



Caché ターミナル

Version 5.1

2006-03-14

Caché ターミナル

Caché Version 5.1 2006-03-14

Copyright © 2006 InterSystems Corporation.

All rights reserved.

このドキュメントは、Sun Microsystems、RenderX Inc.、アドビ システムズ および ワールドワイド・ウェブ・コンソーシアム (www.w3c.org) のツールと情報を使用して、Adobe Portable Document Format (PDF) で作成およびフォーマットされました。主要ドキュメント開発ツールは、InterSystemsが構築したCaché と Javaを使用した特別目的のXML処理アプリケーションです。



Caché 製品とロゴは InterSystems Corporation の登録商標です。



Ensemble 製品とロゴは InterSystems Corporation の登録商標です。



InterSystems という名前とロゴは InterSystems Corporation の登録商標です

このドキュメントは、インターシステムズ社(住所: One Memorial Drive, Cambridge, MA 02142)あるいはその子会社が所有する企業秘密および秘密情報を含んでおり、インターシステムズ社の製品を稼動および維持するためにのみ提供される。この発行物のいかなる部分も他の目的のために使用してはならない。また、インターシステムズ社の書面による事前の同意がない限り、本発行物を、いかなる形式、いかなる手段で、その全てまたは一部を、再発行、複製、開示、送付、検索可能なシステムへの保存、あるいは人またはコンピュータ言語への翻訳はしてはならない。

かかるプログラムと関連ドキュメントについて書かれているインターシステムズ社の標準ライセンス契約に記載されている範囲を除き、ここに記載された本ドキュメントとソフトウェアプログラムの複製、使用、廃棄は禁じられている。インターシステムズ社は、ソフトウェアライセンス契約に記載されている事項以外にかかるソフトウェアプログラムに関する説明と保証をするものではない。さらに、かかるソフトウェアに関する、あるいはかかるソフトウェアの使用から起こるいかなる損失、損害に対するインターシステムズ社の責任は、ソフトウェアライセンス契約にある事項に制限される。

前述は、そのコンピュータソフトウェアの使用およびそれによって起こるインターシステムズ社の責任の範囲、制限に関する一般的な概略である。完全な参照情報は、インターシステムズ社の標準ライセンス契約に記載され、そのコピーは要望によって入手することができる。

インターシステムズ社は、本ドキュメントにある誤りに対する責任を放棄する。また、インターシステムズ社は、独自の裁量にて事前通知なしに、本ドキュメントに記載された製品および実行に対する代替と変更を行う権利を有する。

Caché および InterSystems Caché、Caché SQL、Caché ObjectScript および Caché Object は、インターシステムズ社の商標です。

ここで使われている他の全てのブランドまたは製品名は、各社および各組織の商標または登録商標です。

インターシステムズ社の製品に関するサポートやご質問は、以下にお問い合わせください:

InterSystems ワールドワイド カスタマサポート

Tel: +1 617 621-0700

Fax: +1 617 374-9391

Email: support@InterSystems.com

目次

Caché ターミナル	1
1 全般的な特徴	2
2 Caché ターミナルのメニュー	3
2.1 [ファイル] メニュー	3
2.2 [編集] メニュー	4
2.3 [接続] メニュー	9
2.4 ヘルプ	10
3 スクリプト・ファイル	10
3.1 スクリプト・ファイルの開始	10
3.2 スクリプト・ファイルの停止	10
3.3 スクリプト・ファイルの形式	11
3.4 スクリプト・コマンド表	11
3.5 スクリプト・コマンドの引数	12
3.6 個々のスクリプト・コマンドの概要	13
4 拡張キーボード機能	22
4.1 キー・マッピング	22
4.2 キーボード・ファンクション	23
5 DDE の概要	24
5.1 DDE Layout 接続	25
5.2 DDE Screen 接続	25
5.3 DDE Message 接続	26
6 スクリプト例	26
6.1 バッチ・コマンド行	27
6.2 制御引数	28
6.3 スクリプト 1 - プロセス情報の取得 (バッチ)	29
6.4 例 2 - プロセス情報の取得 (インタラクティブ)	31
6.5 例 3 - ホストへの手動接続	31

Caché ターミナル

このドキュメントの目的は、Caché ターミナルの特徴と機能を説明することと、その使用方法に関する基本的な情報を提供することにあります。ここでは、以下のトピックについて説明します。

- ・ 全般的な特徴
- ・ Caché ターミナルのメニュー
 - [ファイル] メニュー
 - [編集] メニュー
 - [接続] メニュー
 - [ヘルプ] メニュー
- ・ スクリプト・ファイルについて
 - スクリプトの開始
 - スクリプトの停止
 - ファイルの形式
 - コマンド表
 - コマンド引数
 - 個々のコマンドの概要
 - ・ Break、Call Script、Case Match、Closelog、Connect、Debug、Disconnect、Display、Echo、Execute、Exit、GoTo、If Empty、Key_Starttime、Key_Stoptime、Key_Timer、Logfile、Multiwait For、Notify、On Error、Pause、Return、Send、Subroutine、Terminate、Test、Timer、Title、Wait For
- ・ キーボード機能
 - マッピング
 - ファンクション
- ・ Dynamic Data Exchange (DDE) の概要
 - Layout
 - Screen
 - Message

- ・ スクリプト例
 - コマンド行
 - 制御引数
 - 例 1 - プロセス情報の取得 (バッチ)
 - 例 2 - プロセス情報の取得 (インタラクティブ)
 - 例 3 - ホストへの手動接続

1 全般的な特徴

Caché ターミナルは、Caché アプリケーションとのやり取りを目的として設計されています。Caché ターミナルは、Digital Equipment Corporation の VT320 ターミナルが持つ機能の多くをエミュレートします。

Caché ターミナルでは、Caché との通信に、ローカルとネットワークという 2 つのメソッドが使用されます。通信モードは、Caché ターミナル・ウィンドウのタイトル・バーに表示されます。

- ・
 - ここで、pid は、Caché ターミナルが通信している Caché プロセスのプロセス ID です。
 - <configname> は、Caché サーバ・プロセスが実行されている Caché 構成です。
- ・ ネットワーク通信では、TCP/IP を介した telnet プロトコルを使用して、Windows Caché サーバと、あるいは UNIX または OpenVMS ホストと通信します。ネットワーク通信が使用されているときは、タイトル・バーに “<Server> NT - Caché Telnet” と表示されます。ここで、<Server> はリモート・サーバのホスト名です。

Caché では、通信スタックとして Winsock が使用されます。Winsock は、Berkeley Software Distribution (BSD) で幅広く使用されている “ソケット” パラダイムをベースとした、Microsoft Windows 用のネットワーク・プログラミング・インタフェースです。ホスト・エントリの指定には、ローカル・ファイル、IP アドレス、または一般的なホスト名 (Winsock にネーム・サーバへのアクセスが実装されている場合) のいずれかが使用されます。ホスト名の後には、標準以外のポート番号を指定するオプションの #nnn が続く場合があります。

この通信モードから報告されるエラーには、Winsock エラー・コードの名前が使用されます。例えば、“WSAECONNREFUSED” は、接続が拒否されたことを意味します。

注釈: 入力バッファのサイズが大きい場合は、Ctrl-C や Ctrl-S などの入力フローの停止を行うキー操作に遅延が生じる場合があります。この遅延はプロセッサと接続速度にも依存します。キーストロークに対応するための専用の処理が、ホスト入力よりも前に実行されています。

メイン・ウィンドウのサイズを変更すると、最下部に最新行 (ホストからの) が表示された状態が維持されるように調整されます。ユーザはスクロールバックして、前の行に戻ることができます。アクティブなテキストが到着したときは、Caché ターミナルによって、ウィンドウが新しく到着したテキストに移動されます。このバージョンでは、24 x 80 構成でおおよそ 375 行のスクロールバック・バッファがサポートされます。

2 Caché ターミナルのメニュー

Caché ターミナルは Windows アプリケーションとして、そのウィンドウ上部に標準的なメニュー項目が配置されています。ここでは、各メニュー項目とそのサブメニューについて説明します。

2.1 [ファイル] メニュー

[ファイル] コマンドは、ログ・ファイルおよびスクリプト・ファイルを制御するときと、テキスト・ファイルを印刷するときに使用します。

2.1.1 ログイン (Alt-L)

このコマンドは、セッションのログインを開始します。ログイン・データを収集して記録するファイルの名前を指定します。接続からの出力のみがログに記録されます (現行のラップ・モードには関係ありません)。アクティブなログ・ファイルがある場合は、アクティブなファイルへのログインを停止するかどうかを尋ねられます。

既定では、ログ・ファイルは新規作成されず、追加記入されます。ユーザは、[上書き] ボタンをクリックすることで、ログ・ファイルの上書きを選択できます。ログ・ファイルの既定の名前と場所は、次のレジストリ・キーから取得されます。

```
HKEY_CURRENT_USER
  Software
    Microsoft
      Windows
        CurrentVersion
          Explorer
            ComDlg32
              OpenSaveMRU
                <xxx>
```

<xxx> は log になります。スクリプトの場合は、scr から情報が取得されます。

2.1.2 スクリプト (Alt-S)

このコマンドは、スクリプト・ファイルの実行を開始します。スクリプト・ファイルが実行中の場合は、このコマンドを使用してスクリプト・ファイルを停止します。スクリプト・ファイルの詳細は、“[スクリプト・コマンド](#)” の説明を参照してください。

2.1.3 ログの印刷

このコマンドは、作業ディレクトリからログ・ファイルを印刷します。また、任意の ASCII ファイルの印刷にも使用します。改ページ文字を適切に処理する以外は、特別な操作は必要ありません。印刷中は、マウスおよびキーボード入力がメイン・ウィンドウからロック・アウトされ、キャンセル用のダイアログ・ボックスが表示されます。印刷はドラフト・モードで実行されます。

2.1.4 印刷設定

このコマンドは、Caché ターミナル用のプリンタの選択および設定に使用できます。

2.1.5 画面印刷

このコマンドは、Caché ターミナル画面の内容を、既定のプリンタに送信します。

2.1.6 終了

このコマンドを使用すると、Caché ターミナルの現行のコピーが終了し、すべてのオープン・ファイルがクローズされます。ターミナルは Alt-F4 でも終了できます。

Caché ターミナルの現行のコピーが起動時にサーバに接続されている場合は、通信チャンネルがクローズされたときに自動的に終了します。ターミナルが Caché キューブから [リモート システム アクセス]→[Cache Telnet] を使用して起動されている場合は、通信チャンネルがクローズされても、ターミナルは自動的に終了せずアクティブ状態が維持され、[接続] メニューを使用して別の接続を確立できます。

2.2 [編集] メニュー

[編集] コマンドでは、クリップボードの制御とウィンドウ表示の再初期化に加えて、個々のユーザによってカスタマイズされた項目の選択がサポートされます。

2.2.1 コピー (Ctrl-Ins)

このコマンドは、標準的な Windows パラダイムに従って、選択されたテキストを Windows クリップボードにコピーします。

コピーされたテキストに行の境界が含まれる場合は、キャリッジ・リターンと改行としてクリップボードに保存されます。改行の貼り付けは、不要な場合もあります。[ユーザ設定] コマンドを参照してください。

ホストに処理中のマウス・リクエストがあるときに、切り取りおよび貼り付けをローカルで実行したい場合は、対象領域を選択しながら Ctrl キーを押すと、マウスのアクションがホストに報告されなくなります。

2.2.2 貼り付け (Shift-Ins)

このコマンドは、現在のテキストをクリップボードからホストに送信します。このテキストは、エコーが無効にされていない限り、ターミナル・ウィンドウに表示されます。マウスを右クリックすると、貼り付け用の次のオプションを持つメニューが表示されます。

- ・ コピー
- ・ 貼り付け
- ・ コピー+貼り付け

Caché ターミナルのデータ貼り付け速度が、ホストのデータ受信速度よりも速い場合がしばしばあります。貼り付け速度を制御するための設定は、“[ユーザ設定](#)”を参照してください。また、貼り付けコマンドの実行時に、改行を破棄することもできます。

2.2.3 リセット

このコマンドを使用すると、現在のページ上のマージン、スクロール領域、およびその他の処理がリセットされ、ウィンドウが再描画されます。通常、この操作が必要になることはありません。

2.2.4 削除 (Ctrl-Del)

このコマンドは、すべてのデータを消去し、スクロールバック領域を 0 に戻すことで、Caché ターミナル・ウィンドウを再初期化します。

2.2.5 フォント

このコマンドを使用して、各自のモニタと解像度に最適なフォント・スタイルを選択できます。Caché ターミナルを異なるサイズ画面に切り替えると、事前選択されているフォントが使用されます。

注釈: 画面の枠を超えてウィンドウが拡張するようなフォント・サイズに設定した場合は、画面とフォントの両方が使用可能な最大サイズに自動調整されます。

2.2.6 色

このコマンドでは、Caché ターミナルの既定の前景色と背景色を選択できます。[適用] をクリックすると、現在のコピーにのみ変更が適用されます。[保存] をクリックすると現在のコピーは変更されず、Caché ターミナルの新規インスタンス用に色情報が保存されます。

色調整では、ANSI 名で事前指定された色から、ディスプレイ・ボードが提供する任意の色に変更できます。これらの色は、前景色および背景色とともに保存されます。既定色を選択する場合は、[既定] ボタンをクリックします。

2.2.7 ユーザーキー

ユーザ・キーは、Alt-Shift-F1 から Alt-Shift-F10 までの間で選択します。[OK] と [保存] を続けて選択すると、現在のインスタンスが更新されると同時に、以降の Caché ターミナル・インスタンス用にキー・シーケンスが保存されます。

表示不可能な文字を含める場合は、<nnn> を使用します。nnn はその文字と同等の 10 進数値です。また、<CR>、<F10>、<F7>、<DO>、<TAB>、<LF>、<ESC>、<CSI>、<NL> (= <CR><LF>) の 1 つを使用できます。

さらに、<P1>、<P2> などは、コマンド行パラメータを指定します。

注釈: このバージョンの Caché には、[ユーザーキー]機能に関する既知の問題があります。最新の詳細情報は、インターシステムズのサポート窓口にお問い合わせください。

2.2.8 ユーザ設定

[ユーザ設定] は、Caché ターミナルによって使用される各種パラメータの現行設定値と初期値の両方を制御します。シンプルな設定を以下に示します。

設定	説明
Wrap	右側の列での自動折り返しを設定します。
<- Key sends ^H	<X> キーが Delete ではなく Ctrl-H を送信するように設定します。
Application Keypad	アプリケーション・モードのキーパッドを有効化します。
Force Numeric Pad	標準の PC キーパッドを強制します。
Disable Special ID	特殊なターミナル ID を送信しません。
Disable Mouse Reports	マウス・レポートを送信しません。
Enable Fast Paint	高速描画モードを有効化します。
Paste Keeps Linefeed	クリップボード内の LF を (CR とともに) 送信するように設定します。
Paste Size	クリップボードから一度に送信するバイト数を設定します。
Paste Wait	バースト間の待機時間をミリ秒単位で設定します。
Pass XOFF Through	リモート・ホストが XOFF/XON を処理するように設定します。
Paste burst size (in bytes)	一度の送信に貼り付けられる文字数を設定します。
Paste pause time (in msec)	バースト・サイズよりも長いマテリアルが貼り付けられた場合の、連続する送信の間隔をミリ秒単位で設定します。

一部のシステムでは、そのデータ受信速度が、Caché ターミナルのデータ送信速度よりも遅い場合があります。その場合は、[Paste Burst Size] で一度に送信するデータ量をバイト数で指定し、[Paste Pause Time] で送信間の一時停止時間をミリ秒単位で指定します。[Paste Burst Size] または [Paste Pause Time] の値が 1 未満の場合は、クリップボード全体が一度に送信されます。

2.2.9 ウィンドウサイズ

ウィンドウ・サイズを選択できます。列の最大数は 132 で、行の最大数は 64 です。行と列の両方に対して、一般的に使用するサイズをすばやく選択するためのボタンが用意されています。

ウィンドウ・サイズを変更すると、使用可能なスクロールバック行および“ページ”の数がただちに更新されます。[適用] を選択すると現在のインスタンスが更新され、[保存] を選択すると以降の Caché ターミナル・インスタンス用に値が保存されます。

ウィンドウを変更すると、現在の表示ページとすべてのバック・ページにある現行のすべてのデータが消去されます。さらには、新しいサイズ用に選択されているフォントがある場合は、そのフォントも選択されます。

2.2.10 ネットワークエンコーディング

Caché ターミナルでは、ディスプレイ・メモリ内の文字は Unicode で格納されますが、キーボード入力はシングルバイトまたはマルチバイトの文字ストリームとして受け取られ、Windows の入力コード・ページに応じて Unicode に解釈および変換されます。また、ピア・サーバとの通信では、内部の Unicode 文字がネットワーク・エンコードの文字ストリームに変換され、ピアから受け取った文字が Unicode に変換される必要があります。

キーボードから受け取った文字は、現行の Windows 入力文字セットを使用して、シングルバイトまたはマルチバイトの文字ストリームから変換されます。これは、入力言語を変更した場合に、その変更が Caché ターミナルによって認識され、対応処理されることを意味します。これによって、複数言語の混入入力が可能となり、混入入力はターミナルによって認識され、内部の Unicode 表現に適切に変換されます。複数言語を混入入力する場合は、ネットワーク・エンコードおよび Caché \$ZMODE 入出力変換テーブルとして UTF8 を選択する必要があります。

サーバに転送される文字は内部の Unicode 表現からネットワーク・エンコードに変換され、サーバから受け取った文字はネットワーク・エンコードから Unicode に変換されます。この変換は、[ネットワークエンコーディング] メニューで選択したネットワーク・エンコードによって規定されます。選択したネットワーク・エンコードは、使用している接続方法に影響されます。ローカル・エンコードは Caché ターミナルがローカルに接続されているときに [ネットワークエンコーディング] メニューによって設定され、リモート・エンコードは telnet 接続がアクティブなときに設定されます。選択できるエンコードは、UTF8、Windows、ISO、および EUC の 4 つです。これらのエンコードのすべてがあらゆる入力ロケールに関連するわけではないため、関連するエンコードのみがメニューに表示されます。

UTF8 を選択すると、内部の Unicode 文字は、サーバへの出力時は UTF8 に変換され、サーバからの受信時は UTC8 から変換されます。UTF8 を選択する場合は、主入出力デバイスの Caché 入出力変換が UTF8 である必要があります。この入出力変換は、\$ZMODE によって確認できます。“¥”で区切られた 4 番目のフィールドにあります。

Windows を選択すると、Caché ターミナルとサーバ間での内部の Unicode 文字セット・エンコードに対する入出力変換が、現行の Windows 入力コード・ページを使用して実行されます。Windows エンコードを使用するときは、\$ZMODE で表示される Caché 入出力変換が、アクティブな Windows コード・ページが示す文字セットになるように設定される必要があります。

ISO を選択すると、ピア・サーバ間との入出力変換に、以下の ISO 8859-X コード・ページが使用されます。適切な ISO コード・ページは、現行の Windows 入力コード・ページに基づいて選択されます。有効な対応関係は、以下のとおりです。

言語地域	ISO 標準	Windows コード・ページ	ネットワーク・ コード・ページ
西ヨーロッパ	8859-15	1252	28605
中央ヨーロッパ	8859-2	1250	28592
キリル文字	8859-1	1251	28591
ギリシャ語	8859-7	1253	28597
トルコ語	8859-9	1254	28599
ヘブライ語	8859-8	1255	28598
アラビア語	8859-6	1256	28596
バルト・リム語	8859-4	1257	28594
韓国語	iso-2022-kr	949	50225
日本語 (JIS)	N/A	932	50220

ISO ネットワーク・エンコードが選択されている場合は、他のすべての Windows 入力コード・ページで Windows コード・ページが使用されます。

ISO エンコードを使用するときは、\$ZMODE で表示される Caché 入出力変換が、Caché ターミナルによって使用されるアクティブな ISO コード・ページが示す文字セットと矛盾しないように設定される必要があります。

EUC エンコードは、極東地域の言語に関連するもので、一部の UNIX システムとの通信に使用されます。EUC を選択すると、サーバ間との入出力変換に、以下のコード・ページが使用されます。適切な EUC コード・ページは、現行の Windows 入力コード・ページに基づいて選択されます。有効な対応関係は、以下のとおりです。

言語地域	ISO 標準	Windows コード・ページ	ネットワーク・ コード・ページ
日本語	N/A	932	51932
簡体字中国語	N/A	936	51936
韓国語	N/A	949	51949

日本語 (JIS) のサポートは、50220 コード・ページを使用する ISO ネットワーク・エンコードによって実現され、内部の Unicode との変換が行われます。

2.2.11 物理文字表示の設定

ターミナル・ウィンドウに表示される文字の形態を選択できます。オプションは [論理] と [物理] です。両者の相違がわかるのは、マルチバイトの文字セットの使用時のみです。

2.3 [接続] メニュー

[接続] コマンドは、Telnet モジュールの操作をサポートします。

注釈: ターミナルが `/console=` 制御引数を使用して起動された場合は、このメニュー項目は表示されません。

2.3.1 ホスト (Alt-O)

このコマンドは、Caché ターミナルからリモート・ホストへの接続を許可します。この項目を選択すると、目的のホストのアドレスを取得するためのダイアログ・ボックスが表示されます。

2.3.2 ブレーク信号を送る (Alt-B)

このコマンドは、通信チャネル経由でブレークを送信します。

2.3.3 切断

ターミナルがホストに接続済みの場合は、この項目を選択すると、ホストへのターミナル接続が切断されます。リモート・ホストへの接続が試行中で未完了の場合は、試行中の接続が切断されます。

2.3.4 使用可能なホストの一覧

[切断] メニュー項目の下に、便宜上の目的で既知のホストの数値付き一覧が表示されます。

2.4 ヘルプ

[ヘルプ] では、(このヘルプを含む) 様々なタイプのヘルプを選択できます。

2.4.1 [ヘルプ] メニュー

このメニュー項目を選択すると、このマテリアルが表示されます。

2.4.2 バージョン情報

Caché ターミナルの現行のコピーのバージョン番号が表示されます。

3 スクリプト・ファイル

スクリプト・ファイルを使用すると、単調な入力作業が不要になり便利です。Caché ターミナルの重要な制御は、スクリプトによって実行できます。

3.1 スクリプト・ファイルの開始

通常、スクリプト・ファイル (既定の拡張子は .SCR) は作業ディレクトリにあります。任意の場所に配置可能です。

スクリプトは、コマンド行の第一引数として起動されるか、または [ファイル] メニューの [スクリプト...] コマンド (ホットキーの Alt+S でも選択可能) から起動されます。Windows の標準的なファイル検索ボックスが表示され、そこでスクリプトを選択します。

コマンド行の引数としてスクリプトが指定されている場合は、コマンド・モードをロックするスイッチがなければスクリプトが直ちに開始され、スイッチがあればホスト接続が完了するまで開始が延期されます。

注釈: 通信オプションをシングル・モードに編集することは、Caché ターミナルをシングル・オプションにロックすることと同じです。したがって、スクリプト・ファイルの起動はホスト接続の完了後まで延期されます。

3.2 スクリプト・ファイルの停止

スクリプトを停止するには、[ファイル] メニューの [スクリプト...] コマンドを再度選択します。現行のスクリプトを停止するかどうかを尋ねられます。[はい] を選択して、新規スクリプトの選択をキャンセルします。

3.3 スクリプト・ファイルの形式

スクリプトは1行指向で、行継続の表記規則がありません。各行は他の行から完全に分離されています。セミコロンで始まる行はコメントとみなされます。空白行は、読みやすさを向上させるために自由に挿入することができます。通常、無効な行は無視されます。スクリプト・コマンドは、スペースおよび/またはタブの後に続けます。

スクリプト・コマンドを含む行の一般的な形式を、次に示します。

```
<ScriptCommand>: <ScriptArguments>
```

ScriptCommand とその ScriptArguments は、コロンによって分割されます。スクリプト・コマンドの引数は、文字列または数値です。コマンドに引数がない場合は、コマンドと引数を分割するコロンは省略する必要があります。

ラベルは、制御の移動箇所を定義するために使用されます。ラベルはドル記号 (\$) で始まり、大文字と小文字が区別されません。ラベルには、スペースを埋め込むことができます。ラベルは、1行に単独で記述する必要があります。

注釈: <ScriptCommand> が2つ以上の単語で構成されている場合は、コマンドの各語間を1つのスペースで区切る必要があります。

<ScriptCommand> の後にコロンを使用する場合は、コマンドの最後の語の直後に記述する必要があります。

3.4 スクリプト・コマンド表

次の表は、使用可能なスクリプト・コマンドの一覧です。

コマンド	アクション
<code>break</code>	ブレークをサポートする通信デバイスにブレークを送信します。
<code>call script</code>	現在のスクリプトを終了して、他のスクリプトを開始します。
<code>case match</code>	“wait for” コマンドの文字列が大文字/小文字まで一致する必要があるかどうかを指定します。
<code>closelog</code>	ログ・ファイルをクローズします。
<code>connect</code>	ホストに未接続の場合に、ホスト接続を強制します。
<code>debug</code>	スクリプトのデバッグを有効化または無効化します。
<code>disconnect</code>	ホストに接続されている場合に、接続の切断を強制します。
<code>display</code>	ディスプレイにテキストを送信します。
<code>echo</code>	入力文字のエコーのオン/オフを切り替えます。

コマンド	アクション
<code>execute</code>	Windows プログラムを実行します。
<code>exit</code>	スクリプトを終了します。
<code>goto</code>	制御をスクリプト内の別の場所に移します。
<code>if empty</code>	最後のテスト文字列が空の場合に、制御を移動します。
<code>key_starttime</code>	キーの時間計測の開始をシミュレートします。
<code>key_stoptime</code>	キーの時間計測の停止をシミュレートします。
<code>key_timer</code>	キーの時間計測のオン/オフを切り替えます。
<code>logfile</code>	ログ・ファイルを開始します。
<code>multiwait for</code>	通信デバイスからの任意の複数文字列を待機します。
<code>notify</code>	ダイアログ・ボックスを表示して、ユーザからの応答を待ちます。
<code>on error</code>	タイマーが起動された場合の分岐先ラベルを指定します。
<code>pause</code>	スクリプトを一時停止します。
<code>return</code>	スクリプト・ファイルのサブルーチンから戻ります。
<code>send</code>	通信デバイスにテキストを送信します。
<code>subroutine</code>	スクリプト・ファイル内のサブルーチンを呼び出します。
<code>terminate</code>	エミュレータを完全に終了します。
<code>test</code>	テスト対象の文字列を構築します。
<code>timer</code>	“wait for” コマンド用のタイマーを制御します。
<code>title</code>	ウィンドウ・タイトルを設定します。
<code>wait for</code>	通信デバイスからの特定の文字列を待機します。

3.5 スクリプト・コマンドの引数

スクリプト・コマンドは、そのコマンドの機能に応じて、文字列または数値を引数として受け取ります。引数の前後のスペースおよびタブは、すべて無視されます。

数値引数は、すべて整数値です。必須の数値引数が指定されていない場合は、既定で 0 になります。また、OFF は 0 と同等で、ON は 1 と同等です。

文字列は、コマンドの後に続く行上の個々のデータを連結したものにすぎません (先頭と末尾の空白は除く)。引用符は必要ありません。また、次のコマンド行を使用することで、パラメータ置換が実行されます。

<P1>, <P2>, ..., <Pn>

<Pn> が、n 番目のコマンド行パラメータに置換されます。

注釈: NUL (000) 以外のすべての ASCII (拡張) 文字は、<ddd> によって生成できます。ddd はその文字を表す 10 進数値です。

文字	解釈	送信シーケンス
<CR>	キャリッジ・リターン	<13>
<F10>	F10 キー	<27>[21-
<F7>	F7 キー	<27>[18-
<DO>	Do キー	<27>[29-
<TAB>	Tab キー	<9>
<LF>	改行	<10>
<ESC>	Esc キー	<27>
<DCS>	デバイス制御文字列の接頭部	<144>
<ST>	デバイス制御文字列の終了	<156>
<EMU>	拡張エミュレータ・コマンドの開始	<144>i
<NL>	新規行	<CR><LF>
<CSI>	制御文字列の接頭部	<155>

3.6 個々のスクリプト・コマンドの概要

このセクションでは、個々のスクリプト・コマンドの機能とその使用方法について説明します。

3.6.1 Break

このコマンドは、ブレイクをサポートする通信ノードに“ブレイク”を送信します。それ以外の場合には、“空命令”として動作します。引数は取りません。

使用例：

```
break
```

3.6.2 Call Script

call script コマンドは、スクリプトの実行を開始します。このコマンドを実行したときにスクリプトが実行中の場合は、実行中のスクリプトを終了してから、新しいスクリプトが開始されます。

使用例 :

```
call script: login fred <p3>
```

このコマンド例は、現行のスクリプトを停止し (スクリプトが実行中の場合)、**login.scr** という名前の別のスクリプトを開始します。login の第 1 パラメータは "fred" で、第 2 パラメータは現行のスクリプト・ファイル (呼び出しを行っているスクリプト・ファイル) の第 3 パラメータと同じになります。既定の拡張子は ".SCR" と想定されます。最初に、現行の作業ディレクトリで **login.scr** のインスタンスが検索されます。

3.6.3 Case Match

case match コマンドは、wait for コマンドでの大文字/小文字の照合を有効化または無効化します。

使用例 :

```
case match: off
```

このコマンド例は、個々の文字の大文字/小文字が異なっても、文字列が一致しているとみなされます。このスイッチの既定は on です。

3.6.4 Closelog

このコマンドは、現在のオープン・ログ・ファイルをクローズします。オープンされているログ・ファイルがない場合は、このコマンドは何も影響しません。

使用例 :

```
logfile: mydirect.log  
send: dir *.* /FULL<CR>  
wait for: <NL>$  
closelog
```

3.6.5 Connect

このコマンドは、リモート・ホストへの接続を開始するためのダイアログ・ボックスを開きます。

使用例 :

```
connect
```

3.6.6 Debug

debug コマンドは、無効なスクリプト・コマンドの特定に役立ちます。通常、Cache ターミナルは無効なスクリプト・コマンドを無視します。デバッグ・モードを有効にすると、オペレータの注意を喚起するメッセージ・ボックスに、無効なコマンドの最初の部分が表示されます。

使用例 :

```
debug: on
```

3.6.7 Disconnect

Caché ターミナルがホストに接続されている場合、このコマンドは [接続] メニューの [ホスト] の [切断] オプションと同等です。未接続の場合は、何の動作もしません。

使用例 :

```
disconnect
```

3.6.8 Display

display コマンドは、データを画面に出力します。通信デバイスに、データは送信されません。

使用例 :

```
display: <CSI>H<CSI>J<LF>Here are the choices for today:
```

このコマンド例を実行すると、カーソルがホーム位置に戻され、ウィンドウがクリアされます。次に、1 行進めて "Here are the choices for today:" というテキストが書き込まれ、テキストの末尾にカーソルが置かれます。

3.6.9 Echo

echo コマンドは、ウィンドウおよびログ・ファイルへの出力の表示を有効化または無効化します。このコマンドは、作業の実行時に、ウィンドウやログ・ファイルに出力を表示したくない場合に便利です。

使用例 :

```
echo: off
on error: $Failed Login
timer: 50
wait for: Name:
send: <p1><CR>
wait for: Password:
send: <p2><CR>
wait for: <NL>$
echo: on
Notify: Login is complete
display: <CSI>H<CSI>J
send: <CR>
goto $Process

$Failed Login
  echo: on
  notify: Login failed.
  exit

$Process
  ;processing begins
```

スクリプト・ファイル

このコマンド例は、最初の 2 つのスクリプト・パラメータで指定された名前とパスワードを使用するログイン・シーケンスを非表示にします。ログインに成功すると、指定されたラベルの箇所での処理が開始されます。失敗の場合は、ログインに失敗したことを示すダイアログ・ボックスが表示され、ユーザが [OK] をクリックすると終了されます。

3.6.10 Execute

このコマンドは、Windows プログラムを起動し、そのウィンドウに SHOW 属性を設定します。

使用例：

```
execute: notepad.exe myfile.not
```

このコマンド例は、Windows のメモ帳プログラムを起動し、アプリケーション内でファイル **myfile.not** をオープンします。次のような使用方法にも注目してください。

```
logfile: mydat.lst
echo: off
send: dir *.dat/full
wait for: <NL>$
closelog
echo: on
execute: notepad mydat.lst
```

注釈： プログラムが実際に開始されたかどうかを確認するテストは実行されず、その完了も待機されません。

3.6.11 Exit

exit コマンドは、スクリプトを終了するときに使用します。通常、スクリプトは“最下行まで実行されると”終了しますが、特定のイベント（ログインなど）が生じないときに終了させたい場合があります。

使用例：

```
on error: $byebye
timer: 40
wait for: event:
goto: $Got event
```

```
$byebye:
notify: Did not find event prompt, exiting script
exit
```

```
$Got event:
timer: 0
; more commands
```

3.6.12 GoTo

goto コマンドは、制御をスクリプト・ファイル内の別の箇所に移すときに使用します。このコマンドは、ループ処理のコントロール・フローの管理や、タイムアウト分岐への対応の使用に便利です。

使用例 :

```
on error: $Not There
timer: 30
wait for: abc<CR>
goto: $Got It

$Not There:
;failed to see it, send Ctrl+C
send: <3>
goto: $bad

$Got It:
;turn timer off because we got abc<CR>
timer: 0

;more commands ...
```

3.6.13 If Empty

if empty コマンドを使用すると、最後の test コマンドが空文字列を検出した場合に、指定したラベルに分岐させることができます。

使用例 :

```
test: <p1>
if empty: $No First Arg
```

最初のコマンドは、コマンド行で指定された第 1 パラメータが見つかるかどうか、つまり空でないかどうかを検証します。2 番目のコマンドは、それが見つからない場合に、ラベル "\$No First Arg" に分岐させます。

3.6.14 Key_Starttime

このコマンドは、「キー」・シーケンスの時間計測を開始するときに使用します。このコマンドは、1 つの数値引数を取ります。引数が 0 の場合は、「ENTER」と入力されたときに、統計が蓄積されます。それ以外の場合は、F10 が押されたときに、統計が蓄積されます。

使用例 :

```
key_starttime: 0
```

時間計測は、key_stoptime コマンドによって停止できます。

3.6.15 Key_Stoptime

このコマンドは、時間計測がアクティブな場合に、時間計測を停止し統計を蓄積します。

使用例 :

```
key_starttime: 0
wait for: <esc>[14;22H
key_stoptime
```

3.6.16 Key_Timer

key_timer コマンドは、キーの時間計測情報のデータ収集を開始または停止します。代わりに、特殊キーの Alt-Shift-T を使用して、キー・タイマーを開始または停止することもできます。

使用例：

```
key_timer: on
; rest of your script commands
key_timer: off
```

注釈： 時間計測をスクリプト・ファイルから排他的に起動する場合は、<13>、“<27>[21-” の代わりに、それぞれ <CR>、<F10> を使用する必要があります。

3.6.17 Logfile

このコマンドは、指定されたログ・ファイルで受信データの収集を開始します。アクティブなログ・ファイルがある場合は、適切に終了されます。ロギングの停止には、closelog コマンドを使用します。

使用例：

```
logfile: mydirect.log
send: dir *.* /FULL<CR>
wait for: <NL>$
closelog
```

既定のディレクトリは、システム・マネージャ・ディレクトリです。

通常、ログ・ファイルは追加書き込み方式でオープンされます。つまり、ログ・ファイルが既存する場合は、新規データがその末尾に追加されます。

3.6.18 MultiWait For

multiwait for コマンドは、スクリプト・ファイルとホストを同期化します。ホストから受信したデータが、引数に指定されいくつかの文字列の 1 つに一致するまで、処理が中断されます。

使用例：

```
multiwait for: =USER>=***ERROR,=DONE
```

このコマンド例では、指定された 3 つの文字列の中の 1 つが到着するまで、スクリプト・ファイルが待機することになります（永久に待機する可能性もあります）。引数の最初の非空白文字（この例では等号記号）は、引数を複数の部分文字列に分割する“区切り文字”としての役割を持ちます。

したがって、このコマンド例は、“USER>”、“***ERROR”、または“DONE”の中のいずれかが到着するまで、スクリプトを待機させます。

multiwait for コマンドの終了には、タイマーを使用できます。

大文字/小文字の完全一致を有効または無効にする場合は、case match コマンドを参照してください。

case match コマンドの引数には 1 つの部分文字列しか指定できないため、次の 2 つのスクリプト・コマンドの機能は同じになります。

```
multiwait for: =USER>
wait for: USER>
```

3.6.19 Notify

notify コマンドは、ユーザに Windows メッセージ・ボックスを表示し、そこで [OK] ボタンが押されたから、操作を続行します。このコマンドは、ユーザに重要なイベントを通知するときに使用できます。

使用例：

```
notify: Ready to send commands...
send: copy *.lst backup:*.lst<CR>
send: delete *.lst;*
```

注釈： このメッセージ・ボックスは“モーダル”です。つまり、他のユーザが割り込むことはできません。

3.6.20 On Error

on error コマンドは、タイマーの有効時間が経過した場合（主に、テキストの到着を待っているとき）に実行するスクリプト・コマンドのラベルを指定する方法を提供します。

使用例：

- ・ 使用例は、goto コマンドを参照してください。

3.6.21 Pause

このコマンドは、実行中のスクリプトを一時停止するときに使用します。停止時間は 10 分の 1 秒単位で指定します。

使用例：

```
pause: 30
```

このコマンド例では、スクリプトの実行が 3 秒間停止されます。pause に 0 を指定した場合は永久停止となり、スクリプト処理の再開には Alt-P が必要になります。

3.6.22 Return

return コマンドは subroutine コマンドとともに使用して、スクリプト内でそのサブルーチンを呼び出した箇所に戻ります。

使用例：

- ・ 使用例は、subroutine コマンドを参照してください。

3.6.23 Send

send コマンドは、現在接続中のホストに送信する入力データをシミュレートします。

使用例：

```
send:  1<cr>2<cr>A1234<F10><32>
```

このコマンド例は、以下の入力と同等です。

- ・ 文字の 1
- ・ キャリッジ・リターン・キー
- ・ 文字の 2
- ・ キャリッジ・リターン・キー
- ・ 文字列の "A1234"
- ・ F10 キー
- ・ 1 つのスペース文字

スペースはコマンド・インタプリタによって削除されるため、<32> は先頭または末尾のスペースを送信する唯一の方法であることに注意してください。

3.6.24 Subroutine

subroutine は、スクリプト内で同じコマンドを何度も使用する場合に役立ちます。このコマンドは、記憶域を節約すると同時に、異なる多くのラベルを保持する必要性を減じます。このコマンドは return コマンドとともに使用します。

使用例：

```
subroutine: $Send It Again
; some other processing
exit

$Send It Again:
send: <F7>Q
on error: $skip
timer: 30
wait for: [22;5H
timer: 0
return

$skip:
send: <3>
; note on error still set to $skip
timer: 30
wait for: function:
timer: 0
send: <CR>
exit
```

注釈: サブルーチン・スタックは 16 アドレスを保持します。サブルーチン呼び出しのネストがそれよりも深くなると、スクリプトは失敗します。

3.6.25 Terminate

このコマンドは、終了して Windows に戻るよう Caché ターミナルに命令します。すべてのオープン・ファイルがクローズされ、不要なウィンドウが消去され、接続がクローズされます。

使用例 :

```
terminate
```

3.6.26 Test

test コマンドは、パラメータまたはウィンドウ・プロパティが空でないかどうかを調べるのに使用します。このコマンドは、if empty コマンドと連携して使用します。

使用例 :

- ・ 使用例は、if empty コマンドを参照してください。

3.6.27 Timer

timer コマンドは、wait for コマンドと連携して使用されるタイマーを設定します。対象のテキストが到着しない場合、またはそのテキストが時間計測や大文字/小文字の不一致が原因で見つからない場合は、timer が wait for に割り込みスクリプトの実行を継続する唯一の方法です。

使用例 :

```
timer 100
```

引数の数値は待機時間で、10 分の 1 秒単位で指定します。このコマンド例では、タイマーに 10 秒が設定されています。goto コマンドの使用例も参照してください。

3.6.28 Title

title コマンドは、ウィンドウ・タイトルを必要な任意の文字列に変更するときに使用します。

使用例：

```
title: This is my window
```

このコマンドは、拡張エミュレータ・コマンドによってリモートで実行することもできます。

3.6.29 Wait For

wait for コマンドは、スクリプト・ファイルとホストから受信したデータを同期化します。

使用例：

```
wait for: USER>
```

このコマンド例では、“USER>”と完全に一致する文字列が到着するまで、スクリプト・ファイルが待機することになります（永久に待機する可能性もあります）。この特別なシーケンスは、USER ネームスペースにあるときの Caché ターミナルからの既定のプロンプトです。したがって、このコマンド例は、Caché ターミナルが次の入力を受け入れ可能になるまで待機する場合に使用できます。

wait for コマンドの終了には、タイマーを使用できます。

大文字/小文字の完全一致を有効または無効にする場合は、case match コマンドを参照してください。

4 拡張キーボード機能

4.1 キー・マッピング

Caché ターミナルでは、拡張キーボードに対して、次に示すアプリケーション・キーボード・モードがサポートされます。

キー	マップされた値
Num Lock	PF1
キーパッドの除算記号 (/)	PF2
キーパッドの乗算記号 (*)	PF3
キーパッドのマイナス記号 (-)	PF4
キーパッドのプラス記号 (+)	キーパッドのコンマ
Shift+キーパッドのプラス記号 (+)	キーパッドのマイナス記号 (-)
F1、F2、F3、F4	PF1、PF2、PF3、PF4 (それぞれのキーに対応)
Shift+F1 ...Shift+F10	F11 ...F20 (それぞれのキーに対応)

拡張キーボードのキーパッド部分は、次のようにマップされます。

キー	マップされた値
Insert	ここに挿入
Home	検索
Page Up	前の画面
Delete	削除
End	選択
Page Down	次の画面

Pause キーは、1つの XON/XOFF トグル・キーとして機能します。

4.2 キーボード・ファンクション

Caché ターミナルの操作性を向上させるために、いくつかの“ホット・キー”が定義されています。これらのホット・キーを使用すると、通常はメニューからアクセスする機能にすばやくアクセスできます。これらのホット・キーの説明は、対応するメニュー・コマンドの説明を参照してください。

トピック	要約
キーの時間計測	様々な負荷状況におけるホスト・システムのパフォーマンスを測定するのに役立ちます。キーの時間計測は、Alt-Shift-T で有効化または無効化できます。時間計測の“実行結果”の出力先は、システム・マネージャ・ディレクトリにある KEYTIMER.LOG という名前のファイルです。
学習モード	学習モードは、スクリプト・ファイルの迅速なプロトタイプ化を実現する機能です。学習モードは、Alt-Shift-L で有効化または無効化できます。学習モードが有効でロギングがアクティブな場合は、すべての受信文字が単純に収集されるのではなく、“ログ”ファイルが wait for と send スクリプト・コマンドの連鎖になります。このファイルは“ほぼ”再生可能です。wait for コマンドは、“sent”データに先行する文字を 16 文字まで表示します。
スパイ・モード	スパイ・モードは、Caché ターミナルの通信機能を利用する DDL アプリケーションの使用に役立ちます。スパイ・モードは、Alt-Shift-S で有効化または無効化できます。スパイ・モードは一般的には使用されないため、ここでは入力キーに関する情報のみを提供します。
一時停止モード	一時停止モードは、スクリプト・ファイルを手動で一時停止するときに使用します。一時停止モードは、Alt-P で有効化または無効化できます。また、pause コマンドで引数に 0 を指定して有効化することもできます。

5 DDE の概要

DDE は Dynamic Data Exchange の頭字語で、Macintosh、Windows、およびその他のオペレーティング・システムに組み込まれているプロセス間通信 (IPC) のことをいいます。DDE によって、実行中の 2 つのアプリケーションが同じデータを共有できるようになります。例えば、DDE によって、スプレッド・シートのグラフを、ワード・プロセッサで作成したドキュメントに挿入可能になります。スプレッド・シートのデータが変更されると、ドキュメントのグラフもそれに従って変更されます。DDE メカニズムは現在でも多くのアプリケーションによって使用されていますが、共有データをより高度に制御できる OLE (Object Linking and Embedding : Microsoft 社によって開発された複合ドキュメント規格) によって取って代わられつつあります。

Caché ターミナルは DDE (Dynamic Data Exchange) リンクをサポートすることで、他のアプリケーションがターミナルの通信機能を利用してリモート・ホストとやり取りすることを実現しています。

Caché ターミナルでは、以下の 3 つの (非システム) トピックがサポートされます。各トピックの説明は、ユーザに DDE の使用方法に関する知識があることを前提としています。トピックは以下のとおりです。

- ・ Layout : ステータス情報の取得に使用されます。例えば、行や列のサイズ、接続があるかどうかなどが取得されます。
- ・ Screen : ターミナル画面からのデータ収集に使用されます。
- ・ Message : ターミナル画面またはホストへのデータの送信に使用されます。

注釈: Windows タスクには、DDE の使用時に Caché ターミナルの複数インスタンスを区別する方法がありません。DDE の使用が最も役立つのは、実行されている Caché ターミナルが 1 つのときです。

5.1 DDE Layout 接続

Caché ターミナルは、Layout トピックを通じて、静的情報とみなされるものに対する DDE リクエストをサポートします。

アイテム	返り値の意味
Column	ウィンドウの列数。
Row	ウィンドウの行数。
hWnd	メイン・ウィンドウ・ハンドルの 10 進数の同等値。
Connected	接続がない場合は NULL 文字列、それ以外の場合はタイトル文字列の "mode: node" と同等値。
Read	最後に受信した文字が CTRL/A の場合は 1。VMS システムでは、"set terminal/script" によってすべての読み取りでこれを有効化できます。この使用目的は、画面描画の末尾の検出です。
Script	スクリプトが実行中の場合は 1、それ以外の場合は 0。
Title	ウィンドウのタイトル。

5.2 DDE Screen 接続

Caché ターミナルは、Screen トピックを通じて、画面データに対する DDE リクエストをサポートします。現在、対象とする画面行部分の選択には、1 つの POKE コマンドを使用できます。

アイテム	返り値の意味
Cursor	row;col 形式での現在のカーソル位置。
Line	現在の行 (CR LF を除く)。
LeftLine	現在行のカーソル位置より左の部分 (カーソル下の文字は含まない)。
RightLine	現在行のカーソル位置より右の部分 (カーソル下の文字を含む)。
All	画面全体 (各行は CR LF で改行される)。
Piece	現在選択されている画面行部分 (CR LF を除く)。

注釈: アイテム "Piece" は、"RnnCmmLpp" という形式の文字列を使用して POKE コマンドと同様の実行ができます。Piece の要求により、nn 行の mm 列から始まる (最大) pp 文字の文字列が取得されます。画面の左上隅は、行 1、列 1 になります。

5.3 DDE Message 接続

Caché ターミナルは、Message トピックを通じて、データ通信に対する DDE リクエストをサポートします。これらは DDE POKE コマンドによって実装されています。

アイテム	返り値の意味
Send	接続がアクティブな場合、DDE メッセージ値がホストに送信されます。
Display	DDE メッセージ値が、ホストから取得されたかのように "画面" に送信されます。

6 スクリプト例

このセクションでは、ローカル・マシン上で実行され、Windows 2000 の "バッチ" 機能を介して起動されるスクリプト・セッションの例を紹介します。

注釈: Microsoft Windows ではバージョンによって、キャレット文字 (^) とパーセント文字 (%) の解釈が異なります。また、ユーザによって "バッチ" ウィンドウ (DOS プロンプト) に入力されたか、バッチ・スクリプトの入力として読み取られたかによっても、テキストの特定行の解釈が異なります。

次の行を記述するのに必要な入力シーケンスの相違を、この後の表に示します。

```
cterm /console=cn_ap:cache[USER] ^%D
```

各オペレーティング・システムでの入力シーケンス

OS	環境	ルール	入力シーケンス
Windows NT Windows 2000 Windows XP	DOS プロンプト	二重キャレット (^) 文字	cterm /console=cn_ap:cache[USER] ^^%D
Windows NT Windows 2000 Windows XP	バッチ・ファイル	二重キャレット (^) 文字 二重パーセント (%) 文字	cterm /console=cn_ap:cache[USER] ^^%%D
Windows 9x Windows ME	DOS プロンプト または バッチ・ファイル	二重パーセント (%) 文字	cterm /console=cn_ap:cache[USER] ^^%%D

6.1 バッチ・コマンド行

ターミナル・アプリケーションは、Windows 2000 プログラムの cmd.exe を使用する “DOS” コマンド行から起動できます。コマンド行の一般的な形式は次の通りです。

```
cterm <Arg1> <Arg2> ... <ArgN> <ScriptFilePath>
```

各項目の内容は次の通りです。

アイテム	意味
cterm	ターミナル・アプリケーションの起動。Windows 環境変数の PATH に Caché バイナリの場所が設定されている場合は、コマンド名に “cterm” または “cterm.exe” を指定します。それ以外の場合は、完全なパス名または部分的なパス名を指定する必要があります。Caché の既定のインストールでのバイナリの場所は、C:¥CacheSys¥Bin ディレクトリです。
<ArgM>	ターミナルに適用する制御引数の 1 つ (個々の引数は次のセクションで説明)。
<ScriptFilePath>	ターミナル・アプリケーションによって解釈されるスクリプト・ファイルの場所。

6.2 制御引数

ターミナル・セッションの開始環境の制御に使用可能な引数がいくつかあります。引数の一部は、内部使用および/またはデバッグ用に予約されています。それらは、ここでは説明されません。一般的な起動時に役立つ引数には、以下のものがあります。

6.2.1 /console=<ConnectString>

この引数では、接続のタイプと、接続に必要なその他のデータの両方を指定します。重要な接続タイプは、TELNET 接続とローカル・コンソール・アプリケーションの 2 つです。

注釈: 引数の “/console” と “/server” は相互に排他的です。

/console=cn_iptcp:<HostAddr>

この引数は、ターミナルが TELNET 接続を介してやり取りするターゲット・システムを指定します。より一般的な使用法の 1 つは、ローカル・マシンでのスクリプトの実行です。この場合は、HostAddr にローカル・マシンの IP アドレスとポートを指定します。以下はその例です。

```
cterm /console=cn_iptcp:127.0.0.1[23]
```

/console=cn_ap:<Configuration>[<NameSpace>]:<Routine>

アプリケーション用の構成を指定することで、ターミナルでユーザ・アプリケーションを実行できます。Windows 2000 システムによって実行されるバッチ・ファイル (.BAT) に、次の行が記述されているとします。

```
cterm /console=cn_ap:cache[USER]:^^%D
```

この行はターミナル・セッション、および必要に応じて USER ネームスペースの “cache” 構成にある Caché バージョンを起動します。起動に成功すると、ターミナルによって、現在の日付を出力する ^%D ルーチンが実行されます。^%D が値を返したら、セッションはクローズされます。

構成は、システム・マネージャ・ディレクトリにある Caché パラメータ・ファイルの接尾語なしの名前に該当します。既定のパラメータ・ファイルは **cache.cpf** です。

ネームスペース名はオプションです。ネームスペース名を指定しない場合は、既定のネームスペース (%SYS) が使用されます。

“cn_ap” の後のコロンとネームスペース名を囲む角括弧は必須です。ルーチン名を指定する場合は “]” の後のコロンが必須で、指定しない場合は不要です。

6.2.2 /server=<ServerName>

この引数は、このターミナル・セッションと指定されたサーバ間での安全な接続に使用するサーバの名前を指定します。サーバ名は、Caché キューブ上で選択肢の 1 つとして使用可能な [優先接続サーバ] リストに定義されている必要があります。

接続の属性は、サーバ・エントリをこのリストに追加するときに定義されます。

注釈: 引数の “/console” と “/server” は相互に排他的です。

6.3 スクリプト 1 – プロセス情報の取得 (バッチ)

ここでは、基本的なデバッグ・ルーチンの ^%STACK を使用して、現在のユーザおよびターミナル・プロセスに関する情報を表示する例を紹介します。この例では、スクリプト・コマンドの格納場所が **C:\TestScript.scr** で、ユーザが DOS コマンド・ウィンドウに入力して実行することを想定しています。

```
C:\CacheSys\Bin\cterm.exe /console=cn_iptcp:127.0.0.1[23] C:\TestScript.scr
```

スクリプト自体の内容は以下のようになります。

```
;;;
;;; Script for Cache Terminal Example
;;;
; initialization
; turn match off to make comparisons more lenient
case match: off

; wait for the terminal to initialize
; and ask for our identification
echo: off
wait for:Username
send: SYS<CR>
wait for:Password
send: XXX<CR>
title: Terminal Example
echo: on

; log everything in a log
logfile: C:\TermExample.log
; wait a second
pause:10

; display a header to let the user know we are ready
; you need <CR><LF> because "display" does not
; have a prompt to advance to another line
display:<CR><LF>
display:-----<CR><LF>
display:<<< Cache Terminal Example >>><CR><LF>
display:-----<CR><LF>
; wait a second
pause:10

; switch to the USER namespace
send: znospace "USER"<CR>
wait for:USER>

; display some basic information about the system
; Use the debugging routine to do so
send: Do ^%STACK<CR>
wait for: action:

; have it outline our options
send: ?<CR>
wait for: action:

; wait 5 seconds for user to absorb
pause: 50

; ask for the basic process info
send: *s
pause: 50
send: <CR>
wait for: action:

; wait another 10 seconds
pause: 100

; finish the session
send: <CR>

; close the log file
closelog

; finished
terminate
```

6.4 例 2 – プロセス情報の取得 (インタラクティブ)

前の例に示したルーチンの `^%STACK` は、Windows 2000 DOS コマンドを介してターミナル・セッションで起動することもできます。

```
C:\CacheSys\Bin\cterm.exe /console=cn_ap:cache[USER]:^^%STACK
```

ユーザは、前の例でスクリプトが行っているレスポンスを手動で提供できます。ユーザが Enter キーを押して “Stack Display Action:” プロンプトに応答すると、ターミナル・ウィンドウがクローズされます。

6.5 例 3 – ホストへの手動接続

この例では、ターミナルのインスタンスを起動して、それをローカル・ホスト上の TELNET ポートに手動で接続し、コンソール・セッションを有効化します。この例は、既定のユーザ ID とパスワードを使用できることを前提にしています。

DOS ウィンドウからターミナルを起動します (Windows 2000 の場合)。

```
C:\CacheSys\Bin\cterm.exe
```

以下の手順を実行します。

- ・ ウィンドウ・メニュー・バーで [接続]、[ホスト] の順に選択します。
- ・ 表示されるダイアログ・ボックスで、[リモート・システム: アドレス] に “127.0.0.1” を、[ポート番号] に “23” を入力します。[OK] をクリックします。ターミナル・アプリケーションが TELNET ポートを介してローカル・ホストに接続しようとします。
- ・ Caché から “Username:” プロンプトが表示されます。“SYS” を入力して、Enter を押します。
- ・ “Password:” プロンプトが表示されます。“XXX” を入力して、Enter を押します。

すべての手順が問題なく終わると、“%SYS>” プロンプトが表示されます。これは、接続が完了し、%SYS ネームスペースにあることを意味します。例 2 と同じ操作を実行するには、最初にプロンプトへの応答として次のコマンドを発行します。

```
DO ^%STACK
```

次に、表示アクションのリクエストに対して、“*S” を使用して応答します。^%STACK ルーチンを終了するには、プロンプトへの応答として Enter を押します。

セッションを終了する場合は、[接続] メニューの [切断] を選択します。ターミナルの現在のインスタンスを終了する場合は、ウィンドウの [クローズ] ボックスを選択します。

