因幡電機産業株式会社
チョコ停ウォッチャーmini プロバイダ
ユーザーズ ガイド

Version 1.0.1

June 28, 2022

備考：

【改版履歴】

バージョン	日付	内容
1.0.0	2020-11-30	初版.
1.0.1	2022-06-28	GetErrorLog コマンドの不具合を修正. GetMovie コマンドの完了メッセージの不具合を修正. @ERROR_CODE 変数の内部処理改善. モード切換え直後の@IMAGE 変数の不具合を修正.

【対応機種】

機種	バージョン	注意事項
IB-MCT001	--	

【動作確認機種】

機種	バージョン	注意事項
IB-MCT001	72n	

目次

1. はじめに

本書は，因幡電機産業株式会社のチョコ停ウォッチャーmini(Model： IB-MCT001)に接続するプロバイダのユーザーズガイドです．図 1-1 が本プロバイダとデバイスの全体構成図になります．以降本プロバイダを単にプロバイダと呼称します．プロバイダとデバイスは図で示すように HTTP で接続されます．

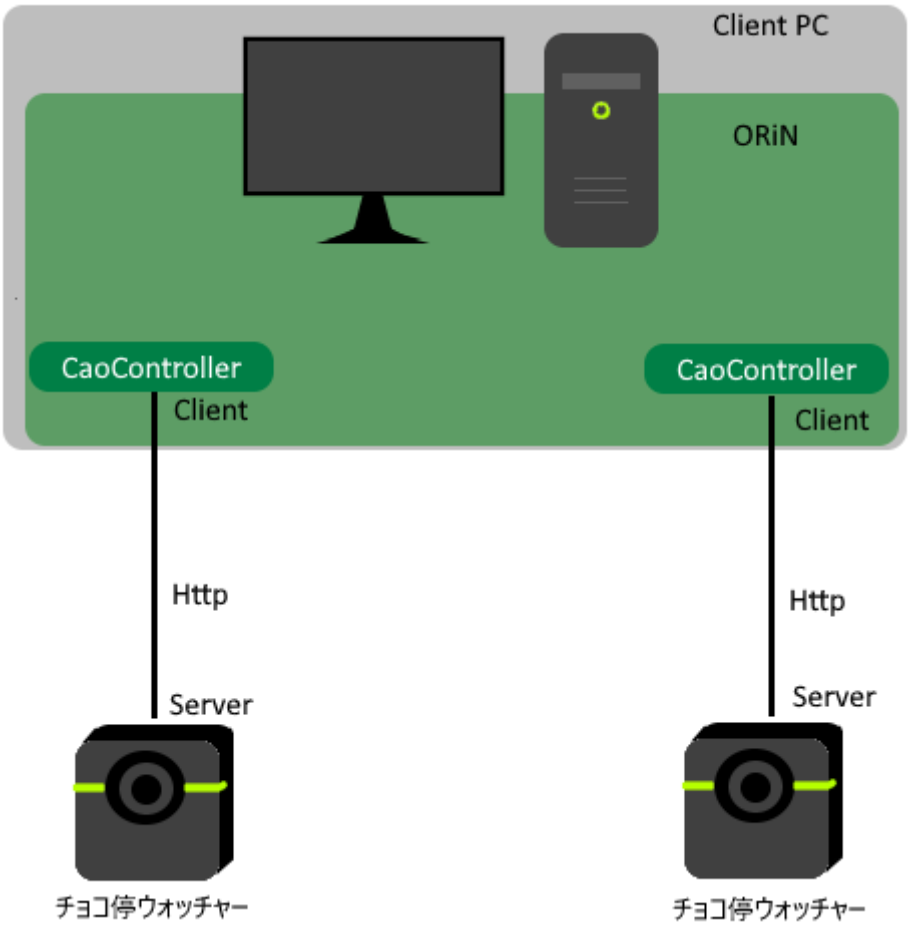


図 1-1 構成図

また，本プロバイダ及びデバイスそれぞれの対応を図 1-2に表します．
（※一例です．全てを表しているわけではありません．）

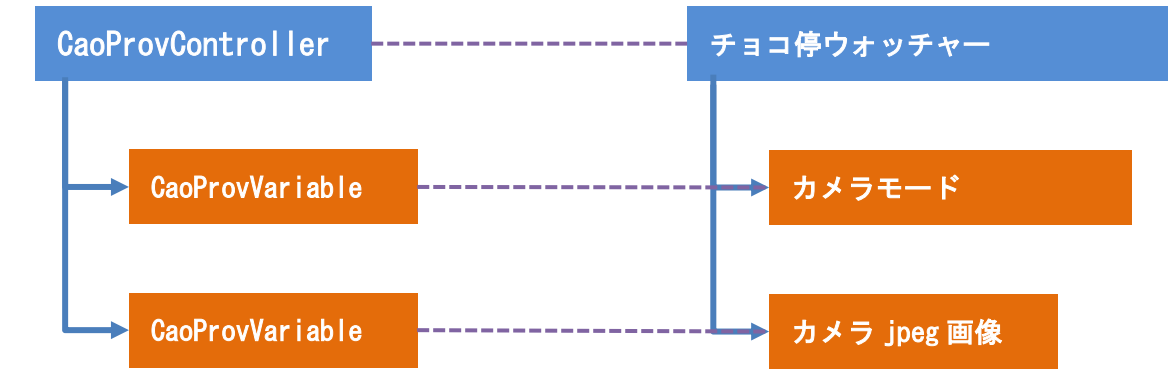


図 1-2 プロバイダの構成とデバイス情報との対応図

2. アプリケーション開発のための環境セットアップ

2.1. デバイスとクライアント PC との接続

クライアント PC とデバイスを接続するための手順を示します.

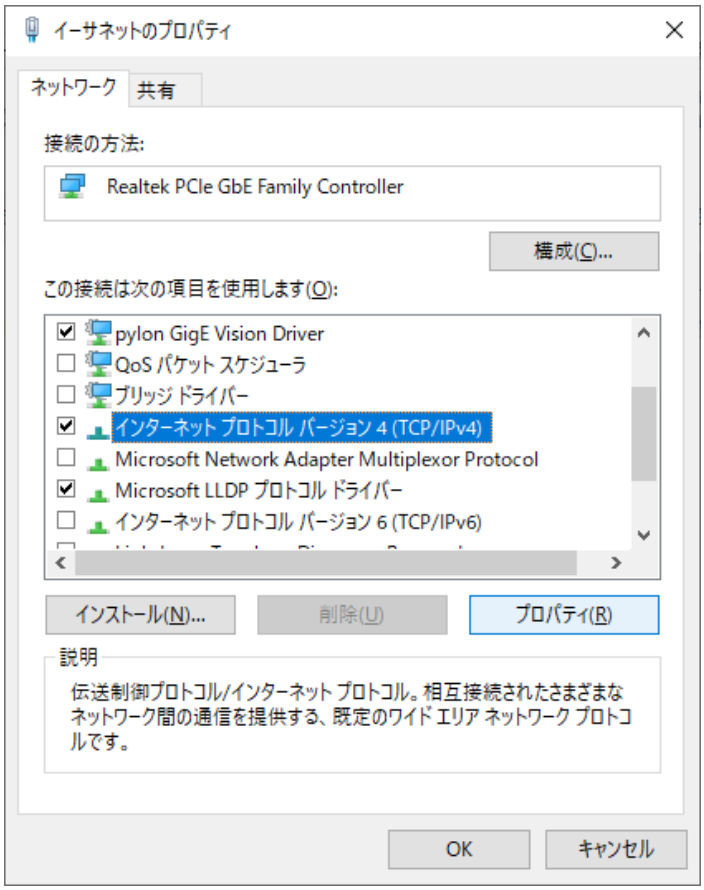
2.1.1. PC のネットワーク設定

クライアント PC とデバイスは固定 IP アドレスで接続します.

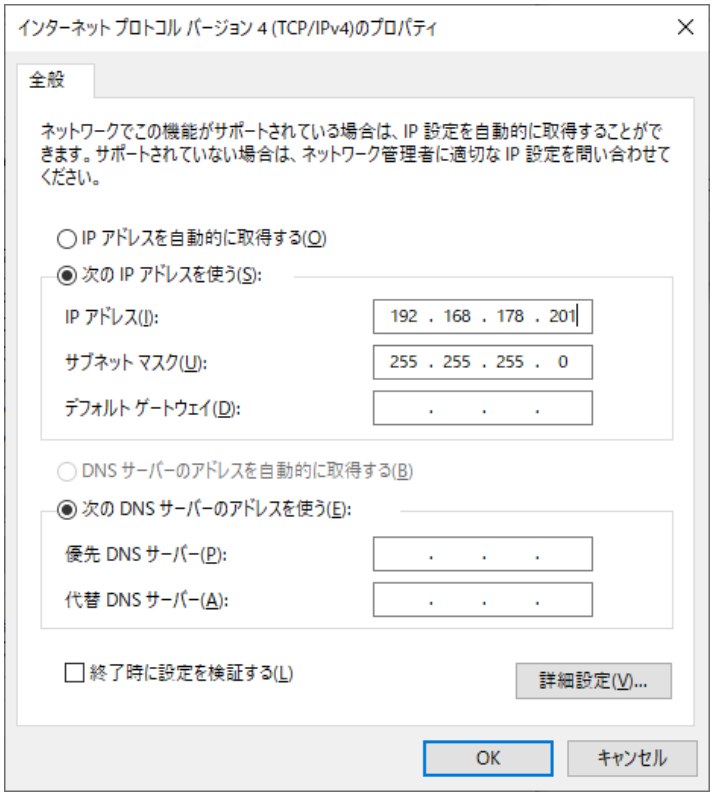
① クライアント PC を起動し,「スタート」→「設定」→「ネットワークとインターネット」→「イーサネット」→「アダプターのオプションを変更する」メニューを実行し,「ネットワーク接続設定」ウィンドウを表示します.

② デバイスと接続する有線 LAN アダプタの「プロパティ」ウィンドウを起動します.

③ プロパティウィンドウ上の「インターネットプロトコルバージョン 4(TCP/IPv4)」項目を選択した状態で「プロパティ(R)」ボタンを押下します.



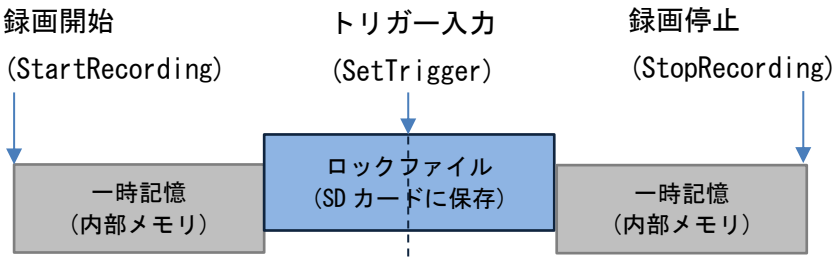
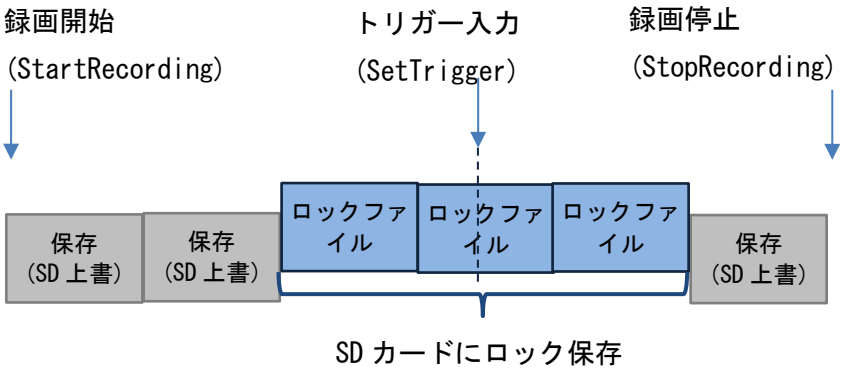
④ IP アドレスとサブネットマスクにデバイスと通信可能な値を入力します. 画像は, デバイス側の IP アドレスが 192.168.178.178, サブネットマスクが 255.255.255.0 の時に, クライアント PC に設定した IP アドレスとサブネットマスクの例です.


インターネット プロトコル バージョン 4 (TCP/IPv4)のプロパティ
全般
ネットワークでこの機能がサポートされている場合は、IP 設定を自動的に取得することができます。サポートされていない場合は、ネットワーク管理者に適切な IP 設定を問い合わせてください。
IP アドレスを自動的に取得する(O)
次の IP アドレスを使う(S):
IP アドレス(I):
192 . 168 . 178 . 201
サブネット マスク(U):
255 . 255 . 255 . 0
デフォルト ゲートウェイ(D):
DNS サーバーのアドレスを自動的に取得する(B)
次の DNS サーバーのアドレスを使う(E):
優先 DNS サーバー(P):
代替 DNS サーバー(A):
終了時に設定を検証する(L)
詳細設定(V)...
OK
キャンセル

2.1.2. デバイスの録画モード設定

デバイスの録画モードにはトリガーオンリーモード(DRAM モード)とループアンドトリガーロックモード(SD モード)の 2 種類の録画モードが存在します．用途に合わせて録画モードを設定してください．

録画モード

トリガーオンリーモード	動画をメモリに一時記憶する録画モードです． トリガー入力を受付けると，トリガータイミングを中心とした動画ファイルを SD カードに保存します． SD カードへのアクセスはトリガー入力のみの保存動作としているため， SD カードの長寿命化を図れます． 録画単位は 20s，40s から選択できます． 下図はトリガーオンリーモードの際の動作イメージと各動作を実行するためのプロバイダの拡張コマンドとの対応を示しています． 
ループアンドトリガーロックモード	動画は録画単位ごとに SD カードに繰り返し保存されます． トリガー入力を受付けると，トリガータイミングの動画とその前後の合計 3 動画を 3 ファイル分上書き禁止(ロック)して保存します． 録画単位は 1m，3m，5m，10m から選択できます． 下図はループトリガーロックモードの際の動作イメージと各動作を実行するためのプロバイダの拡張コマンドとの対応を示しています． 

トリガー入力を受付けて保存されたファイルは今後ロックファイルと呼称します．

2.1.3. プロバイダのインストール

ORiN2 をインストールすればプロバイダを使用する準備が整います.

3. コマンドリファレンス

3.1. メソッド/プロパティー覧

表 3-1 メソッド/プロパティー覧

カテゴリ	メソッド/プロパティ[1]		機能	参照
CaoWorkspace				
	AddController	M	コントローラに接続	P.13
CaoController				
	GetVariableNames	M	接続可能な変数名リストの取得	P.14
	Variables	P	コントローラが保持する変数コレクションの取得	P.14
	AddVariable	M	変数オブジェクトの追加	P.15
	Execute	M	拡張コマンドの実行	P.15
	OnMessage	E	メッセージ受信イベント	P.15
CaoVariable				
	Value	P	値の取得/設定	P.15

3.2. メソッド・プロパティ

3.2.1. CaoWorkspace クラス

3.2.1.1. AddController メソッド

CaoWorkspace にコントローラオブジェクトを追加します. プロバイダは本メソッド実行時に渡されたパラメータを利用してデバイスと接続します. また, 接続確認のために@MODE 変数の値取得処理を実行します.

書式

```
AddController
(
        "<コントローラ名>",             // コントローラ名(任意)
        "CaoProv.inaba.IB-MCT001",      // プロバイダ名 (固定)
        "<マシン名>",                   // プロバイダ実行マシン名(未使用)
        "<オプション>"                  // オプション文字列
)
```

オプション

[1] M:メソッド, P:プロパティ, E:イベントをそれぞれ示します.

以下にオプション文字列に指定するオプションを示します．オプション文字列は下記に示す各オプションをカンマ(,)でつなげた文字列となります．

オプション	必須	説明	値範囲	デフォルト値
Url	○	接続先の URL を指定します． 通常以下となります． http://"デバイス IP アドレス"	-	-
Timeout	--	通信タイムアウト(ms)を指定します．	1 -	2000
BasePath	--	動画取得の保存先基準パスを指定します．動画取得コマンド(GetMovie)を実行する際に保存先パスに相対パスを指定した場合，ここで指定した基準パスからの相対パスとなります． 省略した場合はプロバイダ dll を置いているディレクトリとなります．	--	プロバイダ dll のディレクトリ

使用例(C#)

```
// Engineオブジェクト
ORiN2.ManagedCAO.CCaoEngine engine = new ORiN2.ManagedCAO.CCaoEngine();
// Workspaceオブジェクト
ORiN2.ManagedCAO.CCaoWorkspace workspace = engine.AddWorkspace("NewWrks", "");
// Controllerオブジェクト
ORiN2.ManagedCAO.CCaoController controller
  = workspace.AddController("IB-MCT001",
                            "CaoProv.inaba.IB-MCT001",
                            "",
                            "url = http://192.168.178.178, timeout =
5000,basePath=C:¥¥");
```

3.2.2. CaoController クラス

3.2.2.1. GetVariableNames メソッド

接続可能な変数名リストを取得します．

書式

```
GetVariableNames
(
        "<オプション>"                    // オプション文字列
)
```

3.2.2.2. Variables プロパティ

コントローラが保持する，変数コレクションを取得します．

使用例(C#)

```
// 変数コレクション取得
ORiN2.ManagedCAO.CCaoVariables variables = controller.Variables;
// 変数取得
ORiN2.ManagedCAO.CCaoVariable variable = variables[0];
```

3.2.2.3. AddVariable メソッド

CaoController に変数オブジェクトを追加します. 変数名には 3.3.1 に示すもののみ使用できます.

以下に, AddVariable の仕様を示します.

書式

```
AddVariable
(
        "<変数名>",             // 変数名
        "<オプション>"          // オプション文字列(使用しません)
)
```

3.2.2.4. Execute メソッド

CaoController の拡張コマンドを実行します. 以下に, Execute の仕様を示します. 3.4 に拡張コマンド一覧を定義しています.

書式

```
Execute
(
        "<拡張コマンド名>",             // 拡張コマンド名
        "<オプション文字列>"            // オプション文字列(省略可能)
)
```

3.2.2.5. OnMessage イベント

コントローラのエラー通知や状態の変化を OnMessage イベントとして受け取ることが可能です. 受け取れるイベントについては 3.5 を参照してください.

3.2.3. CaoVariable クラス

3.2.3.1. Value プロパティ

接続したデバイスからデータを取得/設定します. 変数名によって動作が異なります. 詳細は, 3.3. 変数一覧を参照してください.

3.3. 変数一覧

各クラスで使用可能な変数一覧を定義します．なお変数は，CaoVariable クラスのオブジェクトを指します．

3.3.1. CaoController クラス変数

変数名	説明	Value		参照
		get	put	
@MAKER_NAME	メーカー名を取得します.	○	-	P.16
@VERSION	DLL バージョンを取得します.	○	-	P.16
@MODE	カメラモードを取得/設定します. 0: 撮影モード 1: 再生モード	○	○	P.17
@CAMERA_INFO	カメラ情報を取得/設定します.	○	○	P.18
@FILE_LIST	保存されているファイルリストを取得します.	○	-	P.19
@IMAGE	現在のカメラ画像を JPEG コードで取得します.	○	-	P.21
@CAPACITY	カメラの保存残量を取得します.	○	-	P.21
@ERROR_CODE	動作状態およびエラーコードを取得します.	○	-	P.21

3.3.1.1. @MAKER_NAME

メーカー名の取得をします.

データ型

型説明	
VT_BSTR	メーカー名を取得します.

使用例(C#)

```
// 変数追加
ORiN2.ManagedCAO.CCaoVariable var = controller.AddVariable("@MAKER_NAME","");
// 値取得
string value = var.Value as string;
```

3.3.1.2. @VERSION

DLL のバージョンの取得をします.

データ型

型説明	
VT_BSTR	DLL のバージョンを取得します. *.*.*

使用例(C#)

```
// 変数追加
ORiN2.ManagedCAO.CCaoVariable var = controller.AddVariable("@VERSION","");
// 値取得
string value = var.Value as string;
```

3.3.1.3. @MODE

カメラモードを取得/設定します. 変数・拡張コマンドによっては, 実行可能なカメラモードに制約があるものがあります. カメラモードに制約がある変数や拡張コマンドを実行する場合は, 実行前に本変数を利用してカメラモードを設定したのち, 実行してください.

カメラモードによる実行可否

以下に各変数のカメラモードによる実行可否を示します.

変数		撮影モード	再生モード
@CAMERA_INFO	Get	○	○
	Put	×	○
@FILE_LIST	Get	×	○
@IMAGE	Get	○	○
@CAPACITY	Get	○	○
@ERROR_CODE	Get	○	○
@MAKER_NAME	Get	○	○
@VERSION	Get	○	○

以下に各拡張コマンドのカメラモードによる実行可否を示します.

拡張コマンド	撮影モード	再生モード
StartRecording	○	×
StopRecording	○	×
SetTrigger	○	×
GetErrorLog	×	○
GetFileList	×	○
GetMovie	×	○
DeleteFile	×	○

データ型

型説明

VT_I4	カメラモード 0: 撮影モード 1: 再生モード

使用例(C#)

```
// 変数追加
ORiN2.ManagedCAO.CCaoVariable var = controller.AddVariable("@MODE","");
// 値取得
Int32? value = var.Value as Int32?;
// 再生モードに設定
var.Value = 1;
```

3.3.1.4. @CAMERA_INFO

カメラ情報を取得/設定します. 本変数の値設定はカメラモードが再生モードのときのみ実行可能です. また, 文字列は大文字/小文字は区別されますのでご注意ください.

データ型

型説明		
VT_VARIANT | VT_ARRAY		
0	VT_BSTR	カメラ名称(ASCII) 例). CAMERA
1	VT_DATE	現在時刻 例). 2020/10/20 12:15:32
2	VT_BSTR	言語を示す文字列 JP: 日本語 ENG: 英語
3	VT_BSTR	音量を示す文字列 OFF: オフ S: 小 M: 中 L: 大
4	VT_BSTR	カメラの録画モード 録画モードの詳細は, 0 を参照してください. SD: ループアンドトリガロックモード DRAM: トリガーオンリーモード
5	VT_I4	ループアンドトリガロックモードでの録画時間(s) 値範囲: 60, 180, 300, 600

6	VT_I4	トリガーオンリーモードでの録画時間(s) 値範囲: 20, 40
7	VT_I4	カメラの画角設定(度) 値範囲: 110, 180
8	VT_BSTR	バージョン情報 例). 72n
9	VT_I4	工場出荷状態フラグ 0: 工場出荷状態 1: ユーザー設定済み状態
10	VT_I4	通信切断までの時間(min) 無操作の場合, カメラとの接続が切断されます. 0 は無制限であることを示しています. 値範囲: 0, 3, 5, 10

使用例(C#)

```
// 変数追加
ORiN2.ManagedCAO.CCaoVariable cameraInfoVar = controller.AddVariable("@CAMERA_INFO","");
ORiN2.ManagedCAO.CCaoVariable modeVar = controller.AddVariable("@MODE", "");
// 値取得
object[] cameraInfo = cameraInfoVar.Value as object[];

// 名前に CAMERA, 動作モードを DRAM に設定
bool changeMode = false;
if ((modeVar.Value as Int32?).Value != 1)
{
    // カメラモードが撮影モードの場合は, 一度再生モードに設定
    changeMode = true;
    modeVar.Value = 1;
}

cameraInfo[0] = "CAMERA";
cameraInfo[4] = "DRAM";

cameraInfoVar.Value = cameraInfo;

if (changeMode)
{
    modeVar.Value = 0;
}
```

3.3.1.5. @FILE_LIST

カメラに保存されているファイルリスト一覧を取得します. 本変数はカメラモードが再生モード

の時のみ実行可能です.

データ型

<table>
<tr><th colspan="4">型説明</th></tr>
<tr><td colspan="4">VT_VARIANT | VT_ARRAY</td></tr>
<tr><td rowspan="4">i</td><td colspan="3">VT_VARINAT | VT_ARRAY</td></tr>
<tr><td>0</td><td>VT_BSTR</td><td>ファイルパス.
ここで取得したファイルパスは GetMovie コマンド, DeleteMovie コマンドで使用します.
ロックファイルとロックファイル以外のファイルは名前から判断します.
<table><tr><th>ロックファイル</th></tr><tr><td>"IP アドレスの末尾+ 英数字(L or A) + 3 桁のインデックス数字.MOV"</td></tr><tr><th>ロックファイル以外</th></tr><tr><td>"IP アドレスの末尾_3 桁のインデックス数字.MOV"</td></tr></table></td></tr>
<tr><td>1</td><td>VT_DATE</td><td>動画開始日時
例). 2020/10/28 09:00:00</td></tr>
<tr><td>2</td><td>VT_DATE</td><td>動画終了日時
例). 2002/10/28 09:01:00</td></tr>
</table>

使用例(C#)

```
// 変数追加
ORiN2.ManagedCAO.CCaoVariable fileListInfoVar = controller.AddVariable("@FILE_LIST","");
ORiN2.ManagedCAO.CCaoVariable modeVar = controller.AddVariable("@MODE", "");

bool changeMode = false;
if ((modeVar.Value as Int32?).Value != 1)
{
    // カメラモードが撮影モードの場合は, 一度再生モードに設定
    changeMode = true;
    modeVar.Value = 1;
}

object[] fileList = fileListInfoVar.Value as object[];

foreach (object file in fileList)
{
    // 何らかの処理
}

if (changeMode)
{
    modeVar.Value = 0;
```

```
}
```

3.3.1.6. @IMAGE

現在のカメラ画像を JPEG コードで取得します.

データ型

型説明	
VT_UI1 | VT_ARRAY	カメラの JPEG 画像コード カメラモード切替直後は, データが存在しない場合があります.

使用例(C#)

```
// 変数追加
ORiN2.ManagedCAO.CCaoVariable imageVar = controller.AddVariable("@IMAGE","");
Byte[] jpegCodes = imageVar.Value as Byte[];
```

3.3.1.7. @CAPACITY

カメラの保存残量を取得します.

データ型

型説明		
VT_VARIANT | VT_ARRAY		
0	VT_I4	保存残量(sec)
1	VT_I4	ロックファイル有無 0: ロックファイル無 1: ロックファイル有
2	VT_DATE	現在日時 例). 2020/10/28 10:10:15

使用例(C#)

```
// 変数追加
ORiN2.ManagedCAO.CCaoVariable capacityVar = controller.AddVariable("@CAPACITY","");
object[] capacities = capacityVar.Value as object[];

// 保存残量
Int32? capacity = capacities[0] as Int32?;
// ロックファイル有無
Int32? lockFile = capacities[1] as Int32?;
```

3.3.1.8. @ERROR_CODE

動作状態およびエラーコードを取得します.

データ型

型説明		
VT_VARIANT | VT_ARRAY		
0	VT_BSTR	動作状態を示す文字列 この値を見て後述する StartRecording, StopRecording, SetTrigger コマンドの実行を決定してください. また"T000"状態がデバイスのデフォルト状態となります. ほかの状態に遷移していても無通信で一定時間経過するとデバイス側が自動で"T000"状態に遷移するためご注意ください. T000: 録画中 トリガーなし T001: 録画中 トリガーあり T002: 再生中 T003: 録画停止中
1	VT_BSTR | VT_ARRAY	
i	VT_BSTR	エラー状態を示す文字列 E000: 正常動作中 E001: SD カード 未挿入 E002: SD カード 認識不可 E101: Ethernet 初期化エラー E102: 本体システムエラー E103: ファイルシステムエラー

使用例(C#)

```
// 変数追加
ORiN2.ManagedCAO.CCaoVariable errorCodeVar = controller.AddVariable("@ERROR_CODE", "");
object[] errorCodes = errorCodeVar.Value as object[];

// 動作状態
string state = errorCodes[0] as string;
// エラー状態
string[] errorStates = errorCodes[1] as string[];
```

3.4. 拡張コマンド一覧

コントローラクラスの Execute メソッドで使用できる拡張コマンド一覧を記述します.

コマンド	説明	参照
StartRecording	録画を開始します. カメラモードが撮影モード(0)の時のみ実行可能です.	P.24
StopRrecording	録画を停止します. カメラモードが撮影モード(0)の時のみ実行可能です.	P.25
SetTrigger	トリガーを設定し，録画中の動画ファイルをロックします. カメラモードが撮影モード(0)の時のみ実行可能です.	P.26
GetErrorLog	エラーログを取得します.	P.26
GetFileList	現状存在するファイルリストを取得します. カメラモードが再生モード(1)の時のみ実行可能です.	P.26
GetMovie	指定された動画データを非同期で取得します. カメラモードが再生モード(1)の時のみ実行可能です.	P.27
DeleteFile	指定されたファイルを削除します. カメラモードが再生モード(1)の時のみ実行可能です.	P.29

3.4.1. ロックファイルの作成と動画ファイルの取得手順

プロバイダを利用してロックファイルの生成と生成した動画ファイルを取得するための手順を示します.

1. 再生モードに変更する.
   現在のファイルリストを取得するために, **@MODE** 変数を利用して, カメラのモードを再生モードに設定します. (Value プロパティに 1 を設定します.)
2. 現在のファイルリストを取得する.
   @FILE_LIST 変数の値を取得して, 現在のファイルリストをメモリ上に保存しておきます.
3. 撮影モードに変更する.
   録画を開始するために, **@MODE** 変数に 0 を設定しカメラのモードを撮影モードに変更します.
4. 録画を開始する.
   StartRecording コマンドを実行し, 録画を開始します.
5. トリガーをセットする.
   ロックファイルを生成したいタイミングで **SetTrigger** コマンドを実行し, ロックファイルの保存を開始します.
6. ロックファイルの保存が終了するまで待機する.
   @ERROR_CODE の値を取得し, 動作状態がトリガーあり("T001")でなくなるまで待機します. この処理は非同期で行うことをお勧めします.
7. 再生モードに変更する.
   ロックファイルが保存されたら動画ファイルを取得するために, **@MODE** 変数を利用して再生モードに変更します.
8. 現在のファイルリストを取得する.
   @FILE_LIST 変数を利用してロックファイル保存後のファイルリストを取得します.
9. 2 で取得したファイルリストと比較して新しく作成されたロックファイルを抽出する.
   新旧のファイルリストを比較し, 新しくできたロックファイルのファイルパスを取得します. このファイル名は次の工程で使用します.
10. 動画ファイルを取得する.
    GetMovie コマンドを実行し, 生成したロックファイルを保存します.

3.4.2. StartRecording

録画を開始します. カメラモードを撮影モードにして実行してください. デバイスは録画開始状態がデフォルトの状態です.

データ型

項目	型説明
引数	なし
戻り値	なし

使用例(C#)

```
ORiN2.ManagedCAO.CCaoVariable modeVar = controller.AddVariable("@MODE", "");
ORiN2.ManagedCAO.CCaoVariable errorCodeVar = controller.AddVariable("@ERRO_CODE", "");
if ((modeVar.Value as Int32?).Value != 0)
{
    // カメラモードが再生モードの場合は，一度撮影モードに設定
    modeVar.Value = 0;
}

// 録画の開始(通常この操作は必要ありません)
controller.Execute("StartRecording", "");

// ロックファイル保存開始
controller.Execute("SetTrigger", "");

// ロックファイルの保存が終わるまで待機
await System.Threading.Tasks.Task.Run(new Action(() =>
{
    while(true)
    {
        object[] errorCodeValue = errorCodeVar.Value as object[];
        string status = errorCodeValue[0] as string;
        if (status != "T001")
        {
            break;
        }
        System.Threading.Thread.Sleep(0);
    }
}));

// 録画停止(必要であれば実行してください)
controller.Execute("StopRecording", "");

// この後に動画の取得を行いたい場合は GetMovie を参照してください
```

3.4.3. StopRecording

録画を終了します．カメラモードを撮影モードにして実行してください．デバイスのデフォルト状態は録画状態です． 特別な理由がない場合は本コマンドは実行しないことをお勧めします．

データ型

項目	型説明
引数	なし
戻り値	なし

使用例(C#)

StartRecording を参照してください.

3.4.4. SetTrigger

トリガーをセットし，ロックファイルを作成します．録画開始状態で実行してください.

データ型

項目	型説明
引数	なし
戻り値	なし

使用例(C#)

StartRecording を参照してください.

ロックファイル作成完了

ロックファイルを作成している最中は@ERROR_CODE の動作状態文字列が"T001"になります．ロックファイルの作成完了は@ERROR_CODE の動作状態文字列で判断してください.

3.4.5. GetErrorLog

カメラが保持しているエラーログを文字列で取得します．カメラモードを再生モードにして実行してください.

データ型

項目	型説明	
引数	なし	
戻り値	VT_BSTR | VT_ARRAY	
	i	エラーログ

使用例(C#)

```
ORiN2.ManagedCAO.CCaoVariable modeVar = controller.AddVariable("@MODE", "");

if ((modeVar.Value as Int32?).Value != 1)
{
    // カメラモードが再生モードの場合は，一度再生モードに設定
    modeVar.Value = 1;
}

string[] errorLog = controller.Execute("GetErrorLog", "") as string[];
```

3.4.6. GetFileList

カメラに保存されているファイルリストー覧を取得します．カメラモードを再生モードにして実

行してください.

データ型

項目	型説明			
引数	なし			
戻り値	VT_VARIANT | VT_ARRAY			
	i	VT_VARINAT | VT_ARRAY		
		0	VT_BSTR	ファイルパス. ここで取得したファイルパスは GetMovie コマンド, DeleteMovie コマンドで使用します. ロックファイルとロックファイル以外のファイルは名前から判断します. ロックファイル: "IP アドレスの末尾+ 英数字(L or A) + 3 桁のインデックス数字.MOV" ロックファイル以外: "IP アドレスの末尾_3 桁のインデックス数字.MOV"
		1	VT_DATE	動画開始日時 例). 2020/10/28 09:00:00
		2	VT_DATE	動画終了日時 例). 2002/10/28 09:01:00

使用例(C#)

```
ORiN2.ManagedCAO.CCaoVariable modeVar = controller.AddVariable("@MODE", "");

if ((modeVar.Value as Int32?).Value != 1)
{
    // カメラモードが再生モードの場合は, 一度再生モードに設定
    modeVar.Value = 1;
}

object[] fileList = controller.Execute("GetFileList", "") as object[];

foreach (object file in fileList)
{
    // 何らかの処理
}
```

3.4.7. GetMovie

引数で指定された動画ファイルを指定されたパスに保存します. 本コマンドは非同期で実行し, 実

行完了は OnMessage で通知します．ただし，複数同時に実行することはできませんのでご注意ください．また，カメラモードを再生モードにして実行してください．

データ型

<table>
<tr><th>項目</th><th colspan="3">型説明</th></tr>
<tr><td rowspan="3">引数</td><td colspan="3">VT_BSTR | VT_ARRAY</td></tr>
<tr><td>0</td><td>VT_BSTR</td><td>取得先のファイルパスを指定します．</td></tr>
<tr><td>1</td><td>VT_BSTR</td><td>保存先のパスを指定します．
パスの指定は相対パス，絶対パスのどちらでも可能です．
相対パスを指定した場合は AddController 時に指定された基準パスからの相対パスとなります．
指定していない場合はプロバイダ dll が置いてあるパスが基準パスとなります．
パス内に存在しないディレクトリがある場合は自動で作成されますが，管理者権限が必要なパスの場合はディレクトリの作成に失敗します．その場合は事前にディレクトリの作成をお願いします．
<table>
<tr><th>ファイル名を指定しない場合</th></tr>
<tr><td>ファイル名を含まずに指定した場合は以下のファイルが保存されます．
"指定されたディレクトリ" + "取得先ファイル名"
この指定をしたい場合は必ず末尾に"/" or "¥" を指定してください．末尾にこれらの文字がない場合はファイル名と判断されます．</td></tr>
<tr><th>ファイル名まで指定する場合</th></tr>
<tr><td>ファイル名を含んで指定した場合は，指定されたファイルパスにファイルが保存されます．既存のファイルがある場合は上書き保存されるためご注意ください．</td></tr>
</table>
</td></tr>
<tr><td>戻り値</td><td colspan="3">なし</td></tr>
</table>

使用例(C#)

```
// Engineオブジェクト
private ORiN2.ManagedCAO.CCaoEngine engine;
// Workspaceオブジェクト
private ORiN2.ManagedCAO.CCaoWorkspace workspace;
// Controllerオブジェクト
private ORiN2.ManagedCAO.CCaoController controller;

/// <summary>
/// GetMovieのサンプルコードです
```

```
/// </summary>
public void GetMovieSample()
{
    // Engineオブジェクト
    this.engine = new ORiN2.ManagedCAO.CCaoEngine();
    // Workspaceオブジェクト
    this.workspace = engine.AddWorkspace("NewWrks", "");
    // Controllerオブジェクト
    this.controller = this.workspace.AddController("IB-MCT001",
                              "CaoProv.inaba.IB-MCT001",
                              "",
                             "url = http://192.168.178.178, timeout = 5000");
    this.controller.OnMessage += this.OnMessage;

    this.controller.Execute("GetMovie", new string[2] { @"Sample/Sample.mov",
@"C://Sample/Sample.mov" });
}


/// <summary>
/// OnMessageイベントハンドラ
/// </summary>
private void OnMessage (object sender, OnMessageEventArgs e)
{
    if (e.Message.Number == 0)
    {
        // 成功
    }
}
```

3.4.8. DeleteFile

引数で指定されたカメラ内のファイルを削除します.

データ型

項目	型説明	
引数	VT_BSTR	削除するファイルパスを指定します.
戻り値	なし	

使用例(C#)

```
ORiN2.ManagedCAO.CCaoVariable modeVar = controller.AddVariable("@MODE", "");

if ((modeVar.Value as Int32?).Value != 1)
{
    // カメラモードが再生モードの場合は, 一度再生モードに設定
    modeVar.Value = 1;
```

```
}

controller.Execute("DeleteFile", "DeleteFilePath");
```

3.5. イベント一覧

非同期で実行される GetMovie 拡張コマンドの実行結果を OnMessage イベントにより受信することが可能です.

番号	説明
0	動画取得完了イベント 動画取得が正常に完了した際に発行されるイベントです.
1	ファイル操作失敗イベント ファイル操作に失敗した際に発行されるイベントです.
2	動画データ空イベント 取得した動画データが空である際に発行されるイベントです.

Value プロパティデータ型

すべてのイベントでデータ型は共通です.

型説明		
VT_BSTR | VT_ARRAY		
0	VT_BSTR	取得した動画ファイルのパス
1	VT_BSTR	取得した動画データの保存先の絶対パス ファイル操作失敗イベントを受信した際にはこのプロパティに指定されたパスが間違いないか確認してください.

4. プロバイダによるプログラミング

プロバイダでは，以下の手順でクライアント PC とチョコ停ウォッチャーを接続することができます.

- CaoEngine の作成
- CaoWorkspace の作成
- CaoController の作成

チョコ停ウォッチャーに接続した後は，CaoController の Execute メソッドを使用する，もしくは，CaoVariable オブジェクトを生成することで，チョコ停ウォッチャーの情報にアクセスすることができます.

4.1. 現在の画像を取得するするサンプルプログラミング

ここでは例としてチョコ停ウォッチャーの現在の画像を取得するサンプルプログラムを作成します．表 4-1 にサンプルプログラムの要件を，図 4-1 にサンプルプログラムの流れをそれぞれ記述しています.

表 4-1 サンプルプログラムの要件

要件	説明
接続先	接続先 IP アドレスは 192.168.178.178
処理内容	チョコ停ウォッチャーの現在画像を取得する.

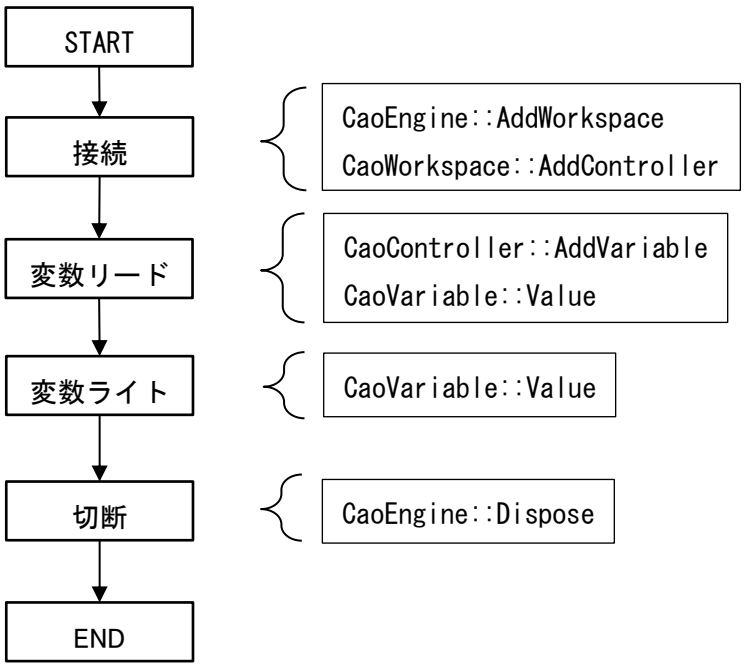


図 4-1 処理の流れ

以降の節から具体的なコードを示します.

4.1.1. サンプルプログラム

以下にサンプルプログラムの全体像を示します.

Sample GetImageSample.cs

```
// オブジェクト
private ORiN2.ManagedCAO.CCaoEngine m_caoEngine = null;
private ORiN2.ManagedCAO.CCaoWorkspace m_caoWorkspace = null;
private ORiN2.ManagedCAO.CCaoController m_caoController = null;
private ORiN2.ManagedCAO.CCaoVariable m_varImage = null;
public void Main()
{
    // 接続
    this.Connect();
    // 変数追加と値の取得
    this.m_varImage = this.m_caoController.AddVariable("@IMAGE","");
    Byte[] jpegCodes = this.m_varImage.Value as Byte[];

    // 切断
    this.Disconnect();
}

// 接続メソッド
private void Connect()
{
    // CaoEngineオブジェクトの生成
    this.m_caoEngine = new ORiN2.ManagedCAO.CCaoEngine();
    // CaoWorkspaceオブジェクトの生成
    this.m_caoWorkspace = this.m_caoEngine.AddWorkspace("NewWrks", "");
    // CaoControllerオブジェクトの生成
    this.m_caoController = this.m_caoWorkspace.AddController("IB-MCT001",
                           "CaoProv.inaba.IB-MCT001",
                           "",
                           "url = http://192.168.178.178, timeout = 5000, basePath=C:¥¥");
}

// 切断メソッド
private void Disconnect()
{
    this.m_caoEngine.Dispose();
    this.m_caoEngine = null;
}
```

4.1.1.1. 接続

チョコ停ウォッチャーと接続するためには, 以下の手順を取ります.

(1) オブジェクトを保持するための変数を用意します. コントローラ接続に必要なオブジェクトは, CaoEngineオブジェクトとCaoWorkspaceオブジェクトとCaoControllerオブジェクトです.

CaoWorkpaceオブジェクトは，CaoControllerオブジェクトをCaoWorkspacesから取得する場合には変数を用意する必要はありません．また変数にアクセスするためのCaoVariableオブジェクトも必要になります．以下にC#でのコード例を示します．

```
// CaoEngine オブジェクト用の変数
private ORiN2.ManagedCAO.CCaoEngine m_caoEngine = null;
// CaoWorkspace オブジェクト用の変数
private ORiN2.ManagedCAO.CCaoWorkspace m_caoWorkspace = null;
// CaoController オブジェクト用の変数
private ORiN2.ManagedCAO.CCaoController m_caoController = null;
// CaoVariable オブジェクト用の変数
private ORiN2.ManagedCAO.CCaoVariable m_varImage = null;
```

(2) CaoEngineオブジェクトを生成します．CaoEngineオブジェクトはNewキーワードを使って生成します．

```
// CaoEngine オブジェクトの生成
this.m_caoEngine = new ORiN2.ManagedCAO.CCaoEngine();
```

(3) CaoWorkspaceオブジェクトを取得もしくは生成します．CaoEngineオブジェクトを生成すると，デフォルトでCaoWorkspacesオブジェクトとCaoWorkspaceオブジェクトを1つずつ生成しています．以下にCaoWorkspaceオブジェクトを新しく生成するコード例とデフォルトのCaoWorkspaceを示します．

```
// CaoWorkspace オブジェクトの生成
this.m_caoWorkspace = this.m_caoEngine.AddWorkspace("NewWrks", "");
```

(4) CaoControllerオブジェクトを生成します．CaoControllerオブジェクトを生成するには，使用するプロバイダ名と使用するためのパラメータを設定します．

```
// CaoControllerオブジェクトの生成
this.m_caoController = this.m_caoWorkspace.AddController("IB-MCT001",
                       "CaoProv.inaba.IB-MCT001",
                       "",
                       "url = http://192.168.178.178, timeout = 5000, basePath=C:¥¥");
```

4.1.1.2. 変数の追加と値の取得

@IMAGE 変数を使用して画像データを取得します．

```
// 変数追加と値の取得
this.m_varImage = this.m_caoController.AddVariable("@IMAGE","");
Byte[] jpegCodes = this.m_varImage.Value as Byte[];
```

4.1.1.3. 切断

コントローラと切断する場合には，生成したオブジェクトを消去すると共に，オブジェクトを管理するコレクションクラスから消去するオブジェクトを削除します．ただし，ORiN.ManagedCAO を使用

した場合は明示的に削除する必要はありません．以下にコード例を示します．

```
// CaoEngine からすべてのオブジェクトを削除
this.m_caoEngine.Dispose();
// CaoEngine の消去
this.m_caoEngine = null;
```

5. プロバイダエラーコード

本プロバイダには,0x8011xxxx ~ 0x8015xxxx でマスクした以下の独自エラーコードが存在します.(表 5-1 独自エラーコード表参照)

表 5-1 独自エラーコード表

エラー番号	説明
0x80110001	AddController時のURLオプション未指定エラーです. AddController時にURLは必ず指定してください.
0x80110002	AddController時のTimeoutオプション指定値エラーです. Timeoutオプションの値範囲外の値を設定した場合に発生します. Timeoutオプションには1 ~ 2147483647 を指定してください.
0x80110003	AddController時のBasePathオプション指定値エラーです. BaePathオプションで指定された基準パスに誤りがある場合に発生します. 正しい基準パスを指定して下さい.
0x80111xxx	AddController時のHTTPエラーです. AddController時の疎通確認に失敗した場合に発生します. URLオプションに指定したURLが正しいかを確認してください. 実際のHTTPレスポンスコードは16進数で 下三桁("xxx") に保存しています.
0x80121xxx	拡張コマンド実行時のHTTPエラーです. 拡張コマンド実行時にHTTPエラーが発生しました. デバイスとPCとの接続状況を確認してください. 実際のHTTPレスポンスコードは16進数で 下三桁("xxx") に保存しています.
0x801220xx	拡張コマンド実行時のデータエラーです. 取得したデータに予期せぬデータが含まれていた, もしくは, コマンド実行時に指定された引数に誤りがあります. xxは拡張コマンド番号表を参照してください.
0x801230xx	コマンド実行時エラー カメラからのエラーです.
0x80124006	動画取得コマンドの重複エラー 動画取得コマンドを実行中に他の動画コマンド取得コマンドを実行しようとしました. 実行中の動画取得コマンドが終了するまでお待ちください.
0x801250xx	拡張コマンド　カメラモードエラーです. 許可されていないカメラモードでコマンドを実行しました.

エラー番号	説明
	xxは拡張コマンド番号表を参照してください.
0x80130000	変数名エラーです. 対象外の変数名が指定されました.
0x80141xxx	値取得時のHTTPエラーです. 値取得時にHTTPエラーが発生しました. 実際のHTTPレスポンスコードは16進数で 下三桁("xxx") に保存しています.
0x801420xx	取得データエラーです. 値取得時に変換できないデータがありました. xxは変数番号表を参照してください.
0x801450xx	許可されていないカメラモードで値取得を実行しました. xxは変数番号表を参照してください.
0x80151xxx	値設定時のHTTPエラーです. 値設定時にHTTPエラーが発生しました. 実際のHTTPレスポンスコードは16進数で 下三桁("xxx") に保存しています.
0x801520xx	設定データエラーです. 値設定で指定されたデータに誤りがあります. xxは変数番号表を参照してください.
0x801530xx	値設定時のカメラからのエラーです. 値設定時にカメラからエラーレスポンスが返ってきました. xxはカメラエラーコード表を参照してください.
0x801540xx	値設定サポート外エラー 値設定動作をサポートしていない変数に対して値設定を行おうとしています. xxは変数番号表を参照してください.
0x801550xx	許可されていないカメラモードで値設定を実行しました. xxは変数番号表を参照してください.

拡張コマンド番号表

拡張コマンド番号	拡張コマンド
0x01	StartRecording
0x02	StopRecording
0x03	SetTrigger
0x04	GetErrorLog
0x05	GetFileList

拡張コマンド番号	拡張コマンド
0x06	GetMovie
0x07	DeleteFile

変数番号表

変数番号	変数
0x01	@MAKER_NAME
0x02	@VERSION
0x03	@MODE
0x04	@CAMERA_INFO
0x05	@FILE_LIST
0x06	@IMAGE
0x07	@CAPACITY
0x08	@ERROR_CODE

カメラエラーコード表

エラーコード	変数
0x01	モード違い
0xFF	コマンド実行失敗(-1)
0xFE	トリガー数エラー(-2)