド」 584 ページ	テーブルが、指定されたパブリケーションに含まれているかどうかをチェックします。

参照

- 「ULTableSchema クラス」 571 ページ
- 「ULTableSchema クラス」 571 ページ
- 「Schema プロパティ」 556 ページ
- 「ULCursorSchema クラス」 233 ページ

IndexCount プロパティ

テーブルのインデックス数を返します。

構文

Visual Basic
Public Readonly Property **IndexCount** As Integer

C#
public int **IndexCount** { get;}

プロパティ値

テーブルのインデックスの数。テーブル・スキーマが閉じている場合は 0。

備考

インデックス ID の範囲は、1 ～ IndexCount です。

注意：インデックスの ID とカウントは、スキーマのアップグレード中に変更されることがあります。インデックスを正しく識別するには、名前でアクセスするか、キャッシュされている ID とカウントをスキーマのアップグレード後にリフレッシュします。

参照

- 「ULTableSchema クラス」 571 ページ
- 「ULTableSchema メンバ」 571 ページ

IsNeverSynchronized プロパティ

テーブルが、まったく同期されないようにマーク付けされているかどうかをチェックします。

構文

Visual Basic
Public Readonly Property **IsNeverSynchronized** As Boolean

C#
public bool **IsNeverSynchronized** { get;}

プロパティ値

テーブルが、まったく同期されないようにマーク付けされている場合は true、そうでない場合は false。

備考

まったく同期されないようにマーク付けされているテーブルは、パブリケーションに含まれているものであっても、まったく同期されていません。このようなテーブルは、「非同期」テーブルと呼ばれることもあります。

参照

- 「ULTableSchema クラス」 571 ページ
- 「ULTableSchema メンバ」 571 ページ

Name プロパティ

テーブルの名前を返します。

構文

Visual Basic
Public Overrides Readonly Property **Name** As String

C#
public override string **Name** { get;}

プロパティ値

文字列として返されるテーブル名。

参照

- 「ULTableSchema クラス」 571 ページ
- 「ULTableSchema メンバ」 571 ページ

PrimaryKey プロパティ

テーブルのプライマリ・キーのインデックス・スキーマを返します。

構文

Visual Basic
Public Readonly Property **PrimaryKey** As ULIndexSchema

C#
public ULIndexSchema **PrimaryKey** { get;}

プロパティ値

テーブルのプライマリ・キーを表す ULIndexSchema オブジェクト。

参照

- 「ULTableSchema クラス」 571 ページ
- 「ULTableSchema メンバ」 571 ページ
- 「ULIndexSchema クラス」 353 ページ

UploadUnchangedRows プロパティ

データベースが、変更されていないローをアップロードするかどうかをチェックします。

構文

Visual Basic
Public Readonly Property **UploadUnchangedRows** As Boolean

C#
public bool **UploadUnchangedRows** { get;}

プロパティ値

テーブルが、同期時に常にすべてのローをアップロードするようにマーク付けされている場合は true、変更されたローのみをアップロードするようにマーク付けされている場合は false。

備考

すべてのローをアップロードするようにマーク付けされているテーブルでは、その同期時に、変更されたローと未変更のローがアップロードされます。このようなテーブルは、「完全同期」テーブルと呼ばれることもあります。

参照

- 「ULTableSchema クラス」 571 ページ
- 「ULTableSchema メンバ」 571 ページ

GetColumnDefaultValue メソッド

指定されたカラムのデフォルト値を返します。

構文

Visual Basic
```
Public Function GetColumnDefaultValue( _
  ByVal columnID As Integer _
) As String
```

C#
```
public string  GetColumnDefaultValue(
  int columnID
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULCursorSchema.ColumnCount-1] の範囲内であることが必要です。テーブルの先頭カラムの ID 値は 0 です。

戻り値

指定されたカラムの文字列としてのデフォルト値。デフォルト値が NULL の場合は NULL 参照 (Visual Basic の Nothing)。

参照

- 「ULTableSchema クラス」 571 ページ
- 「ULTableSchema メンバ」 571 ページ
- 「ColumnCount プロパティ」 234 ページ

GetColumnPartitionSize メソッド

指定されたカラムに割り当てられているグローバル・オートインクリメントの分割サイズを返します。

構文

Visual Basic

```
Public Function GetColumnPartitionSize( _
   ByVal columnID As Integer _
) As UInt64
```

C#

```
public ulong GetColumnPartitionSize(
   int columnID
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULCursorSchema.ColumnCount-1] の範囲内であることが必要です。テーブルの先頭カラムの ID 値は 0 です。

戻り値

カラムのグローバル・オートインクリメントの分割サイズ (System.UInt64)。

備考

テーブルのすべてのグローバル・オートインクリメント・カラムは、同じグローバル・オートインクリメントの分割サイズを共有します。

参照

- 「ULTableSchema クラス」 571 ページ
- 「ULTableSchema メンバ」 571 ページ
- 「IsColumnGlobalAutoIncrement メソッド」 583 ページ
- 「ColumnCount プロパティ」 234 ページ
- UInt64

GetIndex メソッド

指定されたインデックスのインデックス・スキーマを返します。

構文

Visual Basic
```
Public Function GetIndex( _
  ByVal name As String _
) As ULIndexSchema
```

C#
```
public ULIndexSchema GetIndex(
  string  name
);
```

パラメータ

- **name**　インデックスの名前。

戻り値

指定されたインデックスを表す ULIndexSchema オブジェクト。

参照

- 「ULTableSchema クラス」 571 ページ
- 「ULTableSchema メンバ」 571 ページ
- 「ULIndexSchema クラス」 353 ページ

GetIndexName メソッド

指定されたインデックス ID で識別されたインデックスの名前を返します。

構文

Visual Basic
```
Public Function GetIndexName( _
  ByVal indexID As Integer _
) As String
```

C#
```
public string  GetIndexName(
  int indexID
);
```

パラメータ

- **indexID**　インデックスの ID。値は、[1,IndexCount] の範囲内であることが必要です。

戻り値

文字列として返されるインデックス名。

備考

インデックスの ID とカウントは、スキーマのアップグレード中に変更されることがあります。インデックスを正しく識別するには、名前でアクセスするか、キャッシュされている ID とカウントをスキーマのアップグレード後にリフレッシュします。

参照

- 「ULTableSchema クラス」 571 ページ
- 「ULTableSchema メンバ」 571 ページ
- 「IndexCount プロパティ」 574 ページ

GetOptimalIndex メソッド

指定されたカラムを使用してテーブルを検索するために最適なインデックスです。

構文

Visual Basic
```
Public Function GetOptimalIndex( _
  ByVal columnID As Integer _
) As ULIndexSchema
```

C#
```
public ULIndexSchema GetOptimalIndex(
  int columnID
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULCursorSchema.ColumnCount>-1] の範囲内であることが必要です。テーブルの先頭カラムの ID 値は 0 です。

戻り値

指定されたカラムの最適なインデックスを表す ULIndexSchema オブジェクト。

備考

指定されたカラムは、インデックス内の最初のカラムですが、インデックスには複数のカラムがある場合があります。

参照

- 「ULTableSchema クラス」 571 ページ
- 「ULTableSchema メンバ」 571 ページ
- 「ColumnCount プロパティ」 234 ページ
- 「ULIndexSchema クラス」 353 ページ

GetPublicationPredicate メソッド

指定されたパブリケーションに含まれているテーブルのパブリケーション述部を返します。

構文

Visual Basic
```
Public Function GetPublicationPredicate( _
```

```
  ByVal pubName As String _
) As String

C#
public string  GetPublicationPredicate(
  string  pubName
);
```

パラメータ

- **pubName**　パブリケーションの名前。

戻り値

文字列として返されるパブリケーション述部。

参照

- 「ULTableSchema クラス」 571 ページ
- 「ULTableSchema メンバ」 571 ページ

IsColumnAutoIncrement メソッド

指定されたカラムのデフォルトがオートインクリメントに設定されているかどうかをチェックします。

構文

```
Visual Basic
Public Function IsColumnAutoIncrement( _
  ByVal columnID As Integer _
) As Boolean

C#
public bool IsColumnAutoIncrement(
  int columnID
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULCursorSchema.ColumnCount-1] の範囲内であることが必要です。テーブルの先頭カラムの ID 値は 0 です。

戻り値

カラムがオートインクリメントされる場合は true、オートインクリメントされない場合は false。

参照

- 「ULTableSchema クラス」 571 ページ
- 「ULTableSchema メンバ」 571 ページ
- 「ColumnCount プロパティ」 234 ページ

IsColumnCurrentDate メソッド

指定されたカラムのデフォルトが、現在の日付 (ULDbType.Date) に設定されているかどうかをチェックします。

構文

Visual Basic
```
Public Function IsColumnCurrentDate( _
  ByVal columnID As Integer _
) As Boolean
```

C#
```
public bool IsColumnCurrentDate(
  int columnID
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULCursorSchema.ColumnCount-1] の範囲内であることが必要です。テーブルの先頭カラムの ID 値は 0 です。

戻り値

カラムのデフォルトが現在の日付に設定されている場合は true、現在の日付に設定されていない場合は false。

参照

- 「ULTableSchema クラス」 571 ページ
- 「ULTableSchema メンバ」 571 ページ
- 「ColumnCount プロパティ」 234 ページ
- 「ULDbType 列挙体」 317 ページ

IsColumnCurrentTime メソッド

指定されたカラムのデフォルトが、現在の時刻 (ULDbType.Time) に設定されているかどうかをチェックします。

構文

Visual Basic
```
Public Function IsColumnCurrentTime( _
  ByVal columnID As Integer _
) As Boolean
```

C#
```
public bool IsColumnCurrentTime(
  int columnID
);
```


パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULCursorSchema.ColumnCount-1] の範囲内であることが必要です。テーブルの先頭カラムの ID 値は 0 です。

戻り値

カラムのデフォルトが現在の時刻に設定されている場合は true、現在の時刻に設定されていない場合は false。

参照

- 「ULTableSchema クラス」 571 ページ
- 「ULTableSchema メンバ」 571 ページ
- 「ColumnCount プロパティ」 234 ページ
- 「ULDbType 列挙体」 317 ページ

IsColumnCurrentTimestamp メソッド

指定されたカラムのデフォルトが、現在のタイムスタンプ (ULDbType.TimeStamp) に設定されているかどうかをチェックします。

構文

Visual Basic

```
Public Function IsColumnCurrentTimestamp( _
  ByVal columnID As Integer _
) As Boolean
```

C#

```
public bool IsColumnCurrentTimestamp(
  int columnID
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULCursorSchema.ColumnCount-1] の範囲内であることが必要です。テーブルの先頭カラムの ID 値は 0 です。

戻り値

カラムのデフォルトが現在のタイムスタンプに設定されている場合は true、現在のタイムスタンプに設定されていない場合は false。

参照

- 「ULTableSchema クラス」 571 ページ
- 「ULTableSchema メンバ」 571 ページ
- 「ColumnCount プロパティ」 234 ページ
- 「ULDbType 列挙体」 317 ページ

IsColumnGlobalAutoIncrement メソッド

指定されたカラムのデフォルトがグローバル・オートインクリメントに設定されているかどうかをチェックします。

構文

Visual Basic

```
Public Function IsColumnGlobalAutoIncrement( _
   ByVal columnID As Integer _
) As Boolean
```

C#

```
public bool IsColumnGlobalAutoIncrement(
   int columnID
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULCursorSchema.ColumnCount-1] の範囲内であることが必要です。テーブルの先頭カラムの ID 値は 0 です。

戻り値

カラムがグローバル・オートインクリメントされる場合は true、グローバル・オートインクリメントされない場合は false。

参照

- 「ULTableSchema クラス」 571 ページ
- 「ULTableSchema メンバ」 571 ページ
- 「GetColumnPartitionSize メソッド」 577 ページ
- 「DatabaseID プロパティ」 148 ページ
- 「ColumnCount プロパティ」 234 ページ

IsColumnNewUUID メソッド

指定されたカラムのデフォルトが新しい UUID (System.Guid) に設定されているかどうかをチェックします。

構文

Visual Basic

```
Public Function IsColumnNewUUID( _
   ByVal columnID As Integer _
) As Boolean
```

C#

```
public bool IsColumnNewUUID(
   int columnID
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULCursorSchema.ColumnCount-1] の範囲内であることが必要です。テーブルの先頭カラムの ID 値は 0 です。

戻り値

カラムのデフォルトが新しい UUID に設定されている場合は true、新しい UUID に設定されていない場合は false。

参照

- 「ULTableSchema クラス」 571 ページ
- 「ULTableSchema メンバ」 571 ページ
- 「ColumnCount プロパティ」 234 ページ
- Guid

IsColumnNullable メソッド

指定されたカラムが NULL 入力可であるかどうかをチェックします。

構文

Visual Basic
```
Public Function IsColumnNullable( _
  ByVal columnID As Integer _
) As Boolean
```

C#
```
public bool IsColumnNullable(
  int columnID
);
```

パラメータ

- **columnID**　カラムの ID 番号。値は、[0,ULCursorSchema.ColumnCount-1] の範囲内であることが必要です。テーブルの先頭カラムの ID 値は 0 です。

戻り値

カラムが NULL 入力可の場合は true、NULL 入力不可の場合は false。

参照

- 「ULTableSchema クラス」 571 ページ
- 「ULTableSchema メンバ」 571 ページ
- 「ColumnCount プロパティ」 234 ページ

IsInPublication メソッド

テーブルが、指定されたパブリケーションに含まれているかどうかをチェックします。

構文

Visual Basic
```
Public Function IsInPublication( _
  ByVal pubName As String _
) As Boolean
```

C#
```
public bool IsInPublication(
  string  pubName
);
```

パラメータ

- **pubName**　パブリケーションの名前。

戻り値

テーブルがパブリケーション内にある場合は true、パブリケーション内にない場合は false。

参照

- 「ULTableSchema クラス」 571 ページ
- 「ULTableSchema メンバ」 571 ページ

ULTransaction クラス

SQL トランザクションを表します。このクラスは継承できません。

構文

Visual Basic
Public NotInheritable Class **ULTransaction**
  Inherits DbTransaction

C#
public sealed class **ULTransaction**: DbTransaction

備考

ULTransaction にはコンストラクタがありません。ULTransaction オブジェクトを取得するには、ULConnection.BeginTransaction() を使用します。コマンドをトランザクションに関連付けるには、ULCommand.Transaction を使用します。

トランザクションがコミットまたはロールバックされると、接続はすべての操作の実行時にそれらを自動的にコミットする状態に戻ります。さらに操作をグループ化するには、新しいトランザクションを作成する必要があります。

継承：System.Data.Common.DbTransaction

実装：System.Data.IDbTransaction、System.IDisposable

参照

- 「ULTransaction メンバ」 586 ページ
- 「BeginTransaction() メソッド」 153 ページ
- 「Transaction プロパティ」 104 ページ
- DbTransaction
- IDbTransaction
- IDisposable

ULTransaction メンバ

パブリック・プロパティ

メンバ名	説明
「Connection プロパティ」 587 ページ	トランザクションに対応する接続を返します。
「IsolationLevel プロパティ」 588 ページ	トランザクションの独立性レベルを返します。

パブリック・メソッド

メンバ名	説明
「Commit メソッド」 588 ページ	データベース・トランザクションをコミットします。
Dispose (DbTransaction から継承)	DbTransaction によって使用されるアンマネージ・リソースを解放します。
「Rollback メソッド」 589 ページ	データベースへのトランザクションの未処理の変更をロールバックします。

参照

- 「ULTransaction クラス」 586 ページ
- 「BeginTransaction() メソッド」 153 ページ
- 「Transaction プロパティ」 104 ページ
- DbTransaction
- IDbTransaction
- IDisposable

Connection プロパティ

トランザクションに対応する接続を返します。

構文

Visual Basic
Public Readonly Property **Connection** As ULConnection

C#
public ULConnection **Connection** { get;}

プロパティ値

トランザクションに関連付けられている ULConnection オブジェクト。トランザクションが無効な場合は NULL 参照 (Visual Basic の Nothing)。

備考

これは、System.Data.IDbTransaction.Connection と System.Data.Common.DbCommand.Connection が厳密に型指定されたものです。

参照

- 「ULTransaction クラス」 586 ページ
- 「ULTransaction メンバ」 586 ページ
- 「BeginTransaction() メソッド」 153 ページ
- 「ULConnection クラス」 139 ページ
- IDbTransaction.Connection
- DbCommand.Connection

IsolationLevel プロパティ

トランザクションの独立性レベルを返します。

構文

Visual Basic
Public Overrides Readonly Property **IsolationLevel** As IsolationLevel

C#
public override IsolationLevel **IsolationLevel** { get;}

プロパティ値

System.Data.IsolationLevel 値の 1 つ。Ultra Light.NET では、System.Data.IsolationLevel.ReadUncommitted のみがサポートされています。

参照

- 「ULTransaction クラス」 586 ページ
- 「ULTransaction メンバ」 586 ページ
- 「BeginTransaction() メソッド」 153 ページ
- IsolationLevel
- IsolationLevel.ReadUncommitted

Commit メソッド

データベース・トランザクションをコミットします。

構文

Visual Basic
Public Overrides Sub **Commit()**

C#
public override void **Commit();**

備考

トランザクションがコミットまたはロールバックされると、接続はすべての操作の実行時にそれらを自動的にコミットする状態に戻ります。さらに操作をグループ化するには、新しいトランザクションを作成する必要があります。

データベース・エラーが原因で Commit() が失敗すると (たとえば、参照整合性エラーなど)、トランザクションはアクティブなまま残ります。問題を訂正し、もう一度 Commit() メソッドを呼び出します。または、ULTransaction.Rollback() を呼び出してトランザクションを完了させます。

参照

- 「ULTransaction クラス」 586 ページ
- 「ULTransaction メンバ」 586 ページ
- 「Rollback メソッド」 589 ページ

Rollback メソッド

データベースへのトランザクションの未処理の変更をロールバックします。

構文

Visual Basic
Public Overrides Sub **Rollback()**

C#
public override void **Rollback();**

備考

トランザクションがコミットまたはロールバックされると、接続はすべての操作の実行時にそれらを自動的にコミットする状態に戻ります。さらに操作をグループ化するには、新しいトランザクションを作成する必要があります。

参照

- 「ULTransaction クラス」 586 ページ
- 「ULTransaction メンバ」 586 ページ
- 「Commit メソッド」 588 ページ

索引

A

Abort プロパティ
  ULRowsCopiedEventArgs クラス [Ultra Light .NET API], 446
ActiveSyncInvoked メソッド
  ULActiveSyncListener インタフェース [Ultra Light .NET API], 57
ActiveSync 同期
  Ultra Light.NET, 31
Add(Int32, Int32) メソッド
  ULBulkCopyColumnMappingCollection クラス [Ultra Light .NET API], 84
Add(Int32, String) メソッド
  ULBulkCopyColumnMappingCollection クラス [Ultra Light .NET API], 84
Add(Object) メソッド
  ULParameterCollection クラス [Ultra Light .NET API], 397
Add(String, Int32) メソッド
  ULBulkCopyColumnMappingCollection クラス [Ultra Light .NET API], 85
Add(String, Object) メソッド
  ULParameterCollection クラス [Ultra Light .NET API], 399
Add(String, String) メソッド
  ULBulkCopyColumnMappingCollection クラス [Ultra Light .NET API], 86
Add(String, ULDbType, Int32, String) メソッド
  ULParameterCollection クラス [Ultra Light .NET API], 402
Add(String, ULDbType, Int32) メソッド
  ULParameterCollection クラス [Ultra Light .NET API], 401
Add(String, ULDbType) メソッド
  ULParameterCollection クラス [Ultra Light .NET API], 400
Add(ULBulkCopyColumnMapping) メソッド
  ULBulkCopyColumnMappingCollection クラス [Ultra Light .NET API], 83
Add(ULParameter) メソッド
  ULParameterCollection クラス [Ultra Light .NET API], 398
AdditionalParms プロパティ
  ULConnectionParms クラス [Ultra Light .NET API], 192
  ULSyncParms クラス [Ultra Light .NET API], 516
AddRange(Array) メソッド
  ULParameterCollection クラス [Ultra Light .NET API], 404
AddRange(ULParameter[]) メソッド
  ULParameterCollection クラス [Ultra Light .NET API], 404
AppendBytes メソッド
  ULResultSet クラス [Ultra Light .NET API], 422
AppendChars メソッド
  ULResultSet クラス [Ultra Light .NET API], 423
AuthenticationParms プロパティ
  ULFileTransfer クラス [Ultra Light .NET API], 336
  ULSyncParms クラス [Ultra Light .NET API], 517
AuthStatus プロパティ
  ULFileTransfer クラス [Ultra Light .NET API], 335
  ULSyncResult クラス [Ultra Light .NET API], 543
AuthValue プロパティ
  ULFileTransfer クラス [Ultra Light .NET API], 335
  ULSyncResult クラス [Ultra Light .NET API], 543
AutoCommit モード
  Ultra Light.NET 開発, 26

B

BatchSize プロパティ
  ULBulkCopy クラス [Ultra Light .NET API], 66
BeginExecuteNonQuery(AsyncCallback, Object) メソッド
  ULCommand クラス [Ultra Light .NET API], 106
BeginExecuteNonQuery メソッド
  ULCommand クラス [Ultra Light .NET API], 105
BeginExecuteReader(AsyncCallback, Object, CommandBehavior) メソッド
  ULCommand クラス [Ultra Light .NET API], 109
BeginExecuteReader(AsyncCallback, Object) メソッド
  ULCommand クラス [Ultra Light .NET API], 108
BeginExecuteReader(CommandBehavior) メソッド

ULCommand クラス [Ultra Light .NET API], 107
BeginExecuteReader メソッド
ULCommand クラス [Ultra Light .NET API], 107
BeginTransaction(IsolationLevel) メソッド
ULConnection クラス [Ultra Light .NET API], 154
BeginTransaction メソッド
ULConnection クラス [Ultra Light .NET API], 153
BulkCopyTimeout プロパティ
ULBulkCopy クラス [Ultra Light .NET API], 67
BytesReceived プロパティ
ULFileTransferProgressData クラス [Ultra Light .NET API], 349

C

CacheSize プロパティ
ULConnectionParms クラス [Ultra Light .NET API], 196
ULConnectionStringBuilder クラス [Ultra Light .NET API], 208
CancelGetNotification メソッド
ULConnection クラス [Ultra Light .NET API], 155
Cancel メソッド
ULCommand クラス [Ultra Light .NET API], 110
CanCreateDataSourceEnumerator プロパティ
ULFactory クラス [Ultra Light .NET API], 328
CaseSensitive プロパティ
ULCreateParms クラス [Ultra Light .NET API], 224
ChangeDatabase メソッド
ULConnection クラス [Ultra Light .NET API], 156
ChangeEncryptionKey メソッド
ULConnection クラス [Ultra Light .NET API], 157
ChangePassword メソッド
ULConnection クラス [Ultra Light .NET API], 157
ChecksumLevel プロパティ
ULCreateParms クラス [Ultra Light .NET API], 224
Clear メソッド
ULParameterCollection クラス [Ultra Light .NET API], 405
Close メソッド
ULBulkCopy クラス [Ultra Light .NET API], 69
ULConnection クラス [Ultra Light .NET API], 158
ULDataReader クラス [Ultra Light .NET API], 287
CollationName プロパティ
ULDatabaseSchema クラス [Ultra Light .NET API], 266
ColumnCount プロパティ
ULCursorSchema クラス [Ultra Light .NET API], 234
ULIndexSchema クラス [Ultra Light .NET API], 354
ColumnMappings プロパティ
ULBulkCopy クラス [Ultra Light .NET API], 68
Columns プロパティ
ULMetaDataCollectionNames クラス [Ultra Light .NET API], 367
CommandText プロパティ
ULCommand クラス [Ultra Light .NET API], 99
CommandTimeout プロパティ
ULCommand クラス [Ultra Light .NET API], 100
CommandType プロパティ
ULCommand クラス [Ultra Light .NET API], 100
Command プロパティ
ULRowUpdatedEventArgs クラス [Ultra Light .NET API], 451
ULRowUpdatingEventArgs クラス [Ultra Light .NET API], 456
Commit メソッド
ULTransaction クラス [Ultra Light .NET API], 588
ConnectionName プロパティ
ULConnectionParms クラス [Ultra Light .NET API], 197
ULConnectionStringBuilder クラス [Ultra Light .NET API], 209
ConnectionString プロパティ
ULConnection クラス [Ultra Light .NET API], 146
ConnectionTimeout プロパティ
ULConnection クラス [Ultra Light .NET API], 147
Connection クラス
Ultra Light.NET, 13
Connection プロパティ
ULCommand クラス [Ultra Light .NET API], 101

ULTransaction クラス [Ultra Light .NET API], 587
Contains(Object) メソッド
ULParameterCollection クラス [Ultra Light .NET API], 405
Contains(String) メソッド
ULParameterCollection クラス [Ultra Light .NET API], 406
ContainsKey メソッド
ULConnectionStringBuilder クラス [Ultra Light .NET API], 217
Contains メソッド
ULBulkCopyColumnMappingCollection クラス [Ultra Light .NET API], 87
CopyFrom メソッド
ULSyncParms クラス [Ultra Light .NET API], 527
CopyTo メソッド
ULBulkCopyColumnMappingCollection クラス [Ultra Light .NET API], 87
ULParameterCollection クラス [Ultra Light .NET API], 406
CountUploadRows(String, Int64) メソッド
ULConnection クラス [Ultra Light .NET API], 159
CountUploadRows(String, UInt32) メソッド
ULConnection クラス [Ultra Light .NET API], 159
Count プロパティ
ULParameterCollection クラス [Ultra Light .NET API], 394
CreateCommandBuilder メソッド
ULFactory クラス [Ultra Light .NET API], 329
CreateCommand メソッド
ULConnection クラス [Ultra Light .NET API], 160
ULFactory クラス [Ultra Light .NET API], 329
CreateConnectionStringBuilder メソッド
ULFactory クラス [Ultra Light .NET API], 330
CreateConnection メソッド
ULFactory クラス [Ultra Light .NET API], 330
CreateDataAdapter メソッド
ULFactory クラス [Ultra Light .NET API], 330
CreateDatabase メソッド
ULDatabaseManager クラス [Ultra Light .NET API], 257
CreateNotificationQueue メソッド
ULConnection クラス [Ultra Light .NET API], 161
CreateParameter メソッド
ULCommand クラス [Ultra Light .NET API], 111
ULFactory クラス [Ultra Light .NET API], 331
CurrentScript プロパティ
ULSqlProgressData クラス [Ultra Light .NET API], 479

D

DataAdapter プロパティ
ULCommandBuilder クラス [Ultra Light .NET API], 133
DatabaseID プロパティ
ULConnection クラス [Ultra Light .NET API], 148
DatabaseKey プロパティ
ULConnectionStringBuilder クラス [Ultra Light .NET API], 209
DatabaseManager クラス
Ultra Light.NET, 13
DatabaseManager プロパティ
ULConnection クラス [Ultra Light .NET API], 149
DatabaseName プロパティ
ULConnectionStringBuilder クラス [Ultra Light .NET API], 210
DatabaseOnCE プロパティ
ULConnectionParms クラス [Ultra Light .NET API], 197
ULConnectionStringBuilder クラス [Ultra Light .NET API], 211
DatabaseOnDesktop プロパティ
ULConnectionParms クラス [Ultra Light .NET API], 197
ULConnectionStringBuilder クラス [Ultra Light .NET API], 211
Database プロパティ
ULConnection クラス [Ultra Light .NET API], 148
DataSourceInformation プロパティ
ULMetaDataCollectionNames クラス [Ultra Light .NET API], 368
DataSource プロパティ
ULConnection クラス [Ultra Light .NET API], 147
DataTypes プロパティ

ULMetaDataCollectionNames クラス [Ultra Light .NET API], 368
DateFormat プロパティ
ULCreateParms クラス [Ultra Light .NET API], 225
DateOrder プロパティ
ULCreateParms クラス [Ultra Light .NET API], 225
DbType プロパティ
ULParameter クラス [Ultra Light .NET API], 383
DCX
説明, viii
DeclareEvent メソッド
ULConnection クラス [Ultra Light .NET API], 162
DeleteAllRows メソッド
ULTable クラス [Ultra Light .NET API], 557
DeleteCommand プロパティ
ULDataAdapter クラス [Ultra Light .NET API], 248
Delete メソッド
ULResultSet クラス [Ultra Light .NET API], 425
Depth プロパティ
ULDataReader クラス [Ultra Light .NET API], 281
DesignTimeVisible プロパティ
ULCommand クラス [Ultra Light .NET API], 102
DestinationColumn プロパティ
ULBulkCopyColumnMapping クラス [Ultra Light .NET API], 78
DestinationFileName プロパティ
ULFileTransfer クラス [Ultra Light .NET API], 336
DestinationOrdinal プロパティ
ULBulkCopyColumnMapping クラス [Ultra Light .NET API], 79
DestinationPath プロパティ
ULFileTransfer クラス [Ultra Light .NET API], 337
DestinationTableName プロパティ
ULBulkCopy クラス [Ultra Light .NET API], 68
DestroyNotificationQueue メソッド
ULConnection クラス [Ultra Light .NET API], 163
Direction プロパティ
ULParameter クラス [Ultra Light .NET API], 384
Dispose メソッド
ULBulkCopy クラス [Ultra Light .NET API], 70
DML
Ultra Light.NET, 16
DocCommentXchange (DCX)
説明, viii
DownloadedFile プロパティ
ULFileTransfer クラス [Ultra Light .NET API], 337
DownloadFile(ULFileTransferProgressListener) メソッド
ULFileTransfer クラス [Ultra Light .NET API], 345
DownloadFile メソッド
ULFileTransfer クラス [Ultra Light .NET API], 344
DownloadOnly プロパティ
ULSyncParms クラス [Ultra Light .NET API], 517
DropDatabase メソッド
ULDatabaseManager クラス [Ultra Light .NET API], 259

E

EncryptionKey プロパティ
ULConnectionParms クラス [Ultra Light .NET API], 198
EndExecuteNonQuery メソッド
ULCommand クラス [Ultra Light .NET API], 111
EndExecuteReader メソッド
ULCommand クラス [Ultra Light .NET API], 115
EquivalentTo メソッド
ULConnectionStringBuilder クラス [Ultra Light .NET API], 217
ExecuteNextSQLPassthroughScript メソッド
ULConnection クラス [Ultra Light .NET API], 164
ExecuteNonQuery メソッド
ULCommand クラス [Ultra Light .NET API], 118
ExecuteReader(CommandBehavior) メソッド
ULCommand クラス [Ultra Light .NET API], 120
ExecuteReader メソッド
ULCommand クラス [Ultra Light .NET API], 119
ExecuteResultSet(CommandBehavior) メソッド
ULCommand クラス [Ultra Light .NET API], 123
ExecuteResultSet メソッド
ULCommand クラス [Ultra Light .NET API], 122
ExecuteScalar メソッド

ULCommand クラス [Ultra Light .NET API], 124
ExecuteSQLPassthroughScripts メソッド
ULConnection クラス [Ultra Light .NET API], 164
ExecuteTable(CommandBehavior) メソッド
ULCommand クラス [Ultra Light .NET API], 126
ExecuteTable(String, String, CommandBehavior) メソッド
ULConnection クラス [Ultra Light .NET API], 168
ExecuteTable(String, String) メソッド
ULConnection クラス [Ultra Light .NET API], 166
ExecuteTable(String) メソッド
ULConnection クラス [Ultra Light .NET API], 165
ExecuteTable メソッド
ULCommand クラス [Ultra Light .NET API], 125
Explorer、Ultra Light 用 (*参照* SQL Anywhere Explorer、Ultra Light 用)

F

FieldCount プロパティ
ULDataReader クラス [Ultra Light .NET API], 281
FileAuthCode プロパティ
ULFileTransfer クラス [Ultra Light .NET API], 338
FileName プロパティ
ULFileTransfer クラス [Ultra Light .NET API], 338
FileSize プロパティ
ULFileTransferProgressData クラス [Ultra Light .NET API], 349
FileTransferProgressed メソッド
ULFileTransferProgressListener インタフェース [Ultra Light .NET API], 351
FindBegin メソッド
ULTable クラス [Ultra Light .NET API], 557
FindFirst(Int16) メソッド
ULTable クラス [Ultra Light .NET API], 558
FindFirst メソッド
ULTable クラス [Ultra Light .NET API], 558
FindLast(Int16) メソッド
ULTable クラス [Ultra Light .NET API], 560
FindLast メソッド
ULTable クラス [Ultra Light .NET API], 559
FindNext(Int16) メソッド
ULTable クラス [Ultra Light .NET API], 562
FindNext メソッド
ULTable クラス [Ultra Light .NET API], 561
FindPrevious(Int16) メソッド
ULTable クラス [Ultra Light .NET API], 564
FindPrevious メソッド
ULTable クラス [Ultra Light .NET API], 563
FIPS プロパティ
ULCreateParms クラス [Ultra Light .NET API], 226
FLAG_IS_BLOCKING フィールド
ULFileTransferProgressData クラス [Ultra Light .NET API], 349
ULSyncProgressData クラス [Ultra Light .NET API], 529
Flags プロパティ
ULFileTransferProgressData クラス [Ultra Light .NET API], 350
ULSyncProgressData クラス [Ultra Light .NET API], 530
ForceDownload プロパティ
ULFileTransfer クラス [Ultra Light .NET API], 339
ForeignKeys プロパティ
ULMetaDataCollectionNames クラス [Ultra Light .NET API], 369

G

GetBoolean メソッド
ULDataReader クラス [Ultra Light .NET API], 287
GetBytes(Int32, Int64, Byte[], Int32, Int32) メソッド
ULDataReader クラス [Ultra Light .NET API], 289
GetBytes(Int32) メソッド
ULDataReader クラス [Ultra Light .NET API], 290
GetByte メソッド
ULDataReader クラス [Ultra Light .NET API], 288
GetChars メソッド
ULDataReader クラス [Ultra Light .NET API], 292
GetChar メソッド
ULDataReader クラス [Ultra Light .NET API], 291

GetColumnDefaultValue メソッド
ULTableSchema クラス [Ultra Light .NET API], 576
GetColumnID メソッド
ULCursorSchema クラス [Ultra Light .NET API], 235
GetColumnName メソッド
ULCursorSchema クラス [Ultra Light .NET API], 236
ULIndexSchema クラス [Ultra Light .NET API], 359
GetColumnPartitionSize メソッド
ULTableSchema クラス [Ultra Light .NET API], 577
GetColumnPrecision メソッド
ULCursorSchema クラス [Ultra Light .NET API], 237
GetColumnScale メソッド
ULCursorSchema クラス [Ultra Light .NET API], 239
GetColumnSize メソッド
ULCursorSchema クラス [Ultra Light .NET API], 239
GetColumnSQLName メソッド
ULCursorSchema クラス [Ultra Light .NET API], 238
GetColumnULDbType メソッド
ULCursorSchema クラス [Ultra Light .NET API], 240
GetDatabaseProperty メソッド
ULDatabaseSchema クラス [Ultra Light .NET API], 269
GetDataTypeName メソッド
ULDataReader クラス [Ultra Light .NET API], 293
GetDateTime メソッド
ULDataReader クラス [Ultra Light .NET API], 294
GetDecimal メソッド
ULDataReader クラス [Ultra Light .NET API], 294
GetDeleteCommand(Boolean) メソッド
ULCommandBuilder クラス [Ultra Light .NET API], 133
GetDeleteCommand メソッド
ULCommandBuilder クラス [Ultra Light .NET API], 134
GetDouble メソッド
ULDataReader クラス [Ultra Light .NET API], 295
GetEnumerator メソッド
ULDataReader クラス [Ultra Light .NET API], 296
ULParameterCollection クラス [Ultra Light .NET API], 407
GetFieldType メソッド
ULDataReader クラス [Ultra Light .NET API], 296
GetFillParameters メソッド
ULDataAdapter クラス [Ultra Light .NET API], 252
GetFloat メソッド
ULDataReader クラス [Ultra Light .NET API], 297
GetGuid メソッド
ULDataReader クラス [Ultra Light .NET API], 298
GetIndexName メソッド
ULTableSchema クラス [Ultra Light .NET API], 578
GetIndex メソッド
ULTableSchema クラス [Ultra Light .NET API], 577
GetInsertCommand(Boolean) メソッド
ULCommandBuilder クラス [Ultra Light .NET API], 135
GetInsertCommand メソッド
ULCommandBuilder クラス [Ultra Light .NET API], 136
GetInt16 メソッド
ULDataReader クラス [Ultra Light .NET API], 298
GetInt32 メソッド
ULDataReader クラス [Ultra Light .NET API], 299
GetInt64 メソッド
ULDataReader クラス [Ultra Light .NET API], 300
GetLastDownloadTime メソッド
ULConnection クラス [Ultra Light .NET API], 169
GetName メソッド
ULDataReader クラス [Ultra Light .NET API], 300

GetNewUUID メソッド
ULConnection クラス [Ultra Light .NET API], 170
GetNotificationParameter メソッド
ULConnection クラス [Ultra Light .NET API], 172
GetNotification メソッド
ULConnection クラス [Ultra Light .NET API], 171
GetObjectData メソッド
ULException クラス [Ultra Light .NET API], 325
GetOptimalIndex メソッド
ULTableSchema クラス [Ultra Light .NET API], 579
GetOrdinal メソッド
ULDataReader クラス [Ultra Light .NET API], 301
GetPublicationName メソッド
ULDatabaseSchema クラス [Ultra Light .NET API], 271
GetPublicationPredicate メソッド
ULTableSchema クラス [Ultra Light .NET API], 579
GetPublicationSchema メソッド
ULDatabaseSchema クラス [Ultra Light .NET API], 272
GetRowCount メソッド
ULDataReader クラス [Ultra Light .NET API], 302
GetSchema(String, String[]) メソッド
ULConnection クラス [Ultra Light .NET API], 174
GetSchema(String) メソッド
ULConnection クラス [Ultra Light .NET API], 174
GetSchemaTable メソッド
ULCursorSchema クラス [Ultra Light .NET API], 241
ULDataReader クラス [Ultra Light .NET API], 303
GetSchema メソッド
ULConnection クラス [Ultra Light .NET API], 173
GetShortName メソッド
ULConnectionStringBuilder クラス [Ultra Light .NET API], 218
GetSQLPassthroughScriptCount メソッド
ULConnection クラス [Ultra Light .NET API], 173
GetString メソッド
ULDataReader クラス [Ultra Light .NET API], 305
GetTableName メソッド
ULDatabaseSchema クラス [Ultra Light .NET API], 273
GetTimeSpan メソッド
ULDataReader クラス [Ultra Light .NET API], 306
GetUInt16 メソッド
ULDataReader クラス [Ultra Light .NET API], 306
GetUInt32 メソッド
ULDataReader クラス [Ultra Light .NET API], 307
GetUInt64 メソッド
ULDataReader クラス [Ultra Light .NET API], 308
GetUpdateCommand(Boolean) メソッド
ULCommandBuilder クラス [Ultra Light .NET API], 137
GetUpdateCommand メソッド
ULCommandBuilder クラス [Ultra Light .NET API], 138
GetValues メソッド
ULDataReader クラス [Ultra Light .NET API], 309
GetValue メソッド
ULDataReader クラス [Ultra Light .NET API], 308
GlobalAutoIncrementUsage プロパティ
ULConnection クラス [Ultra Light .NET API], 149
grantConnectTo メソッド
Ultra Light.NET 開発, 29
GrantConnectTo メソッド
ULConnection クラス [Ultra Light .NET API], 176

H

HasRows プロパティ
ULDataReader クラス [Ultra Light .NET API], 282

I

iAnywhere.Data.UltraLite ネームスペース
クラス [Ultra Light .NET API], 56
説明, 2
iAnywhere.Data.UltraLite ネームスペース (.NET 2.0)
iAnywhere.Data.UltraLite ネームスペース, 2
iAnywhere デベロッパー・コミュニティ
ニュースグループ, xiv
IgnoredRows プロパティ
ULSyncResult クラス [Ultra Light .NET API], 544
IndexColumns プロパティ
ULMetaDataCollectionNames クラス [Ultra Light .NET API], 370
IndexCount プロパティ
ULTableSchema クラス [Ultra Light .NET API], 574
Indexes プロパティ
ULMetaDataCollectionNames クラス [Ultra Light .NET API], 370
IndexName プロパティ
ULCommand クラス [Ultra Light .NET API], 102
IndexOf(Object) メソッド
ULParameterCollection クラス [Ultra Light .NET API], 407
IndexOf(String) メソッド
ULParameterCollection クラス [Ultra Light .NET API], 408
IndexOf メソッド
ULBulkCopyColumnMappingCollection クラス [Ultra Light .NET API], 88
IndexSchema クラス
Ultra Light.NET 開発, 27
InfoMessage イベント
ULConnection クラス [Ultra Light .NET API], 185
InsertBegin メソッド
ULTable クラス [Ultra Light .NET API], 565
InsertCommand プロパティ
ULDataAdapter クラス [Ultra Light .NET API], 249
Insert メソッド
ULParameterCollection クラス [Ultra Light .NET API], 409
ULTable クラス [Ultra Light .NET API], 564
install-dir
マニュアルの使用方法, xi
Instance フィールド
ULFactory クラス [Ultra Light .NET API], 328
Interactive SQL
Visual Studio から開く, 7
INVALID_DATABASE_ID フィールド
ULConnection クラス [Ultra Light .NET API], 145
IsBOF プロパティ
ULDataReader クラス [Ultra Light .NET API], 282
IsCaseSensitive プロパティ
ULDatabaseSchema クラス [Ultra Light .NET API], 267
IsClosed プロパティ
ULDataReader クラス [Ultra Light .NET API], 283
IsColumnAutoIncrement メソッド
ULTableSchema クラス [Ultra Light .NET API], 580
IsColumnCurrentDate メソッド
ULTableSchema クラス [Ultra Light .NET API], 581
IsColumnCurrentTimestamp メソッド
ULTableSchema クラス [Ultra Light .NET API], 582
IsColumnCurrentTime メソッド
ULTableSchema クラス [Ultra Light .NET API], 581
IsColumnDescending メソッド
ULIndexSchema クラス [Ultra Light .NET API], 360
IsColumnGlobalAutoIncrement メソッド
ULTableSchema クラス [Ultra Light .NET API], 583
IsColumnNewUUID メソッド
ULTableSchema クラス [Ultra Light .NET API], 583
IsColumnNullable メソッド
ULTableSchema クラス [Ultra Light .NET API], 584
IsDBNull メソッド
ULDataReader クラス [Ultra Light .NET API], 310
IsEOF プロパティ
ULDataReader クラス [Ultra Light .NET API], 283

IsFixedSize プロパティ
ULParameterCollection クラス [Ultra Light .NET API], 394
IsForeignKeyCheckOnCommit プロパティ
ULIndexSchema クラス [Ultra Light .NET API], 355
IsForeignKeyNullable プロパティ
ULIndexSchema クラス [Ultra Light .NET API], 356
IsForeignKey プロパティ
ULIndexSchema クラス [Ultra Light .NET API], 355
IsInPublication メソッド
ULTableSchema クラス [Ultra Light .NET API], 584
IsNeverSynchronized プロパティ
ULTableSchema クラス [Ultra Light .NET API], 574
IsNullable プロパティ
ULParameter クラス [Ultra Light .NET API], 384
IsolationLevel プロパティ
ULTransaction クラス [Ultra Light .NET API], 588
IsOpen プロパティ
ULCursorSchema クラス [Ultra Light .NET API], 235
ULDatabaseSchema クラス [Ultra Light .NET API], 267
ULIndexSchema クラス [Ultra Light .NET API], 356
ULPublicationSchema クラス [Ultra Light .NET API], 414
IsPrimaryKey プロパティ
ULIndexSchema クラス [Ultra Light .NET API], 357
IsReadOnly プロパティ
ULParameterCollection クラス [Ultra Light .NET API], 395
IsSynchronized プロパティ
ULParameterCollection クラス [Ultra Light .NET API], 395
IsUniqueIndex プロパティ
ULIndexSchema クラス [Ultra Light .NET API], 357
IsUniqueKey プロパティ
ULIndexSchema クラス [Ultra Light .NET API], 358
Item(Int32) プロパティ
ULDataReader クラス [Ultra Light .NET API], 283
ULParameterCollection クラス [Ultra Light .NET API], 395
Item(String) プロパティ
ULDataReader クラス [Ultra Light .NET API], 284
ULParameterCollection クラス [Ultra Light .NET API], 396
Item プロパティ
ULBulkCopyColumnMappingCollection クラス [Ultra Light .NET API], 82
ULConnectionStringBuilder クラス [Ultra Light .NET API], 212

K

KeepPartialDownload プロパティ
ULSyncParms クラス [Ultra Light .NET API], 518

L

LastIdentity プロパティ
ULConnection クラス [Ultra Light .NET API], 150
LookupBackward(Int16) メソッド
ULTable クラス [Ultra Light .NET API], 566
LookupBackward メソッド
ULTable クラス [Ultra Light .NET API], 566
LookupBegin メソッド
ULTable クラス [Ultra Light .NET API], 567
LookupForward(Int16) メソッド
ULTable クラス [Ultra Light .NET API], 568
LookupForward メソッド
ULTable クラス [Ultra Light .NET API], 568

M

MaxHashSize プロパティ
ULCreateParms クラス [Ultra Light .NET API], 226
Message プロパティ
ULInfoMessageEventArgs クラス [Ultra Light .NET API], 363
MetaDataCollections プロパティ
ULMetaDataCollectionNames クラス [Ultra Light .NET API], 371
MoveAfterLast メソッド

ULDataReader クラス [Ultra Light .NET API], 311
MoveBeforeFirst メソッド
ULDataReader クラス [Ultra Light .NET API], 311
moveFirst クラス
Ultra Light.NET データ取得の例, 18
MoveFirst メソッド
ULDataReader クラス [Ultra Light .NET API], 311
moveFirst メソッド (Table クラス)
Ultra Light.NET 開発, 20
MoveLast メソッド
ULDataReader クラス [Ultra Light .NET API], 312
moveNext クラス
Ultra Light.NET データ取得の例, 18
MoveNext メソッド
ULDataReader クラス [Ultra Light .NET API], 312
moveNext メソッド (Table クラス)
Ultra Light.NET 開発, 20
MovePrevious メソッド
ULDataReader クラス [Ultra Light .NET API], 313
MoveRelative メソッド
ULDataReader クラス [Ultra Light .NET API], 313

N

Name プロパティ
ULCursorSchema クラス [Ultra Light .NET API], 235
ULIndexSchema クラス [Ultra Light .NET API], 358
ULPublicationSchema クラス [Ultra Light .NET API], 414
ULResultSetSchema クラス [Ultra Light .NET API], 444
ULTableSchema クラス [Ultra Light .NET API], 575
NativeError プロパティ
ULException クラス [Ultra Light .NET API], 324
ULInfoMessageEventArgs クラス [Ultra Light .NET API], 363
NearestCentury プロパティ
ULCreateParms クラス [Ultra Light .NET API], 227
NewPassword プロパティ
ULSyncParms クラス [Ultra Light .NET API], 519
NextResult メソッド
ULDataReader クラス [Ultra Light .NET API], 314
NotifyAfter プロパティ
ULBulkCopy クラス [Ultra Light .NET API], 69

O

Obfuscate プロパティ
ULCreateParms クラス [Ultra Light .NET API], 227
Offset プロパティ
ULParameter クラス [Ultra Light .NET API], 385
Open メソッド
ULConnection クラス [Ultra Light .NET API], 177
OrderedTableScans プロパティ
ULConnectionStringBuilder クラス [Ultra Light .NET API], 213

P

PageSize プロパティ
ULCreateParms クラス [Ultra Light .NET API], 228
ParameterName プロパティ
ULParameter クラス [Ultra Light .NET API], 386
Parameters プロパティ
ULCommand クラス [Ultra Light .NET API], 103
PartialDownloadRetained プロパティ
ULSyncResult クラス [Ultra Light .NET API], 544
Password プロパティ
ULConnectionParms クラス [Ultra Light .NET API], 199
ULConnectionStringBuilder クラス [Ultra Light .NET API], 214
ULFileTransfer クラス [Ultra Light .NET API], 340
ULSyncParms クラス [Ultra Light .NET API], 520
PDF
マニュアル, viii
PingOnly プロパティ

ULSyncParms クラス [Ultra Light .NET API], 520
Plan プロパティ
ULCommand クラス [Ultra Light .NET API], 104
Precision プロパティ
ULCreateParms クラス [Ultra Light .NET API], 228
ULParameter クラス [Ultra Light .NET API], 386
Prepare メソッド
ULCommand クラス [Ultra Light .NET API], 127
PrimaryKey プロパティ
ULTableSchema クラス [Ultra Light .NET API], 575
PublicationCount プロパティ
ULDatabaseSchema クラス [Ultra Light .NET API], 268
PublicationSchema クラス
Ultra Light.NET 開発, 27
Publications プロパティ
ULMetaDataCollectionNames クラス [Ultra Light .NET API], 372
ULSyncParms クラス [Ultra Light .NET API], 521

R

Read メソッド
ULDataReader クラス [Ultra Light .NET API], 314
ReceivedBytes プロパティ
ULSyncProgressData クラス [Ultra Light .NET API], 530
ReceivedDeletes プロパティ
ULSyncProgressData クラス [Ultra Light .NET API], 531
ReceivedInserts プロパティ
ULSyncProgressData クラス [Ultra Light .NET API], 531
ReceivedUpdates プロパティ
ULSyncProgressData クラス [Ultra Light .NET API], 531
RecordsAffected プロパティ
ULDataReader クラス [Ultra Light .NET API], 285
ULRowUpdatedEventArgs クラス [Ultra Light .NET API], 452
ReferencedIndexName プロパティ
ULIndexSchema クラス [Ultra Light .NET API], 359
ReferencedTableName プロパティ
ULIndexSchema クラス [Ultra Light .NET API], 359
RegisterForEvent メソッド
ULConnection クラス [Ultra Light .NET API], 178
RemoveAt(Int32) メソッド
ULParameterCollection クラス [Ultra Light .NET API], 410
RemoveAt(String) メソッド
ULParameterCollection クラス [Ultra Light .NET API], 410
RemoveAt メソッド
ULBulkCopyColumnMappingCollection クラス [Ultra Light .NET API], 89
Remove メソッド
ULBulkCopyColumnMappingCollection クラス [Ultra Light .NET API], 88
ULConnectionStringBuilder クラス [Ultra Light .NET API], 219
ULParameterCollection クラス [Ultra Light .NET API], 409
ReservedWords プロパティ
ULMetaDataCollectionNames クラス [Ultra Light .NET API], 372
ReserveSize プロパティ
ULConnectionStringBuilder クラス [Ultra Light .NET API], 215
ResetDbType メソッド
ULParameter クラス [Ultra Light .NET API], 390
ResetLastDownloadTime メソッド
ULConnection クラス [Ultra Light .NET API], 179
Restrictions プロパティ
ULMetaDataCollectionNames クラス [Ultra Light .NET API], 373
ResumedAtSize プロパティ
ULFileTransferProgressData クラス [Ultra Light .NET API], 350
ResumePartialDownload プロパティ
ULFileTransfer クラス [Ultra Light .NET API], 340
ULSyncParms クラス [Ultra Light .NET API], 522
RevokeConnectFrom メソッド

ULConnection クラス [Ultra Light .NET API], 179
revokeConnectionFrom メソッド
Ultra Light.NET 開発, 29
RollbackPartialDownload メソッド
ULConnection クラス [Ultra Light .NET API], 180
Rollback メソッド
ULTransaction クラス [Ultra Light .NET API], 589
RowCount プロパティ
ULDataReader クラス [Ultra Light .NET API], 286
RowsCopied プロパティ
ULRowsCopiedEventArgs クラス [Ultra Light .NET API], 446
RowUpdated イベント
ULDataAdapter クラス [Ultra Light .NET API], 252
RowUpdating イベント
ULDataAdapter クラス [Ultra Light .NET API], 253
RuntimeType プロパティ
ULDatabaseManager クラス [Ultra Light .NET API], 256

S

samples-dir
マニュアルの使用方法, xi
Scale プロパティ
ULCreateParms クラス [Ultra Light .NET API], 229
ULParameter クラス [Ultra Light .NET API], 387
Schema プロパティ
ULConnection クラス [Ultra Light .NET API], 151
ULDataReader クラス [Ultra Light .NET API], 286
ULTable クラス [Ultra Light .NET API], 556
ScriptCount プロパティ
ULSqlProgressData クラス [Ultra Light .NET API], 480
ScriptProgressed メソッド
ULSqlPassthroughProgressListener インタフェース [Ultra Light .NET API], 477
SelectCommand プロパティ
ULDataAdapter クラス [Ultra Light .NET API], 249
SELECT 文
Ultra Light.NET の例, 18
SendColumnNames プロパティ
ULSyncParms クラス [Ultra Light .NET API], 523
SendDownloadAck プロパティ
ULSyncParms クラス [Ultra Light .NET API], 523
SendNotification メソッド
ULConnection クラス [Ultra Light .NET API], 180
SentBytes プロパティ
ULSyncProgressData クラス [Ultra Light .NET API], 532
SentDeletes プロパティ
ULSyncProgressData クラス [Ultra Light .NET API], 532
SentInserts プロパティ
ULSyncProgressData クラス [Ultra Light .NET API], 533
SentUpdates プロパティ
ULSyncProgressData クラス [Ultra Light .NET API], 533
ServerSyncInvoked メソッド
ULServerSyncListener インタフェース [Ultra Light .NET API], 459
ServerVersion プロパティ
ULConnection クラス [Ultra Light .NET API], 151
SetActiveSyncListener メソッド
ULDatabaseManager クラス [Ultra Light .NET API], 260
SetBoolean メソッド
ULResultSet クラス [Ultra Light .NET API], 425
SetBytes メソッド
ULResultSet クラス [Ultra Light .NET API], 427
SetByte メソッド
ULResultSet クラス [Ultra Light .NET API], 426
SetDatabaseOption メソッド
ULDatabaseSchema クラス [Ultra Light .NET API], 273
SetDateTime メソッド
ULResultSet クラス [Ultra Light .NET API], 429
SetDBNull メソッド
ULResultSet クラス [Ultra Light .NET API], 428

SetDecimal メソッド
ULResultSet クラス [Ultra Light .NET API], 429
SetDouble メソッド
ULResultSet クラス [Ultra Light .NET API], 430
SetFloat メソッド
ULResultSet クラス [Ultra Light .NET API], 431
SetGlobalListener メソッド
ULDatabaseManager クラス [Ultra Light .NET API], 261
SetGuid メソッド
ULResultSet クラス [Ultra Light .NET API], 432
SetInt16 メソッド
ULResultSet クラス [Ultra Light .NET API], 433
SetInt32 メソッド
ULResultSet クラス [Ultra Light .NET API], 434
SetInt64 メソッド
ULResultSet クラス [Ultra Light .NET API], 435
SetServerSyncListener メソッド
ULDatabaseManager クラス [Ultra Light .NET API], 262
SetString メソッド
ULResultSet クラス [Ultra Light .NET API], 436
SetTimeSpan メソッド
ULResultSet クラス [Ultra Light .NET API], 437
SetToDefault メソッド
ULResultSet クラス [Ultra Light .NET API], 437
SetUInt16 メソッド
ULResultSet クラス [Ultra Light .NET API], 438
SetUInt32 メソッド
ULResultSet クラス [Ultra Light .NET API], 439
SetUInt64 メソッド
ULResultSet クラス [Ultra Light .NET API], 440
SetValues メソッド
ULSyncProgressData クラス [Ultra Light .NET API], 536
SignalSyncIsComplete メソッド
ULDatabaseManager クラス [Ultra Light .NET API], 263
Size プロパティ
ULParameter クラス [Ultra Light .NET API], 387
SourceColumnNullMapping プロパティ
ULParameter クラス [Ultra Light .NET API], 388
SourceColumn プロパティ
ULBulkCopyColumnMapping クラス [Ultra Light .NET API], 79
ULParameter クラス [Ultra Light .NET API], 388
SourceOrdinal プロパティ
ULBulkCopyColumnMapping クラス [Ultra Light .NET API], 80
SourceVersion プロパティ
ULParameter クラス [Ultra Light .NET API], 389
Source プロパティ
ULException クラス [Ultra Light .NET API], 324
ULInfoMessageEventArgs クラス [Ultra Light .NET API], 363
SQL Anywhere
マニュアル, viii
SQL Anywhere Explorer
Ultra Light 制限事項, 12
Ultra Light でサポートされるプログラミング言語, 7
Ultra Litht.NET 説明, 12
SQL Anywhere Explorer の設定
Ultra Light 説明, 8
SQL 結果セットのナビゲーション
Ultra Light.NET, 18
StartLine プロパティ
ULConnectionStringBuilder クラス [Ultra Light .NET API], 215
StartSynchronizationDelete メソッド
ULConnection クラス [Ultra Light .NET API], 181
StateChange イベント
ULConnection クラス [Ultra Light .NET API], 187
State プロパティ
ULConnection クラス [Ultra Light .NET API], 152
ULSqlProgressData クラス [Ultra Light .NET API], 480
ULSyncProgressData クラス [Ultra Light .NET API], 534
StopSynchronizationDelete メソッド
ULConnection クラス [Ultra Light .NET API], 182
StreamErrorCode プロパティ
ULFileTransfer クラス [Ultra Light .NET API], 341
ULSyncResult クラス [Ultra Light .NET API], 545
StreamErrorParameters プロパティ
ULSyncResult クラス [Ultra Light .NET API], 545
StreamErrorSystem プロパティ

ULFileTransfer クラス [Ultra Light .NET API], 342
ULSyncResult クラス [Ultra Light .NET API], 546
StreamParms プロパティ
ULFileTransfer クラス [Ultra Light .NET API], 342
ULSyncParms クラス [Ultra Light .NET API], 524
Stream プロパティ
ULFileTransfer クラス [Ultra Light .NET API], 341
ULSyncParms クラス [Ultra Light .NET API], 524
Sybase Central
Visual Studio から開く, 7
SYNC_ALL_DB フィールド
ULPublicationSchema クラス [Ultra Light .NET API], 413
SYNC_ALL_PUBS フィールド
ULPublicationSchema クラス [Ultra Light .NET API], 413
Synchronize(ULSyncProgressListener) メソッド
ULConnection クラス [Ultra Light .NET API], 183
Synchronize メソッド
ULConnection クラス [Ultra Light .NET API], 182
SyncParms プロパティ
ULConnection クラス [Ultra Light .NET API], 152
SyncProgressed メソッド
ULSyncProgressListener インタフェース [Ultra Light .NET API], 538
SyncResult プロパティ
ULConnection クラス [Ultra Light .NET API], 153
SyncRoot プロパティ
ULParameterCollection クラス [Ultra Light .NET API], 397
SyncTableCount プロパティ
ULSyncProgressData クラス [Ultra Light .NET API], 534
SyncTableIndex プロパティ
ULSyncProgressData クラス [Ultra Light .NET API], 535

T

TableCount プロパティ
ULDatabaseSchema クラス [Ultra Light .NET API], 268
TableID プロパティ
ULSyncProgressData クラス [Ultra Light .NET API], 535
TableMappings プロパティ
ULDataAdapter クラス [Ultra Light .NET API], 250
TableName プロパティ
ULSyncProgressData クラス [Ultra Light .NET API], 536
TableSchema クラス
Ultra Light.NET 開発, 27
Tables プロパティ
ULMetaDataCollectionNames クラス [Ultra Light .NET API], 373
TimeFormat プロパティ
ULCreateParms クラス [Ultra Light .NET API], 229
TimestampFormat プロパティ
ULCreateParms クラス [Ultra Light .NET API], 230
TimestampIncrement プロパティ
ULCreateParms クラス [Ultra Light .NET API], 230
Timestamp プロパティ
ULSyncResult クラス [Ultra Light .NET API], 546
ToString メソッド
ULConnectionParms クラス [Ultra Light .NET API], 200
ULCreateParms クラス [Ultra Light .NET API], 231
ULInfoMessageEventArgs クラス [Ultra Light .NET API], 364
ULParameter クラス [Ultra Light .NET API], 391
Transaction プロパティ
ULCommand クラス [Ultra Light .NET API], 104
TriggerEvent メソッド
ULConnection クラス [Ultra Light .NET API], 184
Truncate メソッド
ULTable クラス [Ultra Light .NET API], 569
TryGetValue メソッド

ULConnectionStringBuilder クラス [Ultra Light .NET API], 219

U

ULActiveSyncListener インタフェース [Ultra Light .NET API]
ActiveSyncInvoked メソッド, 57
説明, 57
ULAuthStatusCode 列挙体 [Ultra Light .NET API]
説明, 61
ULBulkCopy(String, ULBulkCopyOptions) コンストラクタ
ULBulkCopy クラス [Ultra Light .NET API], 65
ULBulkCopy(String) コンストラクタ
ULBulkCopy クラス [Ultra Light .NET API], 64
ULBulkCopy(ULConnection, ULBulkCopyOptions, ULTransaction) コンストラクタ
ULBulkCopy クラス [Ultra Light .NET API], 66
ULBulkCopy(ULConnection) コンストラクタ
ULBulkCopy クラス [Ultra Light .NET API], 63
ULBulkCopyColumnMapping(Int32, Int32) コンストラクタ
ULBulkCopyColumnMapping クラス [Ultra Light .NET API], 75
ULBulkCopyColumnMapping(Int32, String) コンストラクタ
ULBulkCopyColumnMapping クラス [Ultra Light .NET API], 76
ULBulkCopyColumnMapping(String, Int32) コンストラクタ
ULBulkCopyColumnMapping クラス [Ultra Light .NET API], 77
ULBulkCopyColumnMapping(String, String) コンストラクタ
ULBulkCopyColumnMapping クラス [Ultra Light .NET API], 77
ULBulkCopyColumnMappingCollection クラス [Ultra Light .NET API]
Add(Int32, Int32) メソッド, 84
Add(Int32, String) メソッド, 84
Add(String, Int32) メソッド, 85
Add(String, String) メソッド, 86
Add(ULBulkCopyColumnMapping) メソッド, 83
Contains メソッド, 87
CopyTo メソッド, 87
IndexOf メソッド, 88
Item プロパティ, 82
Remove メソッド, 88
RemoveAt メソッド, 89
説明, 81
ULBulkCopyColumnMapping クラス [Ultra Light .NET API]
DestinationColumn プロパティ, 78
DestinationOrdinal プロパティ, 79
SourceColumn プロパティ, 79
SourceOrdinal プロパティ, 80
ULBulkCopyColumnMapping コンストラクタ, 75
ULBulkCopyColumnMapping(Int32, Int32) コンストラクタ, 75
ULBulkCopyColumnMapping(Int32, String) コンストラクタ, 76
ULBulkCopyColumnMapping(String, Int32) コンストラクタ, 77
ULBulkCopyColumnMapping(String, String) コンストラクタ, 77
説明, 74
ULBulkCopyColumnMapping コンストラクタ
ULBulkCopyColumnMapping クラス [Ultra Light .NET API], 75
ULBulkCopyOptions 列挙体 [Ultra Light .NET API]
説明, 90
ULBulkCopy クラス [Ultra Light .NET API]
BatchSize プロパティ, 66
BulkCopyTimeout プロパティ, 67
Close メソッド, 69
ColumnMappings プロパティ, 68
DestinationTableName プロパティ, 68
Dispose メソッド, 70
NotifyAfter プロパティ, 69
ULBulkCopy(String) コンストラクタ, 64
ULBulkCopy(String, ULBulkCopyOptions) コンストラクタ, 65
ULBulkCopy(ULConnection) コンストラクタ, 63
ULBulkCopy(ULConnection, ULBulkCopyOptions, ULTransaction) コンストラクタ, 66
ULRowsCopied イベント, 73
WriteToServer(DataRow[]) メソッド, 70
WriteToServer(DataTable) メソッド, 71
WriteToServer(DataTable, DataRowState) メソッド, 72
WriteToServer(IDataReader) メソッド, 71

説明, 62
ULCommand(String, ULConnection, ULTransaction) コンストラクタ
ULCommand クラス [Ultra Light .NET API], 98
ULCommand(String, ULConnection) コンストラクタ
ULCommand クラス [Ultra Light .NET API], 97
ULCommand(String) コンストラクタ
ULCommand クラス [Ultra Light .NET API], 96
ULCommandBuilder(ULDataAdapter) コンストラクタ
ULCommandBuilder クラス [Ultra Light .NET API], 132
ULCommandBuilder クラス [Ultra Light .NET API]
DataAdapter プロパティ, 133
GetDeleteCommand メソッド, 134
GetDeleteCommand(Boolean) メソッド, 133
GetInsertCommand メソッド, 136
GetInsertCommand(Boolean) メソッド, 135
GetUpdateCommand メソッド, 138
GetUpdateCommand(Boolean) メソッド, 137
ULCommandBuilder コンストラクタ, 132
ULCommandBuilder(ULDataAdapter) コンストラクタ, 132
説明, 129
ULCommandBuilder コンストラクタ
ULCommandBuilder クラス [Ultra Light .NET API], 132
ULCommand クラス
Ultra Light.NET データ取得の例, 18
Ultra Light.NET データ操作の例, 16
ULCommand クラス [Ultra Light .NET API]
BeginExecuteNonQuery メソッド, 105
BeginExecuteNonQuery(AsyncCallback, Object) メソッド, 106
BeginExecuteReader メソッド, 107
BeginExecuteReader(AsyncCallback, Object) メソッド, 108
BeginExecuteReader(AsyncCallback, Object, CommandBehavior) メソッド, 109
BeginExecuteReader(CommandBehavior) メソッド, 107
Cancel メソッド, 110
CommandText プロパティ, 99
CommandTimeout プロパティ, 100
CommandType プロパティ, 100
Connection プロパティ, 101
CreateParameter メソッド, 111
DesignTimeVisible プロパティ, 102
EndExecuteNonQuery メソッド, 111
EndExecuteReader メソッド, 115
ExecuteNonQuery メソッド, 118
ExecuteReader メソッド, 119
ExecuteReader(CommandBehavior) メソッド, 120
ExecuteResultSet メソッド, 122
ExecuteResultSet(CommandBehavior) メソッド, 123
ExecuteScalar メソッド, 124
ExecuteTable メソッド, 125
ExecuteTable(CommandBehavior) メソッド, 126
IndexName プロパティ, 102
Parameters プロパティ, 103
Plan プロパティ, 104
Prepare メソッド, 127
Transaction プロパティ, 104
ULCommand コンストラクタ, 95
ULCommand(String) コンストラクタ, 96
ULCommand(String, ULConnection) コンストラクタ, 97
ULCommand(String, ULConnection, ULTransaction) コンストラクタ, 98
UpdatedRowSource プロパティ, 105
説明, 91
ULCommand コンストラクタ
ULCommand クラス [Ultra Light .NET API], 95
ULConnection(String) コンストラクタ
ULConnection クラス [Ultra Light .NET API], 144
ULConnectionParms.UnusedEventHandler デリゲート [Ultra Light .NET API]
説明, 202
ULConnectionParms クラス [Ultra Light .NET API]
AdditionalParms プロパティ, 192
CacheSize プロパティ, 196
ConnectionName プロパティ, 197
DatabaseOnCE プロパティ, 197
DatabaseOnDesktop プロパティ, 197
EncryptionKey プロパティ, 198
Password プロパティ, 199
ToString メソッド, 200
ULConnectionParms コンストラクタ, 192
UnusedEvent イベント, 200
UserID プロパティ, 199
説明, 189

ULConnectionParms コンストラクタ
ULConnectionParms クラス [Ultra Light .NET API], 192
ULConnectionStringBuilder(String) コンストラクタ
ULConnectionStringBuilder クラス [Ultra Light .NET API], 208
ULConnectionStringBuilder クラス [Ultra Light .NET API]
CacheSize プロパティ, 208
ConnectionName プロパティ, 209
ContainsKey メソッド, 217
DatabaseKey プロパティ, 209
DatabaseName プロパティ, 210
DatabaseOnCE プロパティ, 211
DatabaseOnDesktop プロパティ, 211
EquivalentTo メソッド, 217
GetShortName メソッド, 218
Item プロパティ, 212
OrderedTableScans プロパティ, 213
Password プロパティ, 214
Remove メソッド, 219
ReserveSize プロパティ, 215
StartLine プロパティ, 215
TryGetValue メソッド, 219
ULConnectionStringBuilder コンストラクタ, 207
ULConnectionStringBuilder(String) コンストラクタ, 208
UserID プロパティ, 216
説明, 203
ULConnectionStringBuilder コンストラクタ
ULConnectionStringBuilder クラス [Ultra Light .NET API], 207
ULConnection クラス [Ultra Light .NET API]
BeginTransaction メソッド, 153
BeginTransaction(IsolationLevel) メソッド, 154
CancelGetNotification メソッド, 155
ChangeDatabase メソッド, 156
ChangeEncryptionKey メソッド, 157
ChangePassword メソッド, 157
Close メソッド, 158
ConnectionString プロパティ, 146
ConnectionTimeout プロパティ, 147
CountUploadRows(String, Int64) メソッド, 159
CountUploadRows(String, UInt32) メソッド, 159
CreateCommand メソッド, 160
CreateNotificationQueue メソッド, 161
Database プロパティ, 148
DatabaseID プロパティ, 148
DatabaseManager プロパティ, 149
DataSource プロパティ, 147
DeclareEvent メソッド, 162
DestroyNotificationQueue メソッド, 163
ExecuteNextSQLPassthroughScript メソッド, 164
ExecuteSQLPassthroughScripts メソッド, 164
ExecuteTable(String) メソッド, 165
ExecuteTable(String, String) メソッド, 166
ExecuteTable(String, String, CommandBehavior) メソッド, 168
GetLastDownloadTime メソッド, 169
GetNewUUID メソッド, 170
GetNotification メソッド, 171
GetNotificationParameter メソッド, 172
GetSchema メソッド, 173
GetSchema(String) メソッド, 174
GetSchema(String, String[]) メソッド, 174
GetSQLPassthroughScriptCount メソッド, 173
GlobalAutoIncrementUsage プロパティ, 149
GrantConnectTo メソッド, 176
InfoMessage イベント, 185
INVALID_DATABASE_ID フィールド, 145
LastIdentity プロパティ, 150
Open メソッド, 177
RegisterForEvent メソッド, 178
ResetLastDownloadTime メソッド, 179
RevokeConnectFrom メソッド, 179
RollbackPartialDownload メソッド, 180
Schema プロパティ, 151
SendNotification メソッド, 180
ServerVersion プロパティ, 151
StartSynchronizationDelete メソッド, 181
State プロパティ, 152
StateChange イベント, 187
StopSynchronizationDelete メソッド, 182
Synchronize メソッド, 182
Synchronize(ULSyncProgressListener) メソッド, 183
SyncParms プロパティ, 152
SyncResult プロパティ, 153
TriggerEvent メソッド, 184
ULConnection コンストラクタ, 143
ULConnection(String) コンストラクタ, 144
ValidateDatabase メソッド, 184
説明, 139
ULConnection コンストラクタ

ULConnection クラス [Ultra Light .NET API], 143
ULCreateParms クラス [Ultra Light .NET API]
CaseSensitive プロパティ, 224
ChecksumLevel プロパティ, 224
DateFormat プロパティ, 225
DateOrder プロパティ, 225
FIPS プロパティ, 226
MaxHashSize プロパティ, 226
NearestCentury プロパティ, 227
Obfuscate プロパティ, 227
PageSize プロパティ, 228
Precision プロパティ, 228
Scale プロパティ, 229
TimeFormat プロパティ, 229
TimestampFormat プロパティ, 230
TimestampIncrement プロパティ, 230
ToString メソッド, 231
ULCreateParms コンストラクタ, 224
UTF8Encoding プロパティ, 231
説明, 221
ULCreateParms コンストラクタ
ULCreateParms クラス [Ultra Light .NET API], 224
ULCursorSchema クラス [Ultra Light .NET API]
ColumnCount プロパティ, 234
GetColumnID メソッド, 235
GetColumnName メソッド, 236
GetColumnPrecision メソッド, 237
GetColumnScale メソッド, 239
GetColumnSize メソッド, 239
GetColumnSQLName メソッド, 238
GetColumnULDbType メソッド, 240
GetSchemaTable メソッド, 241
IsOpen プロパティ, 235
Name プロパティ, 235
説明, 233
ULDataAdapter(String, String) コンストラクタ
ULDataAdapter クラス [Ultra Light .NET API], 247
ULDataAdapter(String, ULConnection) コンストラクタ
ULDataAdapter クラス [Ultra Light .NET API], 246
ULDataAdapter(ULCommand) コンストラクタ
ULDataAdapter クラス [Ultra Light .NET API], 246
ULDataAdapter クラス [Ultra Light .NET API]
DeleteCommand プロパティ, 248
GetFillParameters メソッド, 252
InsertCommand プロパティ, 249
RowUpdated イベント, 252
RowUpdating イベント, 253
SelectCommand プロパティ, 249
TableMappings プロパティ, 250
ULDataAdapter コンストラクタ, 245
ULDataAdapter(String, String) コンストラクタ, 247
ULDataAdapter(String, ULConnection) コンストラクタ, 246
ULDataAdapter(ULCommand) コンストラクタ, 246
UpdateCommand プロパティ, 251
説明, 242
ULDataAdapter コンストラクタ
ULDataAdapter クラス [Ultra Light .NET API], 245
ULDatabaseManager クラス [Ultra Light .NET API]
CreateDatabase メソッド, 257
DropDatabase メソッド, 259
RuntimeType プロパティ, 256
SetActiveSyncListener メソッド, 260
SetGlobalListener メソッド, 261
SetServerSyncListener メソッド, 262
SignalSyncIsComplete メソッド, 263
ValidateDatabase メソッド, 263
説明, 255
ULDatabaseSchema クラス
iAnywhere.Data.UltraLite.NET ネームスペース, 27
ULDatabaseSchema クラス [Ultra Light .NET API]
CollationName プロパティ, 266
GetDatabaseProperty メソッド, 269
GetPublicationName メソッド, 271
GetPublicationSchema メソッド, 272
GetTableName メソッド, 273
IsCaseSensitive プロパティ, 267
IsOpen プロパティ, 267
PublicationCount プロパティ, 268
SetDatabaseOption メソッド, 273
TableCount プロパティ, 268
説明, 265
ULDataReader クラス
Ultra Light.NET データ取得の例, 18

ULDataReader クラス [Ultra Light .NET API]
Close メソッド, 287
Depth プロパティ, 281
FieldCount プロパティ, 281
GetBoolean メソッド, 287
GetByte メソッド, 288
GetBytes(Int32) メソッド, 290
GetBytes(Int32, Int64, Byte[], Int32, Int32) メソッド, 289
GetChar メソッド, 291
GetChars メソッド, 292
GetDataTypeName メソッド, 293
GetDateTime メソッド, 294
GetDecimal メソッド, 294
GetDouble メソッド, 295
GetEnumerator メソッド, 296
GetFieldType メソッド, 296
GetFloat メソッド, 297
GetGuid メソッド, 298
GetInt16 メソッド, 298
GetInt32 メソッド, 299
GetInt64 メソッド, 300
GetName メソッド, 300
GetOrdinal メソッド, 301
GetRowCount メソッド, 302
GetSchemaTable メソッド, 303
GetString メソッド, 305
GetTimeSpan メソッド, 306
GetUInt16 メソッド, 306
GetUInt32 メソッド, 307
GetUInt64 メソッド, 308
GetValue メソッド, 308
GetValues メソッド, 309
HasRows プロパティ, 282
IsBOF プロパティ, 282
IsClosed プロパティ, 283
IsDBNull メソッド, 310
IsEOF プロパティ, 283
Item(Int32) プロパティ, 283
Item(String) プロパティ, 284
MoveAfterLast メソッド, 311
MoveBeforeFirst メソッド, 311
MoveFirst メソッド, 311
MoveLast メソッド, 312
MoveNext メソッド, 312
MovePrevious メソッド, 313
MoveRelative メソッド, 313
NextResult メソッド, 314
Read メソッド, 314
RecordsAffected プロパティ, 285
RowCount プロパティ, 286
Schema プロパティ, 286
説明, 276
ULDateOrder 列挙体 [Ultra Light .NET API]
説明, 316
ULDbType プロパティ
ULParameter クラス [Ultra Light .NET API], 389
ULDbType 列挙体 [Ultra Light .NET API]
説明, 317
ULDBValid 列挙体 [Ultra Light .NET API]
説明, 321
ULException クラス [Ultra Light .NET API]
GetObjectData メソッド, 325
NativeError プロパティ, 324
Source プロパティ, 324
ULException コンストラクタ, 323
説明, 322
ULException コンストラクタ
ULException クラス [Ultra Light .NET API], 323
ULFactory クラス [Ultra Light .NET API]
CanCreateDataSourceEnumerator メソッド, 328
CreateCommand メソッド, 329
CreateCommandBuilder メソッド, 329
CreateConnection メソッド, 330
CreateConnectionStringBuilder メソッド, 330
CreateDataAdapter メソッド, 330
CreateParameter メソッド, 331
Instance フィールド, 328
説明, 326
ULFileTransferProgressData クラス [Ultra Light .NET API]
BytesReceived プロパティ, 349
FileSize プロパティ, 349
FLAG_IS_BLOCKING フィールド, 349
Flags プロパティ, 350
ResumedAtSize プロパティ, 350
説明, 348
ULFileTransferProgressListener インタフェース [Ultra Light .NET API]
FileTransferProgressed メソッド, 351
説明, 351
ULFileTransfer クラス [Ultra Light .NET API]
AuthenticationParms プロパティ, 336
AuthStatus プロパティ, 335

AuthValue プロパティ, 335
DestinationFileName プロパティ, 336
DestinationPath プロパティ, 337
DownloadedFile プロパティ, 337
DownloadFile メソッド, 344
DownloadFile(ULFileTransferProgressListener) メソッド, 345
FileAuthCode プロパティ, 338
FileName プロパティ, 338
ForceDownload プロパティ, 339
Password プロパティ, 340
ResumePartialDownload プロパティ, 340
Stream プロパティ, 341
StreamErrorCode プロパティ, 341
StreamErrorSystem プロパティ, 342
StreamParms プロパティ, 342
ULFileTransfer コンストラクタ, 334
UserName プロパティ, 343
Version プロパティ, 344
説明, 332
ULFileTransfer コンストラクタ
ULFileTransfer クラス [Ultra Light .NET API], 334
ULIndexSchema クラス [Ultra Light .NET API]
ColumnCount プロパティ, 354
GetColumnName メソッド, 359
IsColumnDescending メソッド, 360
IsForeignKey プロパティ, 355
IsForeignKeyCheckOnCommit プロパティ, 355
IsForeignKeyNullable プロパティ, 356
IsOpen プロパティ, 356
IsPrimaryKey プロパティ, 357
IsUniqueIndex プロパティ, 357
IsUniqueKey プロパティ, 358
Name プロパティ, 358
ReferencedIndexName プロパティ, 359
ReferencedTableName プロパティ, 359
説明, 353
ULInfoMessageEventArgs クラス [Ultra Light .NET API]
Message プロパティ, 363
NativeError プロパティ, 363
Source プロパティ, 363
ToString メソッド, 364
説明, 362
ULInfoMessageEventHandler デリゲート [Ultra Light .NET API]
説明, 365
ULMetaDataCollectionNames クラス [Ultra Light .NET API]
Columns プロパティ, 367
DataSourceInformation プロパティ, 368
DataTypes プロパティ, 368
ForeignKeys プロパティ, 369
IndexColumns プロパティ, 370
Indexes プロパティ, 370
MetaDataCollections プロパティ, 371
Publications プロパティ, 372
ReservedWords プロパティ, 372
Restrictions プロパティ, 373
Tables プロパティ, 373
説明, 366
ULParameter(String, Object) コンストラクタ
ULParameter クラス [Ultra Light .NET API], 378
ULParameter(String, ULDbType, Int32, ParameterDirection, Boolean, Byte, Byte, String, DataRowVersion, Object) コンストラクタ
ULParameter クラス [Ultra Light .NET API], 382
ULParameter(String, ULDbType, Int32, String) コンストラクタ
ULParameter クラス [Ultra Light .NET API], 381
ULParameter(String, ULDbType, Int32) コンストラクタ
ULParameter クラス [Ultra Light .NET API], 380
ULParameter(String, ULDbType) コンストラクタ
ULParameter クラス [Ultra Light .NET API], 379
ULParameterCollection クラス [Ultra Light .NET API]
Add(Object) メソッド, 397
Add(String, Object) メソッド, 399
Add(String, ULDbType) メソッド, 400
Add(String, ULDbType, Int32) メソッド, 401
Add(String, ULDbType, Int32, String) メソッド, 402
Add(ULParameter) メソッド, 398
AddRange(Array) メソッド, 404
AddRange(ULParameter[]) メソッド, 404
Clear メソッド, 405
Contains(Object) メソッド, 405
Contains(String) メソッド, 406
CopyTo メソッド, 406
Count プロパティ, 394
GetEnumerator メソッド, 407
IndexOf(Object) メソッド, 407
IndexOf(String) メソッド, 408

Insert メソッド, 409
IsFixedSize プロパティ, 394
IsReadOnly プロパティ, 395
IsSynchronized プロパティ, 395
Item(Int32) プロパティ, 395
Item(String) プロパティ, 396
Remove メソッド, 409
RemoveAt(Int32) メソッド, 410
RemoveAt(String) メソッド, 410
SyncRoot メソッド, 397
説明, 392
ULParameter クラス [Ultra Light .NET API]
DbType プロパティ, 383
Direction プロパティ, 384
IsNullable プロパティ, 384
Offset プロパティ, 385
ParameterName プロパティ, 386
Precision プロパティ, 386
ResetDbType メソッド, 390
Scale プロパティ, 387
Size プロパティ, 387
SourceColumn プロパティ, 388
SourceColumnNullMapping プロパティ, 388
SourceVersion プロパティ, 389
ToString メソッド, 391
ULDbType プロパティ, 389
ULParameter コンストラクタ, 377
ULParameter(String, Object) コンストラクタ, 378
ULParameter(String, ULDbType) コンストラクタ, 379
ULParameter(String, ULDbType, Int32) コンストラクタ, 380
ULParameter(String, ULDbType, Int32, ParameterDirection, Boolean, Byte, Byte, String, DataRowVersion, Object) コンストラクタ, 382
ULParameter(String, ULDbType, Int32, String) コンストラクタ, 381
Value プロパティ, 390
説明, 375
ULParameter コンストラクタ
ULParameter クラス [Ultra Light .NET API], 377
ULPublicationSchema クラス [Ultra Light .NET API]
IsOpen プロパティ, 414
Name プロパティ, 414
SYNC_ALL_DB フィールド, 413
SYNC_ALL_PUBS フィールド, 413
説明, 412
ULResultSetSchema クラス [Ultra Light .NET API]
Name プロパティ, 444
説明, 442
ULResultSet クラス [Ultra Light .NET API]
AppendBytes メソッド, 422
AppendChars メソッド, 423
Delete メソッド, 425
SetBoolean メソッド, 425
SetByte メソッド, 426
SetBytes メソッド, 427
SetDateTime メソッド, 429
SetDBNull メソッド, 428
SetDecimal メソッド, 429
SetDouble メソッド, 430
SetFloat メソッド, 431
SetGuid メソッド, 432
SetInt16 メソッド, 433
SetInt32 メソッド, 434
SetInt64 メソッド, 435
SetString メソッド, 436
SetTimeSpan メソッド, 437
SetToDefault メソッド, 437
SetUInt16 メソッド, 438
SetUInt32 メソッド, 439
SetUInt64 メソッド, 440
Update メソッド, 441
説明, 415
ULRowsCopiedEventArgs クラス [Ultra Light .NET API]
Abort プロパティ, 446
RowsCopied プロパティ, 446
ULRowsCopiedEventArgs コンストラクタ, 445
説明, 445
ULRowsCopiedEventArgs コンストラクタ
ULRowsCopiedEventArgs クラス [Ultra Light .NET API], 445
ULRowsCopiedEventHandler デリゲート [Ultra Light .NET API]
説明, 448
ULRowsCopied イベント
ULBulkCopy クラス [Ultra Light .NET API], 73
ULRowUpdatedEventArgs クラス [Ultra Light .NET API]
Command プロパティ, 451
RecordsAffected プロパティ, 452
ULRowUpdatedEventArgs コンストラクタ, 450

説明, 449
ULRowUpdatedEventArgs コンストラクタ
ULRowUpdatedEventArgs クラス [Ultra Light .NET API], 450
ULRowUpdatedEventHandler デリゲート [Ultra Light .NET API]
説明, 453
ULRowUpdatingEventArgs クラス [Ultra Light .NET API]
Command プロパティ, 456
ULRowUpdatingEventArgs コンストラクタ, 455
説明, 454
ULRowUpdatingEventArgs コンストラクタ
ULRowUpdatingEventArgs クラス [Ultra Light .NET API], 455
ULRowUpdatingEventHandler デリゲート [Ultra Light .NET API]
説明, 457
ULRuntimeType 列挙体 [Ultra Light .NET API]
説明, 458
ULServerSyncListener インタフェース [Ultra Light .NET API]
ServerSyncInvoked メソッド, 459
説明, 459
ULServerSyncListener 列挙体 [Ultra Light .NET API]
説明, 462
ULSqlPassthroughProgressListener インタフェース [Ultra Light .NET API]
ScriptProgressed メソッド, 477
説明, 477
ULSqlProgressData クラス [Ultra Light .NET API]
CurrentScript プロパティ, 479
ScriptCount プロパティ, 480
State プロパティ, 480
説明, 479
ULSqlProgressState 列挙体 [Ultra Light .NET API]
説明, 481
ULStreamErrorCode 列挙体 [Ultra Light .NET API]
説明, 482
ULStreamType 列挙体 [Ultra Light .NET API]
説明, 512
ULSyncParms クラス [Ultra Light .NET API]
AdditionalParms プロパティ, 516
AuthenticationParms プロパティ, 517
CopyFrom メソッド, 527
DownloadOnly プロパティ, 517
KeepPartialDownload プロパティ, 518
NewPassword プロパティ, 519
Password プロパティ, 520
PingOnly プロパティ, 520
Publications プロパティ, 521
ResumePartialDownload プロパティ, 522
SendColumnNames プロパティ, 523
SendDownloadAck プロパティ, 523
Stream プロパティ, 524
StreamParms プロパティ, 524
UploadOnly プロパティ, 525
UserName プロパティ, 526
Version プロパティ, 526
説明, 514
ULSyncProgressData クラス [Ultra Light .NET API]
FLAG_IS_BLOCKING フィールド, 529
Flags プロパティ, 530
ReceivedBytes プロパティ, 530
ReceivedDeletes プロパティ, 531
ReceivedInserts プロパティ, 531
ReceivedUpdates プロパティ, 531
SentBytes プロパティ, 532
SentDeletes プロパティ, 532
SentInserts プロパティ, 533
SentUpdates プロパティ, 533
SetValues メソッド, 536
State プロパティ, 534
SyncTableCount プロパティ, 534
SyncTableIndex プロパティ, 535
TableID プロパティ, 535
TableName プロパティ, 536
説明, 528
ULSyncProgressListener インタフェース [Ultra Light .NET API]
SyncProgressed メソッド, 538
説明, 538
ULSyncProgressState 列挙体 [Ultra Light .NET API]
説明, 540
ULSyncResult クラス [Ultra Light .NET API]
AuthStatus プロパティ, 543
AuthValue プロパティ, 543
IgnoredRows プロパティ, 544
PartialDownloadRetained プロパティ, 544
StreamErrorCode プロパティ, 545
StreamErrorParameters プロパティ, 545
StreamErrorSystem プロパティ, 546
Timestamp プロパティ, 546
UploadOK プロパティ, 547

説明, 542
ULTableSchema クラス [Ultra Light .NET API]
GetColumnDefaultValue メソッド, 576
GetColumnPartitionSize メソッド, 577
GetIndex メソッド, 577
GetIndexName メソッド, 578
GetOptimalIndex メソッド, 579
GetPublicationPredicate メソッド, 579
IndexCount プロパティ, 574
IsColumnAutoIncrement メソッド, 580
IsColumnCurrentDate メソッド, 581
IsColumnCurrentTime メソッド, 581
IsColumnCurrentTimestamp メソッド, 582
IsColumnGlobalAutoIncrement メソッド, 583
IsColumnNewUUID メソッド, 583
IsColumnNullable メソッド, 584
IsInPublication メソッド, 584
IsNeverSynchronized プロパティ, 574
Name プロパティ, 575
PrimaryKey プロパティ, 575
UploadUnchangedRows プロパティ, 576
説明, 571
ULTable クラス [Ultra Light .NET API]
DeleteAllRows メソッド, 557
FindBegin メソッド, 557
FindFirst メソッド, 558
FindFirst(Int16) メソッド, 558
FindLast メソッド, 559
FindLast(Int16) メソッド, 560
FindNext メソッド, 561
FindNext(Int16) メソッド, 562
FindPrevious メソッド, 563
FindPrevious(Int16) メソッド, 564
Insert メソッド, 564
InsertBegin メソッド, 565
LookupBackward メソッド, 566
LookupBackward(Int16) メソッド, 566
LookupBegin メソッド, 567
LookupForward メソッド, 568
LookupForward(Int16) メソッド, 568
Schema プロパティ, 556
Truncate メソッド, 569
UpdateBegin メソッド, 570
説明, 548
Ultra Light
Visual Studio との統合, 34
Ultra Light.NET
ActiveSync 同期, 31
SQL を使用したデータ操作, 16
Ultra Light 用 SQL Anywhere Explorer, 5
アプリケーションでの同期, 30
暗号化, 15
アーキテクチャ, 4
エラー処理, 28
開発, 11
サポートされるプラットフォーム, 3
スキーマ情報のアクセス, 27
せつめい, 1
チュートリアル, 33
テーブル API の概要, 20
データ操作, 16
データの取得, 18
トランザクション処理, 26
同期の例, 30
動的 SQL チュートリアル, 43
配置, 48
ユーザ認証, 29
利点, 2
Ultra Light .NET API
ULActiveSyncListener インタフェース, 57
ULAuthStatusCode 列挙体, 61
ULBulkCopy クラス, 62
ULBulkCopyColumnMapping クラス, 74
ULBulkCopyColumnMappingCollection クラス, 81
ULBulkCopyOptions 列挙体, 90
ULCommand クラス, 91
ULCommandBuilder クラス, 129
ULConnection クラス, 139
ULConnectionParms クラス, 189
ULConnectionParms.UnusedEventHandler デリゲート, 202
ULConnectionStringBuilder クラス, 203
ULCreateParms クラス, 221
ULCursorSchema クラス, 233
ULDataAdapter クラス, 242
ULDatabaseManager クラス, 255
ULDatabaseSchema クラス, 265
ULDataReader クラス, 276
ULDateOrder 列挙体, 316
ULDbType 列挙体, 317
ULDBValid 列挙体, 321
ULException クラス, 322
ULFactory クラス, 326

ULFileTransfer クラス, 332
ULFileTransferProgressData クラス, 348
ULFileTransferProgressListener インタフェース, 351
ULIndexSchema クラス, 353
ULInfoMessageEventArgs クラス, 362
ULInfoMessageEventHandler デリゲート, 365
ULMetaDataCollectionNames クラス, 366
ULParameter クラス, 375
ULParameterCollection クラス, 392
ULPublicationSchema クラス, 412
ULResultSet クラス, 415
ULResultSetSchema クラス, 442
ULRowsCopiedEventArgs クラス, 445
ULRowsCopiedEventHandler デリゲート, 448
ULRowUpdatedEventArgs クラス, 449
ULRowUpdatedEventHandler デリゲート, 453
ULRowUpdatingEventArgs クラス, 454
ULRowUpdatingEventHandler デリゲート, 457
ULRuntimeType 列挙体, 458
ULServerSyncListener インタフェース, 459
ULSQLCode 列挙体, 462
ULSqlPassthroughProgressListener インタフェース, 477
ULSqlProgressData クラス, 479
ULSqlProgressState 列挙体, 481
ULStreamErrorCode 列挙体, 482
ULStreamType 列挙体, 512
ULSyncParms クラス, 514
ULSyncProgressData クラス, 528
ULSyncProgressListener インタフェース, 538
ULSyncProgressState 列挙体, 540
ULSyncResult クラス, 542
ULTable クラス, 548
ULTableSchema クラス, 571
ULTransaction クラス, 586
Ultra Light.NET 開発の概要
説明, 11
Ultra Light.NET チュートリアル
C# チュートリアルのコード・リスト, 50
Visual Basic チュートリアルのコード・リスト, 53
Ultra Light データベース
Ultra Light.NET 情報へのアクセス, 27
Ultra Light.NET 接続, 13
Ultra Light モード
Ultra Light.NET, 21
Ultra Light 用 SQL Anywhere Explorer
Visual Studio との統合, 7
設定, 8
接続, 7
説明, 5
テーブルの操作, 10
データベース・オブジェクトの追加, 9
ULTransaction クラス [Ultra Light .NET API]
Commit メソッド, 588
Connection プロパティ, 587
IsolationLevel プロパティ, 588
Rollback メソッド, 589
説明, 586
UnusedEvent イベント
ULConnectionParms クラス [Ultra Light .NET API], 200
UpdateBegin メソッド
ULTable クラス [Ultra Light .NET API], 570
UpdateCommand プロパティ
ULDataAdapter クラス [Ultra Light .NET API], 251
UpdatedRowSource プロパティ
ULCommand クラス [Ultra Light .NET API], 105
Update メソッド
ULResultSet クラス [Ultra Light .NET API], 441
UploadOK プロパティ
ULSyncResult クラス [Ultra Light .NET API], 547
UploadOnly プロパティ
ULSyncParms クラス [Ultra Light .NET API], 525
UploadUnchangedRows プロパティ
ULTableSchema クラス [Ultra Light .NET API], 576
UserID プロパティ
ULConnectionParms クラス [Ultra Light .NET API], 199
ULConnectionStringBuilder クラス [Ultra Light .NET API], 216
UserName プロパティ
ULFileTransfer クラス [Ultra Light .NET API], 343
ULSyncParms クラス [Ultra Light .NET API], 526
UTF8Encoding プロパティ
ULCreateParms クラス [Ultra Light .NET API], 231

V

ValidateDatabase メソッド
  ULConnection クラス [Ultra Light .NET API], 184
  ULDatabaseManager クラス [Ultra Light .NET API], 263
Value プロパティ
  ULParameter クラス [Ultra Light .NET API], 390
Version プロパティ
  ULFileTransfer クラス [Ultra Light .NET API], 344
  ULSyncParms クラス [Ultra Light .NET API], 526
Visual Studio
  SQL Anywhere Explorer と Ultra Light との統合, 7
  Ultra Light データベース接続, 7
  Ultra Light データベースへのアクセス, 7
  Ultra Light との統合, 34
Visual Studio .NET
  Ultra Light.NET Ultra Light データベースへの接続, 12

W

Windows Mobile
  Ultra Light.NET ターゲット・プラットフォーム, 3
WriteToServer(DataRow[]) メソッド
  ULBulkCopy クラス [Ultra Light .NET API], 70
WriteToServer(DataTable, DataRowState) メソッド
  ULBulkCopy クラス [Ultra Light .NET API], 72
WriteToServer(DataTable) メソッド
  ULBulkCopy クラス [Ultra Light .NET API], 71
WriteToServer(IDataReader) メソッド
  ULBulkCopy クラス [Ultra Light .NET API], 71

あ

アイコン
  ヘルプでの使用, xiii
値
  Ultra Light.NET アクセス, 21
暗号化
  Ultra Light.NET 開発, 15
アーキテクチャ
  Ultra Light.NET, 4

い

インストール
  Ultra Light 用 SQL Anywhere Explorer, 7
インデックス
  Ultra Light.NET スキーマ情報, 27

え

エラー
  Ultra Light.NET 処理, 28
エラー処理
  Ultra Light.NET, 28

お

[オプション] ウィンドウ
  Ultra Light 用 SQL Anywhere Explorer, 8
オフセット
  Ultra Light.NET での相対オフセット, 20
オンライン・マニュアル
  PDF, viii

か

開発
  Ultra Light.NET, 11
開発プラットフォーム
  Ultra Light.NET, 3
カラム
  Ultra Light.NET 値の変更, 22
環境変数
  コマンド・シェル, xii
  コマンド・プロンプト, xii
管理
  Ultra Light.NET トランザクション, 26

き

キャスト
  Ultra Light.NET データ型のキャスト, 22

け

結果セット
  Ultra Light.NET ナビゲーション, 18
結果セット・スキーム
  Ultra Light.NET, 19
検索
  Ultra Light.NET を使用して Ultra Light ローを検索, 22
検索メソッド

Ultra Light.NET, 22
検索モード
Ultra Light.NET, 21

こ

更新
Ultra Light.NET テーブル・ロー, 23
更新モード
Ultra Light.NET, 21
コピー
Visual Studio での Ultra Light データベース・オブジェクト, 9
コマンド・シェル
引用符, xii
カッコ, xii
環境変数, xii
中カッコ, xii
表記規則, xii
コマンド・プロンプト
引用符, xii
カッコ, xii
環境変数, xii
中カッコ, xii
表記規則, xii
コミット
Ultra Light.NET トランザクション, 26
コミット・メソッド
Ultra Light.NET トランザクション, 26
コード・リスト
Ultra Light.NET C# チュートリアル, 50
Ultra Light.NET Visual Basic チュートリアル, 53

さ

削除
Ultra Light.NET テーブル・ロー, 25
サポート
ニュースグループ, xiv
サポートされるプラットフォーム
Ultra Light.NET, 3

し

詳細情報の検索／テクニカル・サポートの依頼
テクニカル・サポート, xiv

す

スキーマ
Ultra Light.NET アクセス, 27
スキーマ情報のアクセス
Ultra Light.NET 説明, 27
スクロール
Ultra Light.NET テーブル API, 20
スレッド
Ultra Light.NET マルチスレッド・アプリケーション, 13

せ

接続
Ultra Light 用 SQL Anywhere Explorer, 7
Ultra Light.NET チュートリアル, 41
Ultra Light.NET データベース, 13
選択
Ultra Light.NET ロー, 18

そ

相対オフセット
Ultra Light.NET テーブル API, 20
挿入
Ultra Light.NET テーブル・ロー, 24
挿入モード
Ultra Light.NET, 21

た

ターゲット・プラットフォーム
Ultra Light.NET, 3

ち

チュートリアル
Ultra Light.NET の C#, 33
Ultra Light.NET の Visual Basic, 33

て

テクニカル・サポート
ニュースグループ, xiv
デベロッパー・コミュニティ
ニュースグループ, xiv
データ・アクセス
Ultra Light.NET テーブル API, 20
データ型
Ultra Light.NET API アクセス, 21
Ultra Light.NET 変換, 22
データ操作
SQL を使用した Ultra Light.NET の操作, 16

Ultra Light.NET テーブル API, 20
データベース・スキーマ
Ultra Light.NET アクセス, 27
データベース・テーブルからのデータ選択
Ultra Light.NET, 18
テーブル
Ultra Light.NET スキーマ情報, 27
テーブル API
Ultra Light.NET の概要, 20

と

同期
Ultra Light.NET, 30
Ultra Light.NET C# の例, 30
Ultra Light.NET の ActiveSync, 31
動的 SQL
Ultra Light.NET チュートリアル, 43
トピック
グラフィック・アイコン, xiii
トラブルシューティング
Ultra Light.NET 開発チェックリスト, 48
Ultra Light.NET でのエラー処理, 28
ニュースグループ, xiv
トランザクション
Ultra Light.NET 管理, 26
トランザクション処理
Ultra Light.NET 管理, 26

な

ナビゲーション
Ultra Light.NET テーブル API, 20
難読化
Ultra Light.NET 開発, 15

に

ニュースグループ
テクニカル・サポート, xiv

は

配置
Ultra Light.NET, 48
バグ
フィードバックの提供, xiv
パスワード
Ultra Light.NET 認証, 29
パブリケーション
Ultra Light.NET スキーマ情報, 27

ひ

表記規則
コマンド・シェル, xii
コマンド・プロンプト, xii
マニュアル, x
マニュアルでのファイル名, xi

ふ

フィードバック
エラーの報告, xiv
更新のご要望, xiv
提供, xiv
マニュアル, xiv
プラットフォーム
Ultra Light.NET サポート, 3

へ

ヘルプ
テクニカル・サポート, xiv
ヘルプへのアクセス
テクニカル・サポート, xiv

ま

マニュアル
SQL Anywhere, viii
表記規則, x
マルチスレッド・アプリケーション
Ultra Light.NET の概要, 13

も

モード
Ultra Light.NET, 21

ゆ

ユーザ認証
Ultra Light.NET 開発, 29
ユーザの認証
Ultra Light.NET 開発, 29

り

利点
Ultra Light.NET, 2

る

ルックアップ・メソッド
Ultra Light.NET, 22
ルックアップ・モード
Ultra Light.NET, 21

ろ

ロック
Ultra Light.NET キーワード, 31
ロー
Ultra Light.NET テーブル・アクセス、現在のロー, 21
Ultra Light.NET テーブル・ナビゲーション, 20
Ultra Light.NET を使用して Ultra Light ローを更新, 23
Ultra Light.NET を使用して Ultra Light ローを削除, 25
Ultra Light.NET を使用して Ultra Light ローを挿入, 24
ロールバック
Ultra Light.NET トランザクション, 26
ロールバック・メソッド
Ultra Light.NET トランザクション, 26

