POP3 プロバイダ

Version 1.0.0

ユーザーズ ガイド

January 15, 2015

備考:

【改版履歴】

バージョン	日付	内容
1.0.0	2015-01-15	初版.

目次

1. はじめに

POP3 プロバイダは，POP3 のプロトコル処理に従い，インターネットメール（e-Mail）の受信を行うプロバイダです．本プロバイダを利用することで，UIDL コマンドに対応している POP3 サーバーからメールをクライアントにダウンロード・削除することができます．

本ドキュメントでは，POP3 プロバイダの概要と，実装されている CAO インタフェース（関数仕様）について説明しています．

2. プロバイダの概要

2.1. インストール

POP3 プロバイダモジュールは，下記の DLL で構成されています．ORiN2 SDK のインストーラでインストールした場合は，インストール作業は不要です．手動でインストールする場合は，表 2-1 のように実行してください．

表 2-1 POP3 プロバイダ

ファイル名	CaoProvPOP3.dll
ProgID	CaoProv.POP3
レジストリ登録	regsvr32 CaoProvPOP3.dll
レジストリ登録の抹消	regsvr32 /u CaoProvPOP3.dll

本プロバイダは POP3 サーバーと接続し，メール取得を行うために POCO C++ Libraries を使用しています．

2.2. 概要

POP3 プロバイダは，UIDL コマンド対応 POP3 サーバーと接続し，メール一覧の取得，メールメッセージの取得，及びメールの削除を行います．

2.3. メソッド・プロパティ

2.3.1. CaoWorkspace::AddController メソッド

書式　AddController (<bstrCtrlName:BSTR>, <bstrProvName:BSTR>, <bstrPcName:BSTR>, [<bstrOption:BSTR>])

bstrOption には，サーバーのドメイン名（または IP アドレス），ポート番号，ユーザー名，パスワード，タイムアウト時間を設定します．

[設定項目]=[設定内容] の形で，カンマ(,)区切りで指定します．

設定例： HOSTNAME=127.0.0.1,USER=Test,PASS=Test,PORT=110,TIMEOUT=10000

< bstrCtrlName > : [in] コントローラ名

< bstrProvName > : [in] プロバイダ名．固定値 =”CaoProv.POP3”

<bstrPcName> : [in] プロバイダの実行マシン名（未使用）

<bstrOption> : [in] オプション文字列

HOSTNAME	:	接続先を指定します．（必須） 接続先となる POP3 サーバーのドメイン名，または IP アドレスを指定してください． 例）HOSTNAME=127.0.0.1
USER	:	ユーザー名を指定します．（必須）
PASSWORD	:	パスワードを指定します．（必須）
PORT	:	ポート番号を指定します．（省略可能） 省略された場合は 110 が使用されます．
TIMEOUT	:	タイムアウト時間をミリ秒で指定します．（省略可能） int 型の範囲内の数値を入力してください． 省略された場合は 0（無限待ち）が使用されます．

2.3.2. CaoController::AddFile メソッド

書式　AddFile (<bstrName:BSTR>, <bstrOption:BSTR>)

指定された UIDL に対応するメールの解析を行い，メールの情報を保持する CaoFile を作成します．

bstrOption には，メールの UIDL を設定します．

[設定項目]=[設定内容] の形で指定します．

設定例： UIDL=TestUIDL

< bstrName > : [in] ファイル名
<bstrOption> : [in] オプション文字列
UIDL : メールの UIDL を指定します.（必須）

2.3.3. CaoFile::get_Value プロパティ

CaoController::AddFile にて指定された UIDL に対応するメールの本文となるメッセージを文字列の配列で取得します.

基本的に Content-Disposition が attachment ではなく, かつ Content-Type が text/plain であるパートの Body 部分を取得します.

例外として, メールが multipart/digest の場合, Content-Disposition が attachment ではなく, かつ Content-Type が message/rfc822 であるパートの Body 部分を取得します.

3. コマンドリファレンス

3.1. Controller クラス

表 3-1 CaoController::Execute コマンド一覧

3.1.1. CaoController::Execute(“MessageCount”) コマンド

POP3 サーバーに存在するメール件数を取得します.

書式　MessageCount ()

戻り値　：　[out] メール件数.（VT_ I4）

使用例

```
result = m_caoController.Execute("MessageCount")
```

3.1.2. CaoController::Execute(“ListUIDL”) コマンド

メールの UIDL 一覧を取得します.

本コマンドで取得した UIDL をもとに, Size, DeleteMessage コマンド, 及び CaoController::AddFile メソッドを実行します.

書式　ListUIDL()

戻り値　：　[out] メール UIDL 一覧.（VT_ARRAY）
メール UIDL 一覧を VT_BSTR の配列として取得します.

使用例

```
result = m_caoController.Execute("ListUIDL")
```

3.1.3. CaoController::Execute(“Size”) コマンド

指定したメールのサイズをバイト数で取得します.

書式　Size (< UIDL >)

< UIDL >	:	[in] メール UIDL.（VT_BSTR）
戻り値	:	[out] メールのサイズ（Byte）.（VT_I4）

使用例

```
result = m_caoController.Execute("UIDL", "TestUIDL")
```

3.1.4. CaoController::Execute(“DeleteMessage”) コマンド

指定したメールをサーバーから削除します.

書式　DeleteMessage (< UIDL >)

< UIDL >	:	[in] メール UIDL.（VT_BSTR）

使用例

```
m_caoController.Execute("DeleteMessage", "TestUIDL")
```

3.1.5. CaoController::Execute(“SetTimeout”) コマンド

タイムアウト時間をミリ秒で設定します.

書式　SetTimeout (< Data >)

< Data >	:	[in] タイムアウト時間（ミリ秒）.（VT_I4）

使用例

```
m_caoController.Execute("SetTimeout", 10)
```

3.1.6. CaoController::Execute(“GetTimeout”) コマンド

タイムアウト時間をミリ秒で取得します.

書式　GetTimeout ()

戻り値　：　[out] タイムアウト時間（ミリ秒）.（VT_I4）

使用例

```
result = m_caoController.Execute("GetTimeout")
```

3.1.7. CaoController::Execute(“ProviderCancel”) コマンド

メール UIDL 一覧取得処理（ListUIDL），CaoController::AddFile メソッド，File クラスの添付ファイル取得処理（GetBody）実行中に本コマンドを実行すると，処理を中断します.

ProviderCancel コマンドにより処理を中断した場合は，ProviderClear コマンドによりキャンセル状態を解除しなければ，他の処理は実行できません.

書式　ProviderCancel ()

使用例

```
m_caoController.Execute("ProviderCancel")
```

3.1.8. CaoController::Execute(“ProviderClear”) コマンド

ProviderCancel コマンドによるキャンセル状態を解除します.

書式　ProviderClear()

使用例

```
m_caoController.Execute("ProviderClear ")
```

3.2. File クラス

表 3-2 CaoFile::Execute コマンド一覧

コマンド	機能	ページ
GetHeaderNames	ヘッダ名一覧を取得します.	P.11
GetHeader	指定されたヘッダ名に対応する値を取得します.	P.11
GetBodyCount	メールの Body(Part)の数を取得します.	P.12
GetBodyFileName	指定された Body(Part)のファイル名を取得します.	P.12
GetBody	指定された Body(Part)の本文を取得します.	P.13
GetCodePage	指定された Body をエンコードしているコードページを取得します.	P.13
GetLowMessage	解析前の生のメールメッセージを取得します.	P.13

3.2.1. CaoFile::Execute(“GetHeaderName”) コマンド

ヘッダ名の一覧を取得します.

書式　GetHeadeName ()

戻り値　:　[out] ヘッダ名の一覧.(VT_ARRAY)
ヘッダ名の一覧を VT_BSTR の配列として取得します.

使用例

```
result = m_caoFile.Execute("GetHeaderName")
```

3.2.2. CaoFile::Execute(“GetHeader”) コマンド

指定されたヘッダ名に対応するヘッダーの値を取得します.
MIME 方式でエンコードされている場合は, デコード後の値を返します.

書式　GetHeader (< HeaderName >)

< HeaderName >　:　[in] ヘッダ名.(VT_BSTR)
戻り値　:　[out] ヘッダ名に対応するヘッダの値.(VT_BSTR | VT_ARRAY)
ヘッダ名が[From], [To], [CC], [Reply-To]の場合, 対応する値である複数のアドレスを VT_ARRAY にて返します.

それ以外は VT_BSTR にて対応する値を返します.

使用例

```
result = m_caoFile.Execute("GetHeader", "HeaderName")
```

3.2.3. CaoFile::Execute(“GetBodyCount”) コマンド

メールの Body 数を取得します.

対象のメールがシングルパートの場合は 1, マルチパートの場合は 1 より大きい数となります.

書式 GetBodyCount ()

戻り値 : [out] メールの Body 数.（VT_I4）

使用例

```
result = m_caoFile.Execute("GetBodyCount")
```

3.2.4. CaoFile::Execute(“GetBodyFileName”) コマンド

指定された番号のメール Body のファイル名を取得します.

指定された Body がメール本文の場合は空文字をファイル名として返します.

ファイル名が MIME, もしくは RFC2231 方式でエンコードされている場合は, デコードしたファイル名を返します.

番号は, {0, 1, …, (メールの Body 数 – 1)}までの範囲で指定してください.

書式 GetBodyFileName (< No >)

< No > : [in] メール Body の番号.（VT_I4）

戻り値 : [out] メール Body のファイル名.（VT_BSTR）
メール本文の場合は空文字, 添付ファイルの場合はファイル名が返ります.

使用例

```
result = m_caoFile.Execute("GetBodyFileName", 0)
```

3.2.5. CaoFile::Execute(“GetBody”) コマンド

指定された番号のメール Body をバイト配列で取得します.
番号は, {0, 1, …, (メールの Body 数 – 1)}までの範囲で指定してください.

書式　GetBody (< No >)

< No > : [in] メール Body の番号.（VT_I4）
戻り値 : [out] メール Body の本文のバイト配列.（VT_ARRAY）

使用例

```
result = m_caoFile.Execute("GetBody", 0)
```

3.2.6. CaoFile::Execute(“GetCodePage”) コマンド

指定された番号のメール Body をエンコードしているコードページを取得します.
エンコーディングが Content-Type に記載されていない場合は, 0 が返ります.
番号は, {0, 1, …, (メールの Body 数 – 1)}までの範囲で指定してください.

書式　GetCodePage (< No >)

< No > : [in] メール Body の番号.（VT_I4）
戻り値 : [out] エンコードしているコードページ.（VT_I4）

使用例

```
result = m_caoFile.Execute("GetCodePage", 0)
```

3.2.7. CaoFile::Execute(“GetLowMessage”) コマンド

解析前の生のメールメッセージをバイト配列で取得します.

書式　GetLowMessage ()

戻り値 : [out] メールメッセージ.（VT_ARRAY）

使用例

```
result = m_caoFile.Execute("GetLowMessage")
```

4. サンプルプログラム

POP3 プロバイダと Visual C# を使った簡単なサンプルプログラムを紹介します. サンプルプログラムの全容は下記のフォルダにありますので参考にしてください.

List 4-1 Pop3SampleForm.cs

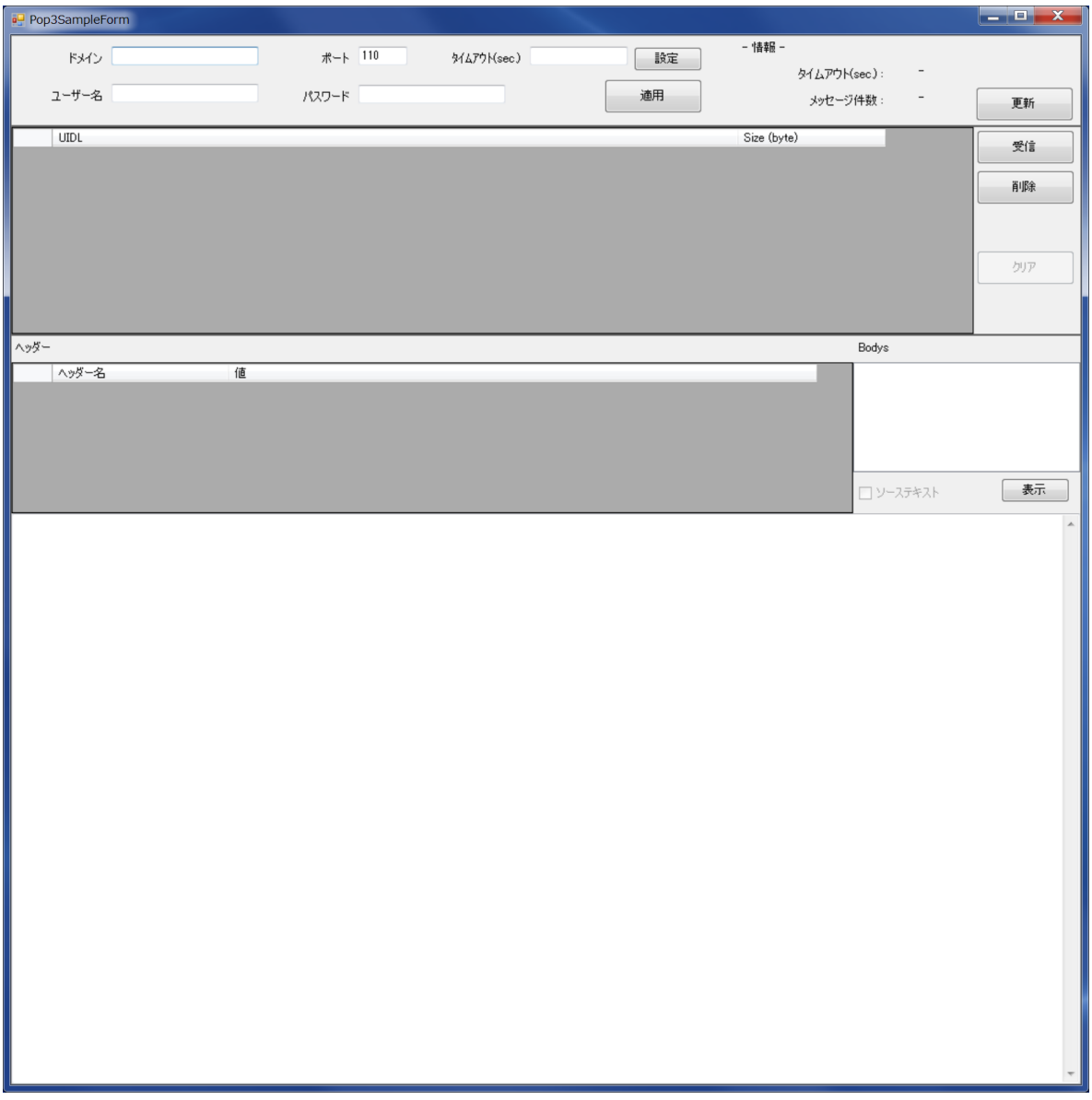

図 4-1CAO Pop3 サンプル

サンプルプログラム

‘ AddController

```
_ctrl = cCaoEngine.Workspaces[0].AddController("POP3", "CaoProv.POP3", string.Empty, optStr);

‘MessagesCount
int count = (int)_ctrl.Execute("MessageCount", null);

‘ ListUIDL
e.Result = _ctrl.Execute("ListUIDL", null);

‘ Size
int size = (int)_ctrl.Execute("Size", id);

‘ DeleteMessage
_ctrl.Execute("DeleteMessage", uidl);

‘ SetTimeout
_ctrl.Execute("SetTimeout", timeout);

‘ GetTimeout
int msec = (int)_ctrl.Execute("GetTimeout", null);

‘ ProviderCancel
_ctrl.Execute("ProviderCancel", null);

‘ ProviderClear
_ctrl.Execute("ProviderClear", null);

‘ AddFile
CCaoFile file = _ctrl.AddFile("mail", string.Format("UIDL={0}", uidl));

‘ Value
string[] texts = _file.Value as string[];

‘ GetHeaderNames
string[] headerNames = file.Execute("GetHeaderNames", null) as string[];

‘ GetHeader
object value = file.Execute("GetHeader", name);
```

```
‘ GetBodyCount
int count = (int)file.Execute("GetBodyCount", null);

‘ GetBodyFileName
string fileName = file.Execute("GetBodyFileName", i).ToString();

‘ GetCodePage
int codePage = (int)_file.Execute("GetCodePage", data.Num);

‘ GetLowMessage
byte[] mess = _file.Execute("GetLowMessage", null) as byte[];
```