DataQueue プロバイダ

汎用データキュー

Version 1.0.5

ユーザーズ ガイド

September 3, 2021

【備考】

【改版履歴】

バージョン	日付	内容
1.0.0	2017-10-08	初版.
1.0.1	2017-12-08	ファイルバックアップ/リストア対応
1.0.2	2018-01-08	手動バックアップ/リストア機能削除, QFile パラメータ必須化
1.0.3	2018-01-15	ファイル退避を行わないモードを追加.QDir オプション追加.
1.0.4	2018-12-27	QFile オプション指定時の@QSNAP の制限事項追記.
1.0.5	2020-02-27	メモリ関連の処理修正.
	2020-06-30	CAO オブジェクトとキューの関係図を追記. AddExtension の説明修正.
	2021-05-24	AutoBackup ビットに注意事項を追記.
	2021-07-26	変数名を大文字に修正
	2021-09-03	Execute コマンドの見出し番号を修正

目次

1. はじめに

本書は，待ち行列的な処理を実現するための機能を提供する DataQueue プロバイダのユーザガイドです.

DataQueue プロバイダは内部に複数のキューを管理することが可能です. 各キューはデータを先入れ先出し（FIFO）のリスト構造で保持し，必要に応じてデータを順に取り出す機能を提供します.

本書は，この DataQueue プロバイダの機能と実装されている変数やメソッドについて説明します.

2. プロバイダの概要

2.1. 概要

DataQueue プロバイダで提供される待ち行列処理はキュー(Queue)構造によって実現されています.

このキューは ORiN2 の CAO(Controller Access Object)の CaoExtension クラスによって管理され, キューに対する操作は CaoExtension クラスで実装されている変数またメソッドによって行うことが可能です.

DataQueue プロバイダでは, CaoController のオブジェクトを介して複数の CaoExtension オブジェクトを生成することで同時に複数のキューを管理することができます. 一つの CaoExtension オブジェクトは一つのキューに対応しています.

以下に CAO オブジェクトとキューとの関係と, DataQueue プロバイダによって提供されるキューに対する操作の概要を示します.

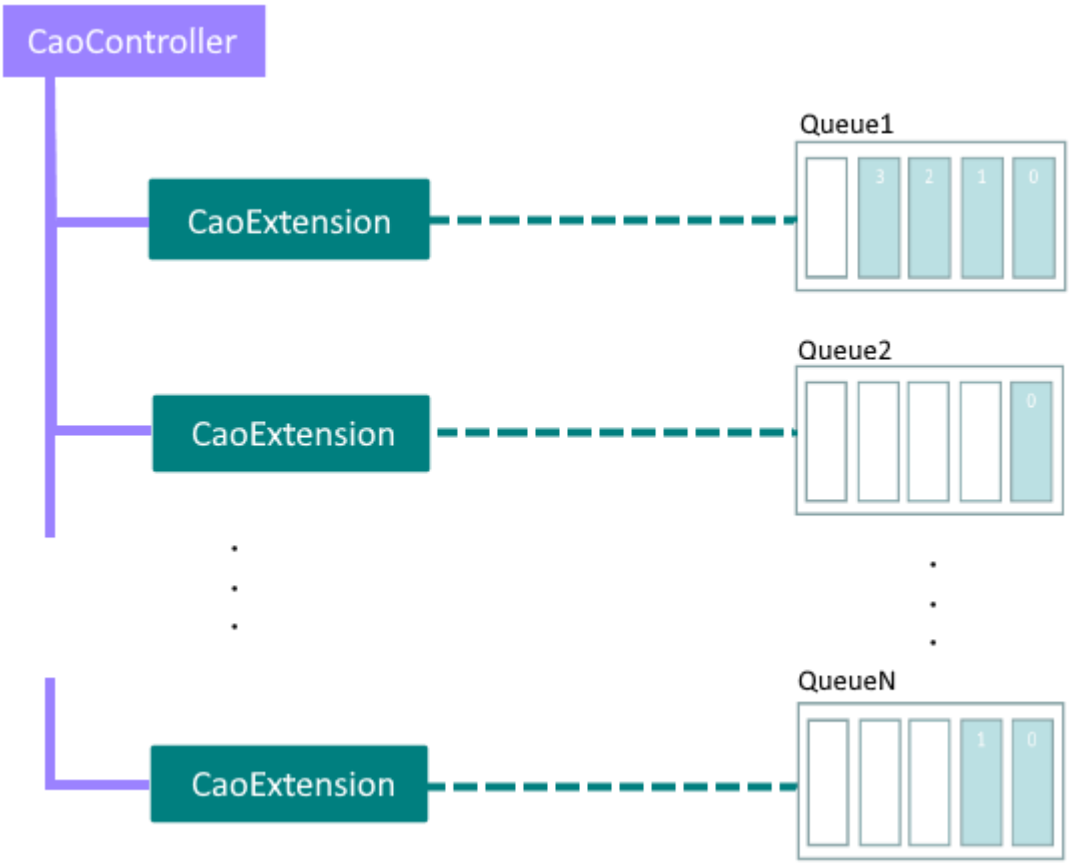


図 2-1 CAO オブジェクトとキューとの関係

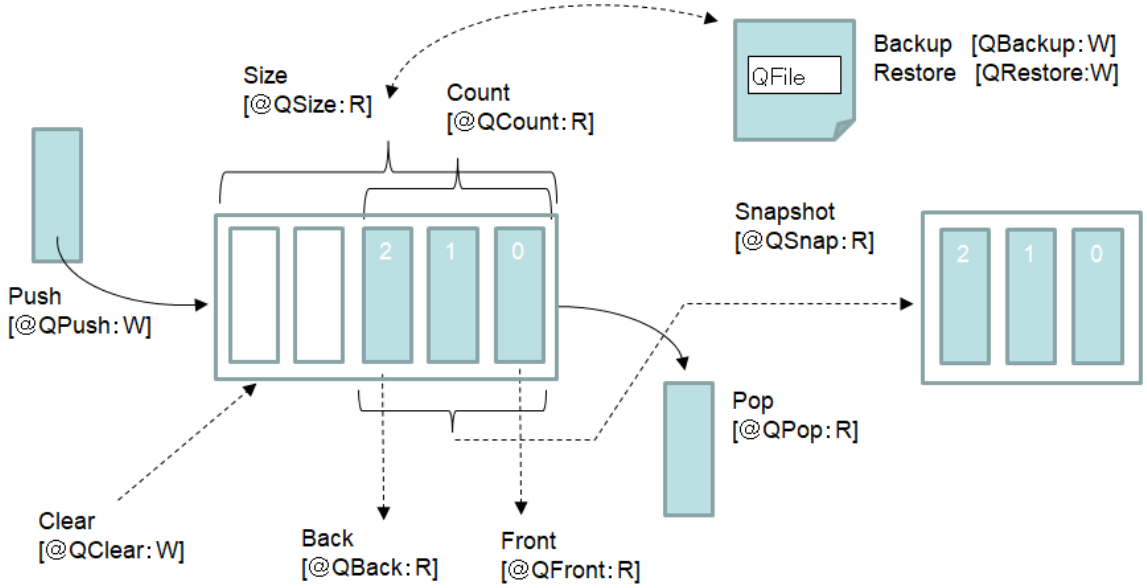


図 2-2 キューに対する操作概要

2.2. インストール

DataQueue プロバイダのモジュールは，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 DataQueue プロバイダ

ファイル名	CaoProvDataQueue.dll
ProgID	CaoProv.DataQueue
レジストリ登録[1]	regsvr32 CaoProvDataQueue.dll
レジストリ登録の抹消	regsvr32 /u CaoProvDataQueue.dll

2.3. メソッド・プロパティ

2.3.1. CaoWorkspace::AddController メソッド

DataQueue プロバイダでは AddController 時に仮想のコントローラ名を指定します.
以下に AddController の引数仕様を示します.

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

< bstrCtrlName > : [in] コントローラ名
< bstrProvName > : [in] プロバイダ名. 固定値 ="CaoProv.DataQueue"
<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(未使用)

以下に AddController メソッドを実行するときの例を示します.

```
AddController
(
    "DQ1",                                  // 仮想コントローラ名 = DQ1
    "CaoProv.DataQueue",
    ""
    ""'
);
```

[1] ORiN SDK でインストールした場合は手動で登録／抹消する必要はありません.

2.3.2. CaoController::AddExtension メソッド

DataQueue プロバイダでは各キューを CaoExtension オブジェクトで実現しています.
キューを作成するためには CaoController オブジェクトに対して AddExtension を実行します.
以下に AddExtension の引数仕様を示します.

書式　AddExtension (<bstrName:BSTR>, [<bstrOption:BSTR>])

< bstrCtrlName > : [in] キュー名
任意の名前を指定します.

<bstrOption> : [in] オプション文字列
以下のオプションをコンマ区切りで指定できます.

QSize : キューに格納可能な最大データ数を指定します.（省略可）
値は整数値で範囲が 0 から 2147483647 が指定できます.
省略時はデフォルト値として 0 が使用されます.
[QSize が 0 の時の最大データ数]

QFile オプション	最大サイズ
指定あり	2147483647
指定なし	1000

例）
QSize=100

QMode : キューの操作に対応した振る舞いを設定します.（省略可）
省略時はデフォルト値として 0 が使用されます.
ビット単位で下記値を指定します.
QMode =
[AutoBackup:4bit]
[SwapData:3bit]
[FullData：2bit]
[NoEmpty:1bit]
[NoOverFlow：0bit]

詳細に関しては「2.3.2.1 QMode オプション」を参照ください.
例）
QMode=1 // Push 時エラーなし
QMode=8 // Swap あり

QDir : キューのファイル保存を行う際に使用するディレクトリを指定します.
省略時は環境変数の DATAQUEUE_PROVIDER_DIR の値が使用されます.

QFile : キューのファイル保存を行う際に使用するファイル名のプレフィックスを指定します.
¥で区切ることによってサブフォルダの指定も可能です.
AutoBackup の実行時および, キューのサイズが大きくなったときにデータをファイルに退避させるときにこのパラメータが使われます.
QMode で AutoBackup を指定した場合は必須です.
指定しなかった場合,データのファイルへの退避は行われません.
QFile を指定し,QDir および環境変数 DATAQUEUE_PROVIDER_DIR が設定されていない場合はエラーになります.
例）
QDir=C:¥Temp,QFile=Queue と指定した場合, C:¥Temp 下に Queue.fro, Queue.bak, Queue.1 などのファイルが生成されます.複数の DataQueue プロバイダを使用する際は 2.3.2.1.5 の注意事項を確認ください.
制約事項:
QFile を指定するとファイル保存が行

われる動作となるため, @QSNAP および GetQueueSnapshot で取得する個数は最大で 500 個となります.

以下に AddExtension メソッドを実行するときの例を示します.

```
AddExtension
(
    // キュー名 = Q1
    "Q1",
    // Push 時エラーなし, 退避ディレクトリ"C:¥Temp", 退避ファイルプレフィックス"Queue"
    "QMode=1,QDir=C:¥Temp,QFile=Queue"
);
```

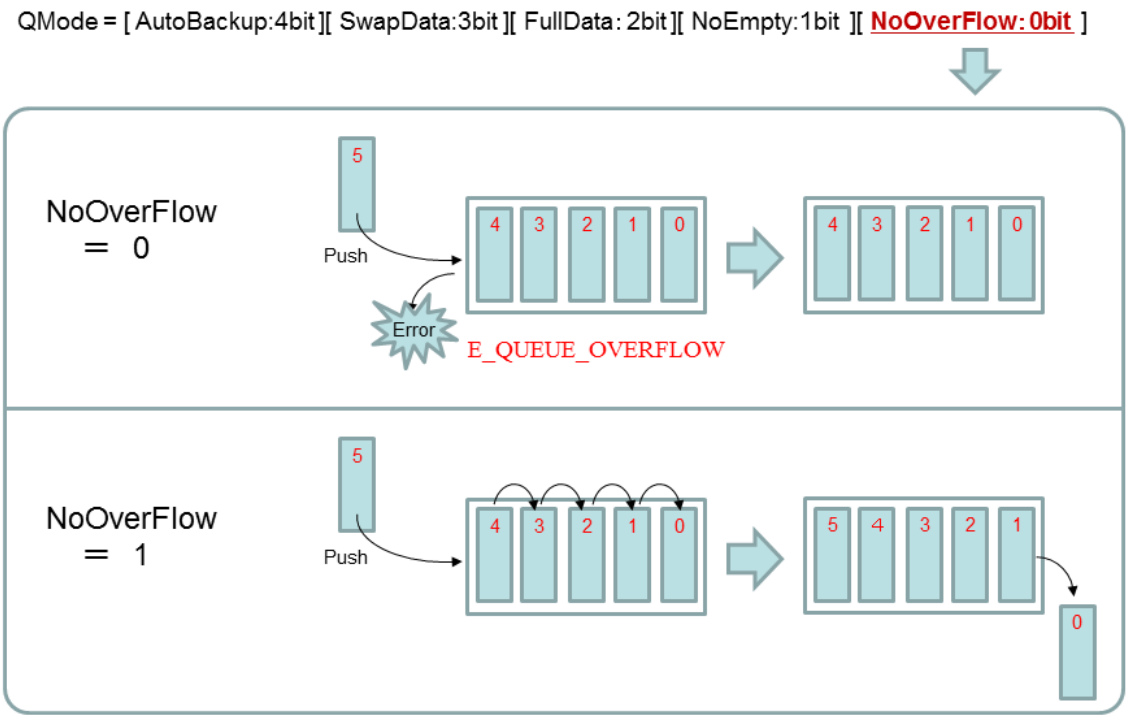
2.3.2.1. QMode オプション

AddExtension メソッドの QMode オプションに対してビット単位で値指定することにより,作成されるキューの振る舞いを設定することが可能です.

以下に QMode オプションのビット単位での指定方法を示します.

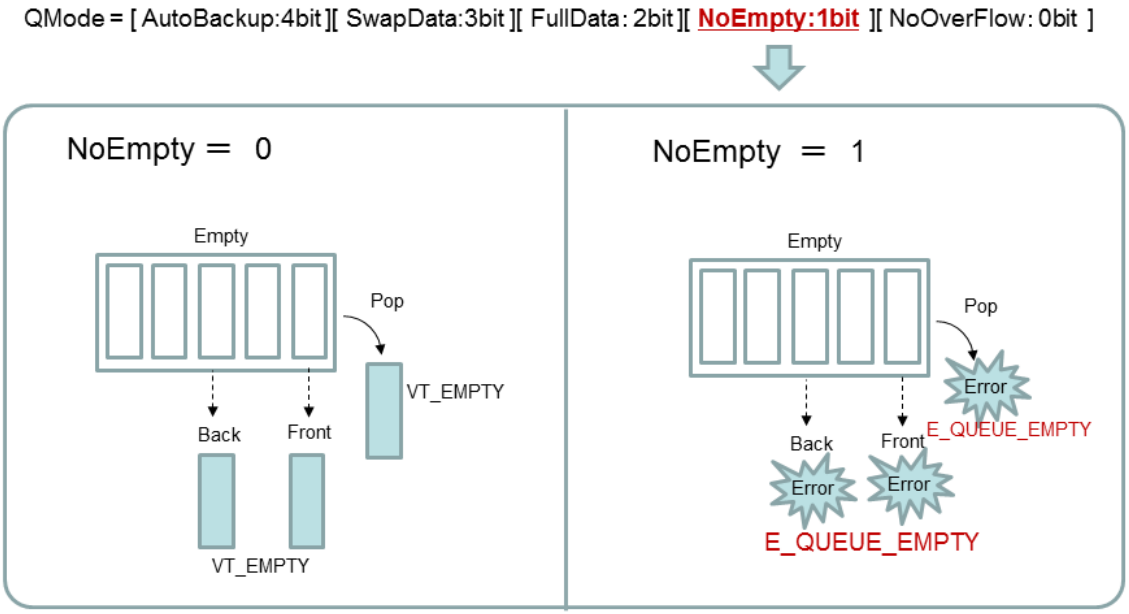
2.3.2.1.1. NoOverFlow ビット

NoOverFlow ビットの指定は, キューにデータを追加する際にいっぱいの場合エラーとするかの指定です. エラーとしない場合は追加の際に前方データが破棄され, 後方に指定データが追加されます.


QMode = [AutoBackup:4bit][SwapData:3bit][FullData : 2bit][NoEmpty:1bit][NoOverFlow : 0bit]
NoOverFlow
＝ 0
5
Push
4 3 2 1 0
Error
E_QUEUE_OVERFLOW
4 3 2 1 0
NoOverFlow
＝ 1
5
Push
4 3 2 1 0
5 4 3 2 1
0

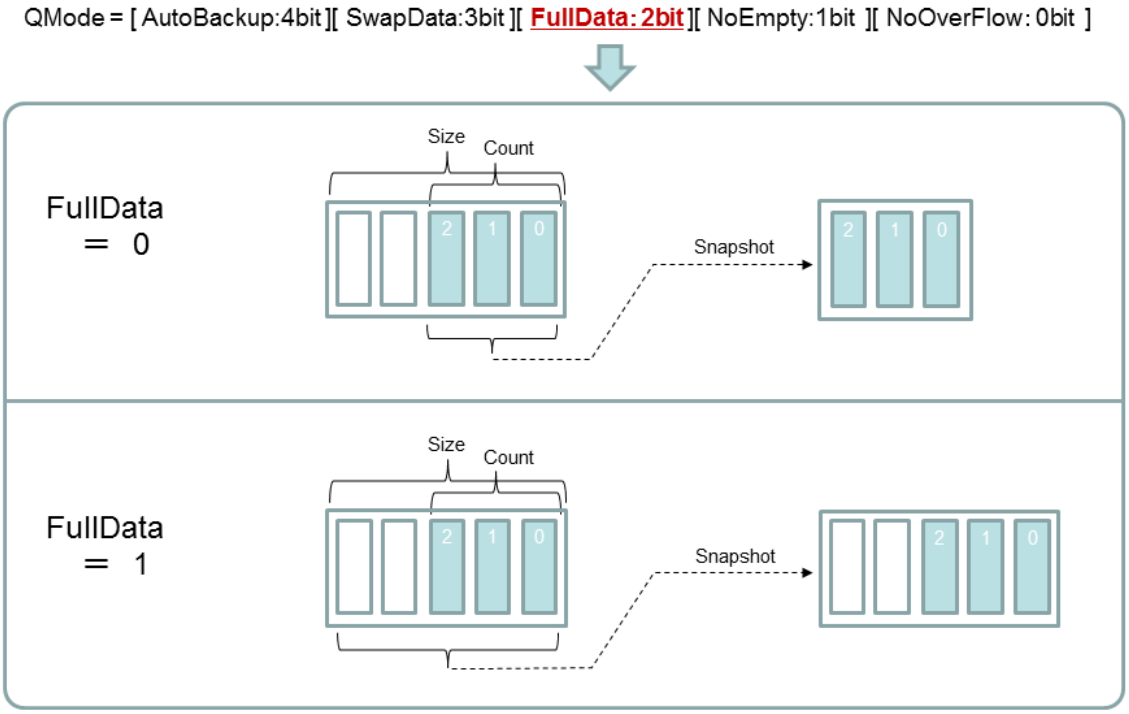
2.3.2.1.2. NoEmpty ビット

NoEmpty ビットの指定は，キューが空の場合エラーとするかの指定です．



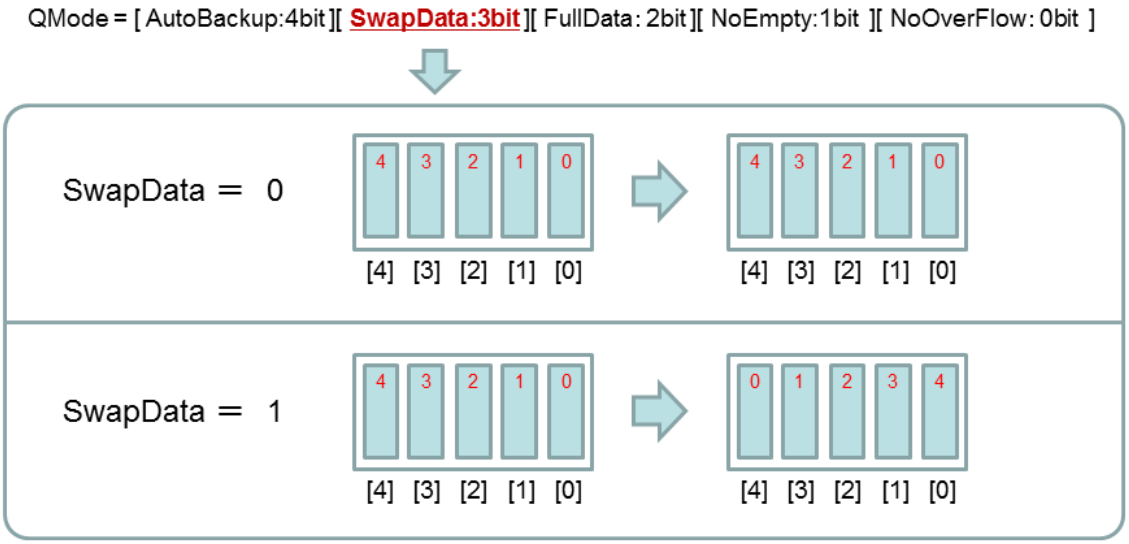
2.3.2.1.3. FullData ビット

FullData ビットの指定は，キューに格納されている全データを一括取得（Snapshot 操作）する際に空き領域も含めるかの指定です.



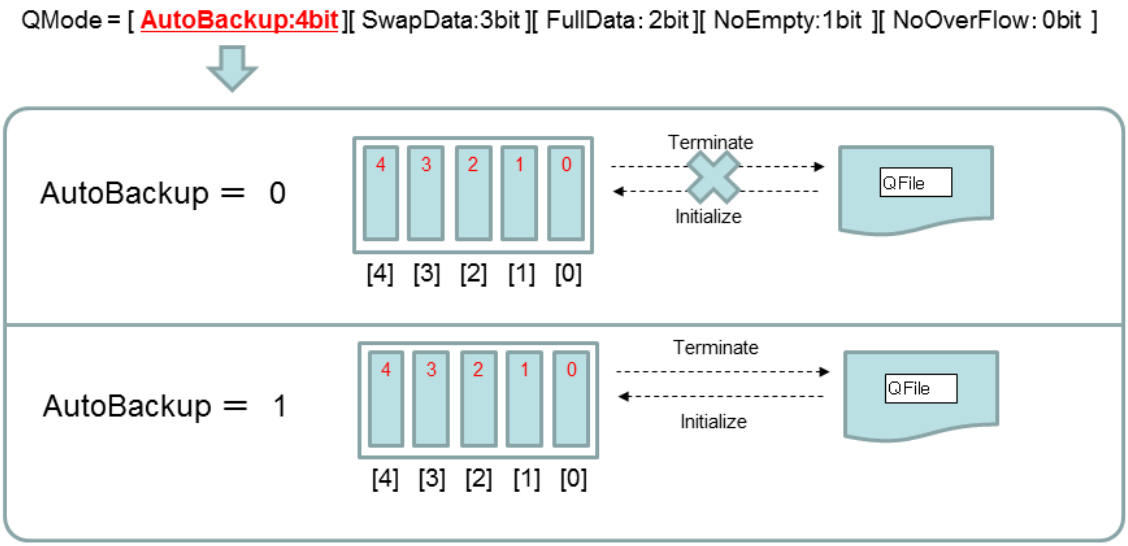
2.3.2.1.4. SwapData ビット

SwapData ビットの指定は, キューに格納されている全データを一括取得(Snapshot 操作)する際に返されるデータ列の並びを逆転させるかの指定です.



2.3.2.1.5. AutoBackup ビット

AutoBackup ビットの指定は,Extension の初期化と解放時にデータのバックアップ/レストアを実行するかの指定です.



注意事項

複数の DataQueue プロバイダを同時に使い, バックアップファイル指定(QDir, QFile)が同一のものを指している場合, 最後に保存したものに上書きされます. レストア時には DataQueue プロバイダ一つ分のキューしかレストアされません.

CaoExtension::AddVariable メソッド

CaoExtension オブジェクトでは管理しているキューに対して, 変数の読み書きにより, キューに対する操作が行えます.

以下に AddVariable の引数仕様を示します.

書式 AddVariable (<bstrName:BSTR>, [<bstrOption:BSTR>])

< bstrCtrlName > : [in] 変数名

以下のシステム変数名或いはユーザ変数名を指定できます.

システム変数 : '@'で始まる以下の予約文字列を指定します.

@QMODE
@QSIZE
@QCOUNT
@QCLEAR
@QSNAP
@QPUSH
@QPOP
@QBACK
@QFRONT

ユーザ変数 : '@'で始まらない任意の文字列を指定します. この場合は, オプション文字列の Type を指定することが必須です.

<bstrOption> : [in] オプション文字列

以下のオプションをコンマ区切りで指定できます.

Type : ユーザ変数の場合, 振る舞いを決めるためにシステム変数名を指定します.

@QMODE
@QSIZE,
@QCOUNT
@QCLEAR
@QSNAP
@QPUSH
@QPOP
@QBACK

@QFRONT
このオプションはシステム変数に対して効果がありません.
例）
Type=@QCOUNT

以下に AddVariable メソッドを実行するときの例を示します.

```
AddVaraible
(
     "@QCOUNT",                    // システム変数
     ""                            //
);

AddVaraible
(
     "MyQueueCount",               // ユーザ変数
     "Type=@QCOUNT"                // システム変数の@QCOUNT
);
```

DataQueue プロバイダで実装されているシステム変数は 2.4.1 を参照して下さい.

2.3.3. CaoExtension::Execute メソッド

CaoExtension オブジェクトのコマンドを実行します.

以下に CaoExtension::Execute メソッドのコマンド一覧を示します.

表 2-2 CaoExtension::Execute メソッドのコマンド実装一覧

コマンド	機能	対応変数
GetQueueMode	キューの動作モードを取得します.	@QMODE
SetQueueMode	キューの動作モードを設定します.	@QMODE
GetQueueSize	キューに格納可能な最大データ数を取得します.	@QSIZE
SetQueueSize	キューに格納可能な最大データ数を設定します.	@QSIZE
GetQueueCount	キューに格納されているデータ数を取得します.	@QCOUNT
ClearQueue	キューを空にします.	@QCLEAR
GetQueueSnapshot	キューに格納されているデータを一括取得します.	@QSNAP
PushQueue	キューの待ち行列の最後にデータを追加します.	@QPUSH
PopQueue	キューの待ち行列にある前方データを取得します.	@QPOP
GetQueueBack	キューの待ち行列にある後方データを参照します.	@QBACK
GetQueueFront	キューの待ち行列にある前方データを参照します.	@QFRONT

2.3.3.1. CaoExtension::Execute("GetQueueMode") コマンド

キューの動作モードを取得します.

書式　GetQueueMode ()

戻り値　：　[out]　動作モード(VT_I4)
動作モードの値に関しては「2.3.2.1 QMode オプション」を参照ください.

2.3.3.2. CaoExtension::Execute("SetQueueMode") コマンド

キューの動作モードを設定します.

書式 SetQueueMode ([<Mode>])

< Mode > : [in] 動作モード(VT_I4)
値の範囲: 任意の整数値
指定された値は有効ビット分だけマスクされた値に変換されます.
動作モードの値に関しては「2.3.2.1 QMode オプション」を参照ください.

戻り値 : [out] なし

2.3.3.3. CaoExtension::Execute("GetQueueSize") コマンド

キューに格納可能な最大データ数を取得します.

書式 GetQueueSize ()

戻り値 : [out] 格納可能な最大データ数(VT_I4)

2.3.3.4. CaoExtension::Execute("SetQueueSize") コマンド

キューに格納可能な最大データ数を設定します.

書式 SetQueueSize ([<Size>])

< Size > : [in] 格納可能な最大データ数(VT_I4)
値の範囲: 1 から 2147483647
既存に格納しているデータ数が指定の格納可能な最大数以上の場合, 前方データが破棄されます.

戻り値 : [out] なし

2.3.3.5. CaoExtension::Execute("GetQueueCount") コマンド

キューに格納されているデータ数を取得します.

書式　GetQueueCount ()

戻り値　：[out]　データ数(VT_I4)

2.3.3.6. CaoExtension::Execute("ClearQueue") コマンド

キューを空にします.

書式　ClearQueue ()

戻り値　：[out] なし

2.3.3.7. CaoExtension::Execute("GetQueueSnapshot") コマンド

キューに格納されているデータを一括取得します.
取得できるデータはメモリ上に展開されているデータのみになります.

書式　GetQueueSnapshot ()

戻り値　：[out] メモリ上に展開されているデータ
(VT_VARIANT|VT_ARRAY)

2.3.3.8. CaoExtension::Execute("PushQueue") コマンド

キューの待ち行列の最後にデータを追加します.

書式　PushQueue ([<Data>])

< Data >　：[in] 追加データ(VT_VARIANT)
QMode 動作モードの NoOverFlow ビットが 1 で
格納可能な最大数の場合，前方データが破棄されて，最後尾にデータが追加されます.

戻り値　：[out] なし

2.3.3.9. CaoExtension::Execute("PopQueue") コマンド

キューの待ち行列にある前方データを取得します.

書式　PopQueue ()

戻り値　：　[out]　前方データ(VT_VARIANT)

2.3.3.10. CaoExtension::Execute("GetQueueBack") コマンド

キューの待ち行列にある後方データを参照します

書式　GetQueueBack ()

戻り値　：　[out]　後方データ(VT_VARIANT)

2.3.3.11. CaoExtension::Execute("GetQueueFront") コマンド

キューの待ち行列にある前方データを参照します

書式　GetQueueFront ()

戻り値　：　[out]　前方データ(VT_VARIANT)

2.3.4. CaoExtension::get_VariableNames プロパティ

get_VariableNames プロパティでは, AddVariable で指定可能はシステム変数の一覧を文字列型の配列を格納した VARIANT 型で取得します.

2.4. 変数一覧

2.4.1. CaoExtension クラス

DataQueue プロバイダでは，システム変数のみが予約されています．ユーザ変数には，任意の名前を使用することができます．

表 2-3 CaoExtension クラス システム変数一覧

変数名	データ型	説明	属性	
			get	put
@QMODE	VT_I4	キューの動作モードを取得/設定します． 値の範囲：任意の整数値 指定された値は有効ビット分だけマスクされた値に変換されます．	○	○
@QSIZE	VT_I4	キューに格納可能な最大データ数を取得/設定します． 値の範囲：1 から 2147483647 設定する場合で既存に格納しているデータ数が指定の格納可能な最大数以上の場合，前方データが破棄されます．	○	○
@QCOUNT	VT_I4	キューに格納されているデータ数を取得します．	○	-
@QCLEAR	なし	キューを空にします．	-	○
@QSNAP	VT_VARIANT| VT_ARRAY	メモリ上に保持されている全データを一括取得します．ただし，QFile オプション指定時は最大 500 個． データがない場合は VT_EMPTY が返ります．	○	-
@QPUSH	VT_VARIANT	キューの待ち行列の最後にデータを追加します． QMode 動作モードの NoOverFlow ビットが 1 で格納可能な最大数の場合，前方データが破棄されて，最後尾にデータが追加されます．	-	○
@QPOP	VT_VARIANT	キューの待ち行列にある前方データを取得します． データがない場合は VT_EMPTY が返ります．	○	-
@QBACK	VT_VARIANT	キューの待ち行列にある後方データを参照します． データがない場合は VT_EMPTY が返ります．	○	-
@QFRONT	VT_VARIANT	キューの待ち行列にある前方データを参照します． データがない場合は VT_EMPTY が返ります．	○	-

2.5. エラーコード

DataQueue プロバイダでは，以下の固有エラーコードが定義されています．ORiN2 共通エラーについては，「ORiN2 プログラミングガイド」のエラーコードの章を参照してください．

表 2-4 独自エラーコード一覧

エラー名	エラー番号	説明
E_FAILED_CREATE	0x80100000	生成に失敗しました．
E_QUEUE_OVERFLOW	0x80100001	PUSH 時 OVERFLOW 発生しました．
E_QUEUE_EMPTY	0x80100002	POP，SNAPSHOT，BACK，FRONT のデータ取得時時，キューが空（データなし）のエラー発生しました．

3. サンプルプログラム

以下にキュー(Q1)を作成して，それに対して値を追加，取出しするサンプルを示します．

List 3-1 SampleApp1.frm

```
Private Eng As CaoEngine
Private Ctrl As CaoController
Private Ext As CaoExtension
Private VarPush As CaoVariable
Private VarPop As CaoVariable

' フォームのロード
Private Sub Form_Load()
    Set Eng = New CaoEngine

    '変数オブジェクトを取得します.
    Set Ctrl = Eng.Workspaces(0).AddController("DQ", "CaoProv.DataQueue")
    Set Ext = Ctrl.AddExtension("Q1","QMode=1,QDir=C:¥Temp,QFile=Queue")

    Set VarPush = Ext.AddVariable("@QPUSH")
    Set VarPop = Ext.AddVariable("@QPOP")

End Sub

'キューに値を PUSH
Private Sub Command1_Click ()
    VarPush = Text1.Text
End Sub

'キューから値を Pop
Private Sub Command2_Click ()
    Text2.Text = VarPop
End Sub
```