

Oracle Connector for iOS/Android

version 2.5



目次

はじめに.....	5
Oracle connector for iOS/Androidについて	5
ご利用上の注意事項.....	6
本製品の使用許諾について	6
禁止事項	6
保証規定	6
プロダクト・サポート	7
販売元.....	9
開発元、サポート	9
インストール.....	10
サーバーシステム要件.....	10
モバイルデバイス要件.....	10
開発環境.....	10
セットアッププログラムの実行.....	11
IISサーバー設定について	11
データベース接続について	13
モバイルデバイスプログラミング	14
iOS swift/objective-cサンプルコード.....	14
Xcodeプロジェクトでの設定について	14
Info.plist 設定	14
Androidサンプルコード	15
Androidアプリケーション参照について	15
BUILD.GRADLE DEPENDENCY 追加について	15
AndroidManifest 設定について	15
Flutter / dart サンプルコード	16
IOSクラスリファレンス	17
メソッド	17
beginTrans.....	17
commitTrans.....	17
connect.....	17
disconnect.....	18
execute	18
query.....	18
queryRows.....	19
rollbackTrans.....	19
プロパティ	19
lastErrorString	19
lowerColName.....	20
pwd.....	20
rawUrl	20
results.....	20

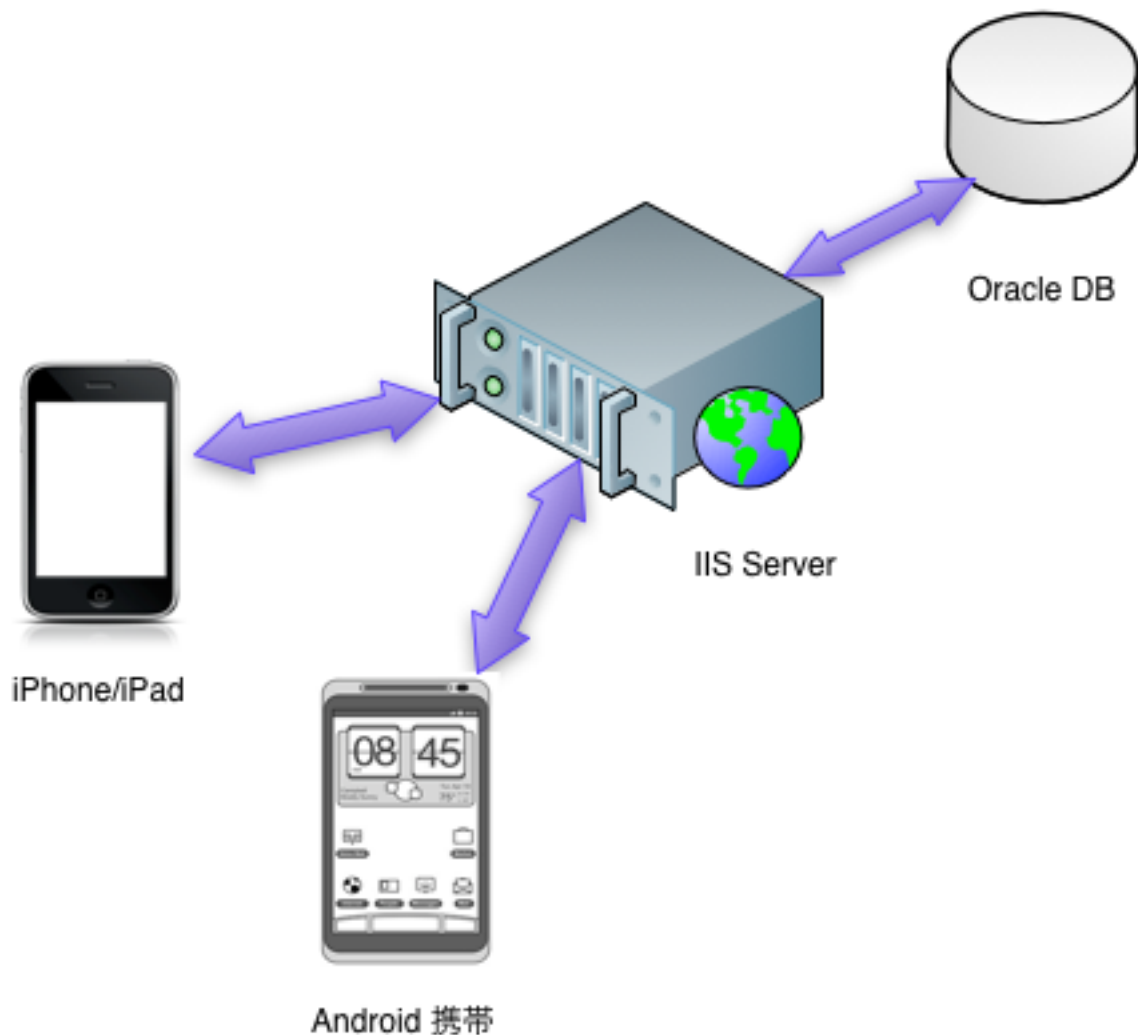
uid	20
uidl	21
url	21
verbose	21
デリゲート	21
queryCompleted	21
queryRowsCompleted	22
requestCompleted	22
requestFailed.....	22
rowFetched	23
ANDROIDクラスリファレンス	24
メソッド	24
beginTrans.....	24
commitTrans.....	24
connect.....	24
disconnect.....	25
execute	25
query.....	25
queryRows.....	26
rollbackTrans.....	26
プロパティ	26
lastErrorString	26
lowerColName.....	27
pwd.....	27
rawUrl	27
Row.....	27
Rows	28
uid	28
uidl	28
url	28
verbose	28
FLUTTER / DART クラスリファレンス	30
メソッド	30
beginTrans.....	30
commitTrans.....	30
connect.....	30
disconnect.....	31
execute	31
query.....	31
queryRows.....	32
rollbackTrans.....	32
プロパティ	33
lastException.....	33
lastHTTPStatusCode.....	33
lastServerError	33
networkTimeout.....	33
rawUrl	33

url	34
verbose	34
APPENDIX-A エラーコード	35
APPENDIX-B SWIFT サンプル	37
APPENDIX-C IOS OBJECTIVE-Cサンプルコード	39
APPENDIX-D サンプル WEB.CONFIG.....	40
APPENDIX-E ANDROID サンプルコード	41
APPENDIX-F FLUTTER/DART サンプルコード	42
ORACLE CONNECTOR FOR IOS/ANDROID 調査依頼.....	43

はじめに

Oracle connector for iOS/Androidについて

本製品はiPad,iPhoneやAndroidにOracleデータを開示してスマートフォンアプリケーションを作成したいお客様向けのモバイルミドルウェアです。モバイルデバイス向けのデータベースアプリケーションを容易に開発可能とします。



以下は製品の概要です。

1. Oracleサーバーとの通信はIISのWeb Service経由となります。
2. クライアントライブラリは以下の形式で提供します。
 - Xcode/Objective-Cに対応ユニバーサルスタティックライブラリ
 - Android SDK 対応.jarファイル
 - flutter/dart 対応 package
3. iOSクライアントライブラリは非同期通信でdelegateに処理完了などが通知されます。
4. Oracleトランザクションサポートしてますので更新系業務も対応可能です。

5. SSL通信に対応しています。
6. 各種 Web Server認証に対応しています。
7. flutterライブラリはdartで記述されており、flutterがサポートするプラットフォーム全般で動作します。

ご利用上の注意事項

本製品では以下のような注意事項・ご利用制限がございます。

1. BLOB/CLOBなどラージオブジェクトには対応していません。
本製品についてはWebServiceを利用してOracleと通信しているので現在のところラージオブジェクト（画像など）はObjective-CのNSURLRequestなどでBLOB/CLOBデータを転送する実装をお願いいたします。
2. Xcodeバージョンアップについて
例年9月のXcodeバージョンアップに伴いiOS frameworkバージョンアップによる仕様変更で動作が変わることがあります。迅速に対応しますのでサポートにお問い合わせください。
3. Flutter バージョンアップについて
頻繁にアップデートがありますのでgithubに対応版をアップロードする予定です。

本製品の使用許諾について

本製品のご利用ライセンスはサーバー毎となります。1 台のサーバーに複数のモバイルデバイスから接続数は無制限でご利用いただけます。開発ライブラリは複数の開発環境でご利用いただけます。複数サーバーでの運用、異なるアプリケーションを別サーバーで運用する場合などはご利用サーバー数と同じ数のライセンスを販売会社システムラボからご購入ください。

禁止事項

1. 本製品の不正複製を禁止します。
2. 本製品のリバースエンジニアリングを禁止します。
3. 本製品をラップし同種の製品を作成し販売することを禁止します。

保証規定

本製品、および付随する著作物に対して商品性及び特定の目的への適合性などについての保証を含まい

かなる保証もそれを明記するしないに関わらず提供されることはありません。

本製品の著作権及び、製造、配布に関わるいかなる者も、当ソフトウェアの不具合によって発生する損害に対する責任は、それが直接的であるか間接的であるか、必然的であるか偶発的であるかに関わらず、負わないものとします。それは、その損害の可能性について、開発会社に事前に知らされていた場合でも同様です。

プロダクト・サポート

- ユーザー登録

まことにお手数ですが販売会社システム・ラボにてユーザー登録をお願いします。ユーザー登録が行われていないとお客様がユーザー・サポートが受けられない場合がございます。

- お問い合わせの方法

どうしても解決できない問題が発生した場合には、技術サポートをご利用ください。あらかじめ後ページの調査依頼書にお問い合わせ事項を記入していただき、インターネット・メールまたはファックスでお送りいただければ、折り返しご連絡をさせていただきます。**本製品につきましては、複雑な内容のお問い合わせになることが多い為、電話によるユーザーサポートは実施しておりません。ご了承をお願いいたします。**また、問い合わせの内容によっては、再現調査などのために、回答までに時間がかかる場合がありますので、かさねてご了承をお願いいたします。

サポートメールアドレス： support@techknowledge.co.jp

- 登録内容の変更について

転居などによるご住所や電話番号など登録内容に変更が生じた場合には、メールまたはファックスにて、販売会社システム・ラボまでご連絡をいただきますようお願いいたします。なお、電話による口頭での連絡変更は受けかねますのでよりしくお願いいたします。

- 併用される他社製品について

当社製品と併用される、他社製品の使い方等についてのご質問をお受けすることがあります。しかし、他社製品に関しましては、お答えできない場合があります。他社製品につきましては、該当開発・販売会社にご連絡ください。

- サポート対象

ご質問はご登録ユーザー様に限定させていただきます。ご登録ユーザー様以外からのご質問にはお答えできません。当ソフトウェアの料金にはご登録ユーザー様1名に限りサポート料が含まれています。

- サポート期間

製品のユーザー登録後、初回のお問い合わせから90日間は無償サポート期間とさせていただきます。また無償サポートは2件を上限とさせていただきます。無償サポート上限を超える場合には無償サポート終了以降もサポートをご希望の場合は有償サポートを承ります。有償サポートにつきましては販社システム・ラボにてお取り扱いしております。キャンペーン製品などディスカウント販売に該当する製品では無償サポート期間および回数の設定が短くなる場合がありますのであらかじめご了承ください。

- 最新版のご提供について
弊社webにて最新版の実行モジュールや技術情報、サンプル・コードの提供をしてしておりますのでサポートにご連絡になる前に弊社webをご参照いただけるようお願いいたします。URLは<http://www.techknowledge.co.jp>となります。
- ご質問の内容について
製品サポートは本製品に関連するご質問に限定させていただきます。

販売元

Systemlab®

(株) システムラボ

東京都北区田端6-1-1 田端アスカタワー12F

電話	03-5809-0893
FAX	03-4578-9261
Internet-Mail	info@systemlab.co.jp
URL	www.systemlab.co.jp

開発元、サポート

TechKnowledge

(株) テクナレッジ

東京都世田谷区駒沢2丁目16番1号 サンドービル9F

電話	03-3421-7621
FAX	03-3421-6691
Internet-Mail	info@techknowledge.co.jp
URL	www.techknowledge.co.jp

商標登録

本マニュアルに記載される商標、登録商標は該当会社の商標または登録商標です。

インストール

Oracle connector iOS/Androidのインストールについてご説明します。

サーバーシステム要件

以下のソフトウェアがサーバーシステム構成の要件となります。

1. Windows Server 2012 ～ 2022
2. 上記のサーバーにて動作するMicrosoft IIS (Internet Information Server)
3. Microsoft .NET framework 4.72 以上
4. Oracle 10g ～ 21c (.NET framework 4.72のOracle Clientで接続可能なOracleサーバー)

モバイルデバイス要件

Apple社のモバイルデバイスとして以下をサーバーに接続可能です。

1. iPhone/iPad/iPod touch iOS 12以降
2. Android 携帯およびAndroidタブレット Android OS 6以降

開発環境

iOSモバイルデバイス向けアプリケーション開発環境としては以下が必要になります。Apple社のMac App Storeから無償でご利用いただけます。

Apple Xcode 13 以降

モバイルデバイス実機でのデバッグやアプリケーションの配布にはApple Developer Program の有償会員になる必要があります。（年次契約）

Apple Developer Program のウェブサイト

<https://developer.apple.com/jp/>

Androidモバイルデバイス向けアプリケーション開発環境としてはAndroid Studio が必要となります。開発環境の設定等についてはGoogleの該当サイトをご参照ください。

<https://developer.android.com/studio/>

Flutterでの開発に関してはAndroid StudioまたはVS code等で開発環境を整えてください。flutter 2.10.5～3.0.5で動作確認しました。

セットアッププログラムの実行

製品パッケージのsetup.exeをサーバー上で実行すると以下がインストールされます。

1. サーバー実行環境
2. モバイルデバイス向け開発環境

全てサーバー上に配布されますので開発環境はネットワーク接続等で開発用MAC/PCに転送してください。

IISサーバー設定について

サーバー動作に必要なファイルはインストールフォルダー下のhtdocsフォルダーに転送されます。

このフォルダー自体をIISで公開するか別のフォルダーに内容をコピーしてIISでウェブサイトとして公開してください。

I I Sの設定要件は以下になります。

1. .NET framework 4を利用可能。
2. System.Data.OracleClientが利用可能な設定。
3. セッションを有効化。
4. Oracleへの接続文字列を[Web.Config](#)に設定。

上記2番について[Web.Config](#)では以下のような設定となります。

```
<compilation debug="true">
<assemblies>
<add assembly="System.Data.OracleClient, Version=4.0.0.0, Culture=neutral,
PublicKeyToken=B77A5C561934E089"/></assemblies>
</compilation>
```

IIS 構成参考URL:

<https://docs.microsoft.com/ja-jp/aspnet/web-forms/overview/deployment/configuring-server-environments-for-web-deployment/configuring-a-web-server-for-web-deploy-publishing-offline-deployment>

以下はWindows2022サーバーでサーバー役割設定で[ASP.NET](#) 4.8をインストールするスクリーンショットです。

サーバーの役割の選択

開始する前に

インストールの種類

サーバーの選択

サーバーの役割

機能

確認

結果

選択したサーバーにインストールする役割を 1 つ以上選択します。

役割

- ☐ ログ ツール
- ☐ 要求の監視
- ☒ アプリケーション開発
 - ☐ .NET 拡張機能 3.5
 - ☒ .NET 拡張機能 4.8
 - ☐ Application Initialization
 - ☐ ASP
 - ☐ ASP.NET 3.5
 - ☒ ASP.NET 4.8
 - ☐ CGI
 - ☒ ISAPI フィルター
 - ☒ ISAPI 拡張
 - ☐ WebSocket プロトコル
 - ☐ サーバー側インクルード
- ☐ FTP サーバー
- ☒ 管理ツール (1/7 個をインストール済み)
 - ☒ IIS 管理コンソール (インストール済み)
 - ☐ IIS 6 管理互換
 - ☒ IIS 管理スクリプトおよびツール

インターネット インフォメーション サービス (IIS) マネージャー

WIN-OVE4HK503LM > サイト > test

ファイル(F) 表示(V) ヘルプ(H)

接続

- スタート ページ
- WIN-OVE4HK503LM (WIN-OV)
 - アプリケーション プール
 - サイト
 - Default Web Site
 - test
 - aspnet_client
 - bin

test ホーム

フィルター: 検索(G) すべて表示(A) グループ化:

ASP.NET

サイトの編集

サイト名(S): test アプリケーション プール(L): test 選択(E)...

物理パス(P): C:\Program Files (x86)\TechKnowledge\Oracle connecto ...

パススルー認証

接続(C)... テスト設定(G)...

OK キャンセル

ルール

IIS

HTTP 応答ヘッダー ISAPI フィルター MIME の種類 SSL 設定

機能ビュー コンテンツ ビュー

準備完了

操作

- エクスプローラー
- アクセス許可の編集...
- サイトの編集
 - バインド...
 - 基本設定...
 - アプリケーションの表示
 - 仮想ディレクトリの表示
- Web サイトの管理
 - 再起動
 - 開始
 - 停止
- Web サイトの参照
 - *:80 (http) 参照
 - 詳細設定...
- 構成
 - 制限...
 - HSTS...
- ヘルプ

データベース接続について

[Web.config](#)ファイルに接続文字列セクションに以下のようなOracleデータベース接続情報を追加してください。

```
<connectionStrings>  
<add name="oraLib" connectionString="User Id=scott;Password=tiger;Data  
Source=192.168.1.99:1521/orcl;" />  
</connectionStrings>
```

モバイルデバイスプログラミング

サンプルコードはOracle EMPテーブルにクエリを実行します。EMPテーブルを作成してデータを追加する必要がある場合はインストールフォルダー/sql にテーブルを生成、データを挿入するsqlがありますのでご利用ください。

iOS swift/objective-cサンプルコード

iOS向けサンプルコードはインストールディレクトリのios_samples/swift. ios_samples/objc フォルダに以下の2サンプルがございます。

1. querySample.zip
2. queryRowsSample.zip

それぞれ展開してXCodeで.xcodeprojファイルを読み込んでください。urlプロパティに設定しているサーバーのアドレス等に変更する必要があります。（プライベートメソッド startConnect でurlプロパティを設定しています）

OracleのサンプルEMP表にアクセスするようにサーバー側の接続文字列を設定していただければすぐに動作するようになります。

Xcodeプロジェクトでの設定について

Xcodeのプロジェクトから本製品でOracleデータベースアクセス機能を追加するには以下の2ファイルをプロジェクトに追加してください。

1. dblib.h
2. libdblib.a

これらのファイルはサンプルコードやサーバー側にinclude/lib フォルダに保存されています。liboralib.a はデバイス用とシュミレータ用に別々のファイルをご用意されておりますのでご注意ください。ヘッダーファイルは共通でご利用いただけます。

Info.plist 設定

iOS14以降のデバイスでローカルネットワークを使う場合の設定例です。

```
<key>NSLocalNetworkUsageDescription</key>
<string>${PRODUCT_NAME} uses the local network.</string>
```

Androidサンプルコード

Android向けサンプルコードはインストールディレクトリのandroid_samplesフォルダーに以下の3サンプルがごさいます。

1. querySample.zip
2. queryRowsSample.zip
3. executeSample.zip

それぞれ展開してAndroid StudioからJavaプロジェクトをインポートしてください。dbLibクラスのurlプロパティに設定しているサーバーのアドレス等は変更する必要があります。

OracleのサンプルEMP表にアクセスするようにサーバー側の接続文字列を設定していただければすぐに動作するようになります。

Androidアプリケーション参照について

アプリケーションプロジェクトのLibsには以下のaarファイルを追加してください。

dblib.aar

ソースコード上では以下のimport が必要になります。

```
import jp.co.techknowledge.dbLib;  
import jp.co.techknowledge.dbLib.Row;  
import jp.co.techknowledge.dbLib.Column;  
import jp.co.techknowledge.dbLib.status;
```

Build.gradle dependency 追加について

okhttp3を使っていますので以下をdependencyへ追加してください。

```
dependencies {  
    // snip ...  
    implementation 'com.squareup.okhttp3:okhttp:4.9.0'  
    implementation 'com.squareup.okhttp3:okhttp-urlconnection:4.9.0'  
    implementation fileTree(dir: 'libs', include: ['*.aar', '*.jar'], exclude: [])  
}
```

AndroidManifest 設定について

当クラスライブラリの利用にあたって以下のパーミッション指定を必要とします。

・ INTERNET

Android.Manifestファイルでの設定例は以下です。

```
<uses-permission android:name="android.permission.INTERNET" />
```

ローカルサーバーとSSLなしで接続する場合は以下を追加してください。

```
android:usesCleartextTraffic="true"
```

Flutter / dart サンプルコード

サンプルコードはリポジトリ `techknowledge-dev/dblib_dart/example` にありますのでご参照ください。null safety 対応です。

pubspec.yaml の dependency: に以下のように設定してください。

dependencies:

dblib:

git:

url: https://github.com/techknowledge-dev/dblib_dart.git

各プラットフォームでのビルド・実行方法はflutterのウェブなどをご確認ください。

<https://docs.flutter.dev/get-started/codelab>

実デバイスに必要なネットワーク利用系のパーミッションはiOS/Androidの項と同様です。

iOSクラスリファレンス

メソッド

beginTransaction

書式

-(dbLibStatus) beginTrans

戻り値

Normal時は呼び出し成功。それ以外はAppendix-A参照。

解説

Oracleトランザクションを開始します。サーバーには接続完了状態、トランザクションクローズ状態で呼び出してください。サーバーからの実行結果はrequestCompleted/requestFailedデリゲートに通知されます。

commitTrans

書式

-(dbLibStatus) commitTrans

戻り値

Normal時は呼び出し成功。それ以外はAppendix-A参照。

解説

Oracleトランザクションをコミットします。サーバーには接続完了状態、トランザクション状態で呼び出してください。サーバーからの実行結果はrequestCompleted/requestFailedデリゲートに通知されます。

connect

書式

-(dbLibStatus) connect

戻り値

Normal時は呼び出し成功。それ以外はAppendix-A参照。

解説

プロパティURLで指定されるサーバーに接続します。サーバーからの実行結果はrequestCompleted/requestFailedデリゲートに通知されます。

disconnect

書式

-(dbLibStatus) disconnect

戻り値

Normal時は呼び出し成功。それ以外はAppendix-A参照。

解説

サーバー接続を遮断します。サーバー側ではOracle接続の遮断、トランザクション中の場合はトランザクションの破棄が実行されます。サーバーからの実行結果はrequestCompleted/requestFailedデリゲートに通知されます。

execute

書式

-(dbLibStatus) execute:(NSString*) sql

パラメータ

非クエリ系Oracle SQL

戻り値

Normal時は呼び出し成功。それ以外はAppendix-A参照。

解説

非クエリ系Oracle SQLを実行します。サーバーからの実行結果はrequestCompleted/requestFailedデリゲートに通知されます。

query

書式

-(dbLibStatus) query:(NSString*) sql

戻り値

Normal時は呼び出し成功。それ以外はAppendix-A参照。

解説

SQLで指定されるクエリを発行して1行ずつrowFetchedデリゲートにデータが通知されます。データの読み込みが完了するとqueryCompletedデリゲートに通知されます。

queryRows

書式

-(dbLibStatus) queryRows:(NSString*) sql: (NSInteger) maxRows

戻り値

Normal時は呼び出し成功。それ以外はAppendix-A参照。

解説

SQLで指定されるクエリを発行して複数行をqueryRowsCompletedデリゲートに通知します。HTTPプロトコルで転送されるデータ量に上限がありますので一定のデータ量で制限したいときには第2パラメータmaxRowsに取得するデータ行の最大数を設定します。maxRowsにゼロ設定の場合は全データを取得します。

rollbackTrans

書式

-(dbfLibStatus) rollbackTrans

戻り値

Normal時は呼び出し成功。それ以外はAppendix-A参照。

解説

Oracleトランザクションをロールバックします。サーバーには接続完了状態、トランザクション状態で呼び出してください。サーバーからの実行結果はrequestCompleted/requestFailedデリゲートに通知されます。

プロパティ

lastErrorString

データ型

NSString*

解説

本製品のメソッドを呼び出してエラーが発生した場合にその理由が明確な場合はこのプロパティに保持されます。エラー情報は以下となります。

1. ネットワークエラー情報
2. サーバー側エラー情報
3. Oracleエラーメッセージ

lowerColName

データ型

bool

解説

True設定時にクエリー結果のカラム名を小文字に変換します。

pwd

データ型

NSString*

解説

IISでウェブ認証を使う場合にはこのプロパティにパスワードを設定してください。

rawUrl

データ型

bool

解説

サーバーがLinux版の場合にはTrueします。Windows ServerではFalse設定します。

results

データ型

NSMutableArray*

解説

Query / QueryRowsメソッドの結果を保持します。配列の要素はNSMutableDictionary*型でカラム名指定でカラムデータを取得する事が出来ます。

uid

データ型

NSString*

解説

IISでウェブ認証を使う場合にはこのプロパティにユーザーIDを設定してください。

uidl

データ型

NSString*

解説

デバイスを特定出来るユニークな文字列を設定してください。設定しない場合はデバイスUIDL値をライブラリが設定します。将来的にUIDL値をiOSから得ることが出来なりましたら、設定は必須となることがあります。

url

データ型

NSString*

解説

接続先のサーバURLを指定します。（例：<http://some.server.jp/>） サーバーの構成によりSSL接続やポート番号指定も可能です。

verbose

データ型

bool

解説

サーバーから受信したXMLデータ内容をログに出力します。

デリゲート

queryCompleted

書式

-(void) queryCompleted

戻り値

なし。

解説

queryメソッドが完了したときに呼び出されます。データがそろった状態なのでUITableViewのリロードなどを呼び出します。

queryRowsCompleted

書式

-(void) queryRowsCompleted:(NSMutableArray*) rows

戻り値

なし。

解説

queryRowsメソッドが完了したときに呼び出されます。パラメータにはNSMutableArrayで取得したレコード行が複数返されます。NSMutableArrayが保持するのはNSDictionaryオブジェクトでカラム名をキーとしてデータを取得します。キーのカラム名は通常大文字となります。以下サンプルコードです。

```
NSMutableDictionary* row = [_rows objectAtIndex:indexPath.row];
NSString* empno = [row objectForKey:@"EMPNO"];
NSString* ename = [row objectForKey:@"ENAME"];
cell.textLabel.text = [NSString stringWithFormat:@"%@@ %@", empno, ename];
```

requestCompleted

書式

-(void) requestCompleted:(NSString*)methodName;

戻り値

なし。

解説

メソッドの正常完了を通知するデリゲートです。クエリ系以外のメソッドでこのデリゲートに通知となります。methodNameには発行したメソッドの名前が通知されます。ただしconnectとdisconnectは通知メソッド名Login/Logoutとなります。

requestFailed

書式

-(void) requestFailed:(NSString*)methodName :(NSError*) err

戻り値

なし。

解説

メソッドの異常終了を通知するデリゲートです。クエリ系以外のメソッドでこのデリゲートに通知となります。methodNameには発行したメソッドの名前が通知されます。ただしconnectとdisconnectは通知メソッド名Login/Logoutとなります。NSErrorについてはネットワーク系エラー以外はnilが指定されます。NSErrorの詳細はAppleのマニュアルをご参照ください。

rowFetched

書式

-(bool) rowFetched:(NSMutableDictionary*) row

戻り値

Trueを返すと次のレコードを取得します。Falseを返すとこのレコードで終了となります。

解説

queryメソッド実行後にレコードを受信するたびにこのデリゲートが呼び出されます。パラメータにはレコードイメージがNSDictionary型で保持されます。キーはカラム名となります。カラム名は通常大文字になります。lowerColNameプロパティのTrue設定により小文字に変換することが出来ます。以下はサンプルコードです。

```
NSString* empno = [row objectForKey:@"EMPNO"];
NSString* ename = [row objectForKey:@"ENAME"];
```

Androidクラスリファレンス

dbLibクラスメソッドの呼び出しにつきましてはサーバーとの通信が伴いますのでメインスレッド以外から呼び出すように実装してください。

メソッド

beginTransaction

書式

```
status beginTrans();
```

戻り値

status.Normal時は呼び出し成功。それ以外はAppendix-A参照。

解説

Oracleトランザクションを開始します。サーバーには接続完了状態、トランザクションクローズ状態で呼び出してください。サーバーからの実行結果はrequestCompleted/requestFailedデリゲートに通知されます。

commitTrans

書式

```
status commitTrans();
```

戻り値

status.Normal時は呼び出し成功。それ以外はAppendix-A参照。

解説

Oracleトランザクションをコミットします。サーバーには接続完了状態、トランザクション状態で呼び出してください。サーバーからの実行結果はrequestCompleted/requestFailedデリゲートに通知されます。

connect

書式

```
status connect();
```


戻り値

status.Normal時は呼び出し成功。それ以外はAppendix-A参照。

解説

プロパティURLで指定されるサーバーに接続します。サーバーからの実行結果はrequestCompleted/requestFailedデリゲートに通知されます。

disconnect

書式

```
status disconnect();
```

戻り値

status.Normal時は呼び出し成功。それ以外はAppendix-A参照。

解説

サーバー接続を遮断します。サーバー側ではOracle接続の遮断、トランザクション中の場合はトランザクションの破棄が実行されます。サーバーからの実行結果はrequestCompleted/requestFailedデリゲートに通知されます。

execute

書式

```
status execute(String sql);
```

パラメータ

非クエリ系Oracle SQL

戻り値

status.Normal時は呼び出し成功。それ以外はAppendix-A参照。

解説

非クエリ系Oracle SQLを実行します。サーバーからの実行結果はrequestCompleted/requestFailedデリゲートに通知されます。

query

書式

```
status query(String sql);
```

戻り値

status.Normal時は呼び出し成功。それ以外はAppendix-A参照。

解説

SQLで指定されるクエリを発行して1行ずつrowFetchedデリゲートにデータが通知されます。データの読み込みが完了するとqueryCompletedデリゲートに通知されます。

queryRows

書式

```
status queryRows(String sql, int maxRows);
```

戻り値

status.Normal時は呼び出し成功。それ以外はAppendix-A参照。

解説

SQLで指定されるクエリを発行して複数行をqueryRowsCompletedデリゲートに通知します。HTTPプロトコルで転送されるデータ量に上限がありますので一定のデータ量で制限したいときには第2パラメータmaxRowsに取得するデータ行の最大数を設定します。maxRowsにゼロ設定の場合は全データを取得します。

rollbackTrans

書式

```
status rollbackTrans();
```

戻り値

status.Normal時は呼び出し成功。それ以外はAppendix-A参照。

解説

Oracleトランザクションをロールバックします。サーバーには接続完了状態、トランザクション状態で呼び出してください。サーバーからの実行結果はrequestCompleted/requestFailedデリゲートに通知されます。

プロパティ

lastErrorString

データ型

String

解説

本製品のメソッドを呼び出してエラーが発生した場合にその理由が明確な場合はこのプロパティに保持されます。エラー情報は以下となります。

1. ネットワークエラー情報
2. サーバー側エラー情報
3. Oracleエラーメッセージ

lowerColName

データ型

bool

解説

True設定時にクエリー結果のカラム名を小文字に変換します。

pwd

データ型

NSString*

解説

IISでウェブ認証を使う場合にはこのプロパティにパスワードを設定してください。

rawUrl

データ型

bool

解説

サーバーがLinux版の場合にはTrueします。Windows ServerではFalse設定します。

Row

データ型

Row クラス

解説

Query メソッドの結果を保持します。

Rows

データ型

ArrayList<Row>

解説

Query メソッド(オーバーロードの上限レコード数指定)の結果を保持します。

uid

データ型

String

解説

IISでウェブ認証を使う場合にはこのプロパティにユーザーIDを設定してください。

uidl

データ型

String

解説

デバイスを特定出来るユニークな文字列を設定してください。アプリ側から設定しない場合はライブラリ内部にてデバイスIDを自動設定します。実際に設定する値は`android.os.Build.SERIAL`になりますがこの値が取得出来ないモバイルデバイスではこのプロパティ値の設定は必須となります。このID単位にてサーバーにデータを保持しますのでサーバーのデータベースに同時アクセスする複数のデバイスについて重複しない値を設定する必要があります。

url

データ型

String

解説

接続先のサーバURLを指定します。(例: <http://some.server.jp/>) サーバーの構成によりSSL接続やポート番号指定も可能です。

verbose

データ型

bool

解説

サーバーから受信したXMLデータ内容をログに出力します。ログのタグは”dblib”となります。

flutter / dart クラスリファレンス

DbLib クラスに定義されるメソッドはネットワークI/Oを伴いますので全て `async` 属性を持ちます。設定必須プロパティは `url` のみです。

メソッド

beginTransaction

書式

```
Future<status> beginTrans() async;
```

戻り値

`status.Normal`時は呼び出し成功。それ以外はAppendix-A参照。

解説

Oracleトランザクションを開始します。サーバーには接続完了状態、トランザクションクローズ状態で呼び出してください。サーバーからの実行結果は`requestCompleted/requestFailed`デリゲートに通知されます。

commitTrans

書式

```
Future<status> commitTrans() async;
```

戻り値

`status.Normal`時は呼び出し成功。それ以外はAppendix-A参照。

解説

Oracleトランザクションをコミットします。サーバーには接続完了状態、トランザクション状態で呼び出してください。サーバーからの実行結果は`requestCompleted/requestFailed`デリゲートに通知されます。

connect

書式

```
Future<status> connect() async;
```

戻り値

status.Normal時は呼び出し成功。それ以外はAppendix-A参照。

解説

プロパティURLで指定されるサーバーに接続します。サーバーからの実行結果はrequestCompleted/requestFailedデリゲートに通知されます。

disconnect

書式

Future<status> disconnect() async;

戻り値

status.Normal時は呼び出し成功。それ以外はAppendix-A参照。

解説

サーバー接続を遮断します。サーバー側ではOracle接続の遮断、トランザクション中の場合はトランザクションの破棄が実行されます。サーバーからの実行結果はrequestCompleted/requestFailedデリゲートに通知されます。

execute

書式

Future<status> execute(String sql) async;

パラメータ

非クエリ系Oracle SQL

戻り値

status.Normal時は呼び出し成功。それ以外はAppendix-A参照。

解説

非クエリ系Oracle SQLを実行します。サーバーからの実行結果はrequestCompleted/requestFailedデリゲートに通知されます。

query

書式

Future<QueryResult> query(String sql) async;

戻り値

QueryStatus.status == normal時は呼び出し成功。それ以外はAppendix-A参照。

解説

SQLで指定されるクエリを発行成功すると先頭1行のデータがQueryStatus.resultに戻されます。後続データはfetchメソッドで同様に取得します。

queryRows

書式

```
Future<QueryRowsResult> queryRows(String sql, int maxRows) async;
```

戻り値

status.Normal時は呼び出し成功。それ以外はAppendix-A参照。

解説

SQLで指定されるクエリを発行して複数行をQueryRowsResult.resultに取得します。HTTPプロトコルで転送されるデータ量に上限がありますので一定のデータ量で制限したいときには第2パラメータmaxRowsに取得するデータ行の最大数を設定します。maxRowsにゼロ設定の場合は全データを取得します。

rollbackTrans

書式

```
Future<status> rollbackTrans() async;
```

戻り値

status.Normal時は呼び出し成功。それ以外はAppendix-A参照。

解説

Oracleトランザクションをロールバックします。サーバーには接続完了状態、トランザクション状態で呼び出してください。サーバーからの実行結果はrequestCompleted/requestFailedデリゲートに通知されます。

プロパティ

lastException

データ型

Exception

詳細

最後に発生した例外を保持します。

lastHTTPStatusCode

データ型

int

詳細

最後に発生したhttp status codeを保持します。

lastServerError

データ型

String

詳細

最後に発生したサーバー側のエラーや例外を保持します。

networkTimeout

データ型

int

詳細

サーバーとのhttp/https通信タイムアウトをミリ秒単位で指定します。

rawUrl

データ型

bool

詳細

Windows Server にてサーバーアプリをルートフォルダ以外に設置した場合はTrue設定してurlプロパティにパスを指定します。

url

データ型

String

詳細

サーバーURIを指定します。プロトコル指定必要でサーバーアプリをルートに設置した場合はサーバーアセンブリ名は省略できます。

verbose

データ型

bool

詳細

true設定でログを詳細に出力します。

Appendix-A エラーコード

iOS / Android メソッド呼び出し時に返るdbLibStatusです。

定義値	値	詳細
Normal	0	正常終了
NetworkFail	1	ネットワーク接続が出来ませんでした。URLプロパティと実際のネットワーク接続状態をご確認ください。
NoURL	2	URLプロパティの指定がありませんでした。
NoUID	3	UIDプロパティの指定がありませんでした。
NoPwd	4	PWDプロパティの指定がありませんでした。
WebServiceFailed	5	Web Serviceがエラーを返しました。
WebServiceException	6	Web Serviceが例外を返しました。lastErrorTextに詳細が保持される場合がありますのでご確認ください。
SqlEmpty	7	SQLパラメータ指定がありません。
StillInRequest	8	他のリクエストが終了していないため、新たなメソッドの呼び出しが出来ません。
ioError	9	ネットワーク通信時にエラーとなりました。安定したネットワークに接続して、再度実行してください。
protocolError	10	Soapサービスのプロトコルに沿っていないデータを受信しました。接続先のサーバーが当製品のサーバーでは無い可能性があります。
resultParseError	11	サーバーから戻されたレスポンスを解析出来ませんでした。
NoSql	12	メソッドパラメータに必要なSQL指定がありませんでした。
unknownSoapService	13	当システムにて認識できないSoapService名がサーバーから戻されました。

flutter メソッド呼び出し時に返るStatusEnumです。

定義値	詳細
normal	正常終了
notConnected	connectメソッドが呼び出されていません。
connectFail	サーバーに接続できませんでした。lastHttpStatusCodeプロパティなどを確認してください。
endOfData	Web Serviceがエラーを返しました。
methodSequeneError	メソッドの呼び出し順序が違います。
parseXMLError	サーバーからのデータをパースできませんでした。
sqlEmpty	SQLパラメータ指定がありません。
networkFail	ネットワークが正常に動作していません。
serverError	サーバーから戻されたレスポンスを解析出来ませんでした。
httpError	http Status 200以外がサーバーから返されました。
fatal	致命的なエラー。例外発生していますのでlastExceptionを参照するかverbose=true設定でログを参照してください。

Appendix-B swift サンプル

```
import UIKit

class ViewController: UIViewController,UITableViewDelegate,UITableViewDataSource,
dbLibDelegate {
    @IBOutlet weak var _tableView: UITableView!
    private var _rows:NSMutableArray!
    private var _db:dbLib!

    //
    // MARK: dblib code
    //
    func startConnect() {

        _db = dbLib.init()
        _db.delegate = self
        _db.url = "http://192.168.179.8/"
        _db.verbose = true
        let rc = _db.connect()
        if rc != Normal {
            print("connect failed \(rc)")
            return
        }
    }

    func requestCompleted(_ methodName: String!) {
        print("request completed:" + methodName)

        if methodName == "Login" {
            let qrc = _db.queryRows("select EMPNO,ENAME from emp order by empno", 100)
            print("query result= \(qrc)")
        }
    }

    func requestFailed(_ methdName: String!, _ error: Error!) {
        print("request failed \(String(describing: error))")
    }

    func queryRowsCompleted(_ rows: NSMutableArray!) {
        DispatchQueue.main.async {
            self._rows = rows
            self._tableView.reloadData()
            self._db.disconnect()
        }
    }
}
```

```

override func viewDidLoad() {
    super.viewDidLoad()
    _tableView.register(UITableViewCell.self, forCellReuseIdentifier: "Cell")
}

override func viewWillAppear(_ animated: Bool) {
    super.viewWillAppear(animated)
    self.startConnect()
}

func tableView(_ tableView: UITableView, numberOfRowsInSection section: Int) -> Int {
    if _rows == nil {
        return 0
    }
    return _rows.count
}

func tableView(_ tableView: UITableView, cellForRowAt indexPath: IndexPath) ->
UITableViewCell {
    let cell:UITableViewCell = tableView.dequeueReusableCell(withIdentifier: "Cell")!

    let row:NSDictionary = _rows[indexPath.row] as! NSDictionary
    let empno = row.value(forKey: "EMPNO") as! String
    let ename = row.value(forKey: "ENAME") as! String
    cell.textLabel!.text = empno + " " + ename
    return cell
}

func tableView(_ tableView: UITableView, willDisplay cell: UITableViewCell, forRowAt indexPath:
IndexPath) {

    tableView.separatorInset = UIEdgeInsets.zero
    tableView.layoutMargins = UIEdgeInsets.zero
    cell.layoutMargins = UIEdgeInsets.zero

}

func tableView(_ tableView: UITableView, didSelectRowAt indexPath: IndexPath) {
    tableView.deselectRow(at: indexPath, animated: true)
}
}

```

Appendix-C iOS Objective-Cサンプルコード

```
-(void) startConnect {

    [UIApplication sharedApplication].networkActivityIndicatorVisible = YES;

    _db = [[dbLib alloc] init];
    _db.url = @"http://some.domain.jp";
    _db.delegate = self;

    dbLibStatus rc = [_db connect];

    if(rc != Normal) {

        NSLog(@"connect failed %d",rc);
        return;
    }

}

-(void) startQueryRows {

    bool rc = [_db queryRows:@"select empno,ename from emp order by empno" :0];

    if(rc != Normal){
        NSLog(@"query rows request failed");
    }
}

-(void) queryRowsCompleted:(NSMutableArray *)rows {

    _rows = [rows retain];

    [_table reloadData];
    [_db disconnect];
    [UIApplication sharedApplication].networkActivityIndicatorVisible = NO;

}
```

Appendix-D サンプル [Web.config](#)

```
<?xml version="1.0"?>
<configuration>
  <appSettings/>
  <connectionStrings>
    <add name="oraLib" connectionString="User Id=scott;Password=tiger;Data
Source=192.168.1.71:1521/orcl;>/>
  </connectionStrings>
  <system.web>
    <compilation debug="true">
      <assemblies>
        <add assembly="System.Data.OracleClient, Version=4.0.0.0, Culture=neutral,
PublicKeyToken=B77A5C561934E089">/>
      </assemblies>
    </compilation>
    <authentication mode="Windows"/>
  </system.web>
  <system.webServer>
    <directoryBrowse enabled="true"/>
  </system.webServer>
</configuration>
```


Appendix-E Android サンプルコード

```
try {
    dbLib.status st;
    dbLib db = new dbLib();
    db.setURL("http://192.168.0.5");

    st = db.connect();
    if(st == status.Normal){
        st = db.query("select empno,ename from emp",100);
        if(st == status.Normal){
            //
            ArrayList<dbLib.Row> rows = db.getRows();
            //
            for(int i=0; i<rows.size(); i++){
                //
                Row row = rows.get(i);
                Column col = row.columns.get(0);
                String empno = col.value;
                col = row.columns.get(1);
                String ename = col.value;
                Log.v("test","result=" + empno + "," + ename);
            }
        }
    }
    db.disconnect();
}
catch(Exception ex){
    Log.v("test",ex.getMessage());
    ex.printStackTrace();
}
```

Appendix-F flutter/dart サンプルコード

```
import 'package:dblib/dblib.dart';

Future<void> sample() async {

    final dblib = DbLib();
    dblib.url = "http://192.168.128.10";
    dblib.verbose = true;
    var st = await dblib.connect();
    var res = await dblib.query("select * from EMP");
    while (res.status==StatusEnum.normal){
        _dumpRow(res.result);
        res = await dblib.fetch();
    }
    st = await dblib.disconnect();
}

void _dumpRow(res){
    print('-----');
    res.forEach((key,item){
        print('$key = $item');
    });
}
```

Oracle connector for iOS/Android 調査依頼

日付	
会社名	
登録ユーザー名	
製品シリアル番号	
製品バージョン	
電話番号	
ファックス番号	
電子メールアドレス	
使用パソコン機種	
利用端末名 (iPhone/iPad/iPod/Android)	
iOSまたはAndroid バージョン	
開発環境バージョン	
お問合わせ内容、問題記述など、具体的に再現可能なようにご記入ください。	
添付資料	

Oracle connector for iOS/Android
システムご利用の手引

第2.5版

2022年9月26日

版權・著作 株式会社テクノレッジ
Printed In Japan