ターミナルの使用法

Version 2023.1
2024-01-02

InterSystems Corporation　1 Memorial Drive　Cambridge MA 02142　www.intersystems.com

ターミナルの使用法
InterSystems IRIS Data Platform　Version 2023.1　2024-01-02

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 Worldwide Response Center (WRC)
Tel:　+1-617-621-0700
Tel:　+44 (0) 844 854 2917
Email:　support@InterSystems.com

目次

テーブル一覧

このドキュメントについて

このドキュメントでは、InterSystems IRIS® データ・プラットフォームのコマンド行インタフェースである、ターミナルの使用方法について説明します。このドキュメントは、以下の章で構成されています。

- ターミナルの概要
- ターミナルのインタラクティブな使用法
- ターミナルの外観および動作の制御
- ターミナル・スクリプトの使用法
- スクリプト・コマンドのリファレンス
- バッチ・モードでのターミナルの使用法
- ターミナル使用に関するその他のトピック

詳細は、"目次" を参照してください。

1

ターミナルの概要

ターミナルは、ObjectScript コマンドを入力したり現在値を表示したりするためのシンプルなコマンド行インタフェースです。これは、学習、開発、およびデバッグのときに有用です。

注釈　ターミナルに **[スパイ・モードがオン]** というメッセージが示されたダイアログ・ボックスが表示された場合、これは、Alt-Shift-S が誤って押されたことを意味します。このモードを終了するには、Alt-Shift-S をもう一度押します。このモードは、一般的には使用されないため、ドキュメント化されていません。

さらに、ターミナルが応答しないように見える場合は、Ctrl-S が押されていて、自動スクロールが一時停止されている可能性があります。その場合は、Ctrl-Q を押して再開します。

1.1 ターミナル・プロセスを所有するユーザ・アカウント

プロセスは、Windows にログインしてターミナル・プログラム (**iristerm.exe**) を実行しているユーザによって所有されます。

また、すべての環境変数および共有ドライブ文字の指定は、ターミナルを実行しているユーザによって定義されます。

1.2 ターミナルの起動

ターミナルをインタラクティブに、またはバッチ・モードで使用できます。

ターミナルをインタラクティブに使用するには、次のいずれかを実行します。

- ローカル・データベースを使用してターミナルで作業するには、[InterSystems ランチャー] を選択し、次に **[ターミナル]** を選択します。
- リモート・サーバ上のデータベースを使用してターミナルで作業するには、[InterSystems ランチャー] を選択し、次に **[リモート・システム・アクセス]→[ターミナル]** を選択します。次にサーバ名を選択します。

  または、[InterSystems ランチャー] を選択し、**[リモート・システム・アクセス]→[Telnet]** を選択します。ユーザ名とパスワードを使用してログオンします。

  詳細および追加のオプションは、"システム管理ガイド" の "リモート・サーバへの接続" の章を参照してください。
- ターミナルを UNIX から起動するには、UNIX のコマンド・ラインで **iris terminal instancename** コマンドを実行します。

いずれの場合でも、ターミナル・ウィンドウが表示されます。表示される既定のプロンプトは、現在作業しているネームスペースを示しています。次に例を示します。

```
USER>
```

バッチ・モードでは、実行するスクリプト・ファイルの名前をオペレーティング・システムのコマンド行に渡して、そこからターミナルを起動します。このモードは、一部のオペレーティング・システムでは使用できません。

1.3 背景

ターミナルは、InterSystems IRIS® 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。ターミナルでは、InterSystems IRIS との通信に、ローカルとネットワークという 2 種類の方法が使用されます。タイトル・バーには現在使用されている通信モードが示されます。

- ターミナルがインストールされている InterSystems IRIS サーバとターミナルが通信するとき、ローカル通信が使用されます。この場合は、タイトル・バーに "InterSystems IRIS TRM:pid(instancename)" と表示されます。
  - pid は、ターミナルの通信先である InterSystems IRIS プロセスのプロセス ID です。
  - instancename は、プロセスが実行されている InterSystems IRIS インスタンスです。
- 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 InterSystems IRIS サーバまたは UNIX® ホストと通信します。この場合は、タイトル・バーに "(server NT — InterSystems IRIS Telnet)" と表示されます。server はリモート・サーバのホスト名です。

InterSystems IRIS の通信スタックは Winsock です。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。

1.4 一般的な使用

ターミナルでは、あらゆる ObjectScript コマンドを入力できます。ターミナルでは複数行にわたるコマンド入力がサポートされていないので、コマンド全体を 1 行に入力する必要があります。次に例を示します。

```
d ^myroutine

set dirname = "c:\test"

set obj=##class(Test.MyClass).%New()
write obj.Prop1
```

ただし、ターミナルで ZLOAD、ZINSERT、ZSAVE の各コマンドを実行するか、Tab キーの省略表現を使用することによって複数行のルーチンを定義できます。このようなルーチンは DO コマンドを使用して実行できます。

注釈　ターミナルでは、入力した各行の後に Use 0 コマンドが暗黙的に発行されます。これは、その他のデバイスへの直接出力に対して Use コマンドを発行した場合、このコマンドは基本的には無視されることを意味しています。

また、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試みるキー操作に遅延が生じることがあります。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特別な処理が、ホスト入力よりも前に実行されています。

ターミナル・スクリプトも実行できます。ターミナル・スクリプトとは、ファイル・システムに存在する拡張子 .scr が付いたファイルです。ターミナルには、これらのスクリプトで使用できる小規模なコマンド・セットが用意されています。このコマンド・セットには、手動でコマンドを入力したかのようにターミナルに ObjectScript コマンドを送信するコマンドなどが含まれます。

1.5 ZWELCOME ルーチン

ターミナルが実行を開始すると、コードは、%SYS ネームスペースに ZWELCOME という名前のルーチンが存在するかどうかをチェックします。該当するルーチンが見つかると、ターミナルのログイン・シーケンスがある場合は、その直前にそのルーチンを呼び出します。このルーチンは、その名前が意図するように、カスタムな識別情報やようこそメッセージをユーザに表示する目的で使用されます。

注釈　%SYS ネームスペースに ZWELCOME をインストールするには、管理者特権と IRISSYS データベースへの書き込みアクセスが必要です。

注意　ZWELCOME ルーチンは、空の $USERNAME および **%ALL** に設定した $ROLES を指定して、%SYS ネームスペースで実行します。使用の際は、ZWELCOME が失敗した場合でも影響が出ないようにする必要があります。また、このルーチンで $ROLES 変数を変更しないでください。

以下はその簡単な例です。

```
ZWELCOME() PUBLIC ;
    ; Example
    Write !

    Set ME = ##class(%SYS.ProcessQuery).%OpenId($JOB)

    Write "Now: ", $ZDATETIME($HOROLOG, 3, 1), !
    Write "Pid/JobNo: ", ME.Pid, "/", ME.JobNumber, !
    Write "Priority: ", ME.Priority, !

    Quit
```

1.6 実行開始ネームスペース

ターミナルを起動すると、特定のネームスペースで開きます。このオプションは、ユーザ定義の [**開始ネームスペース**] オプションで制御されます。″セキュリティ管理ガイド″ の “ユーザ” を参照してください。

既定のプロンプトには、次のような現在のネームスペースが表示されます。

```
USER>
```

1.6.1 ネームスペースの変更

新しいネームスペースに変更するには、$namespace 変数を使用します。

```
USER>set $namespace="SAMPLES"
SAMPLES>
```

詳細は、″ObjectScript リファレンス″ の ″$namespace″ のリファレンス・ページを参照してください。

1.7 ターミナル・プロンプト

ターミナルの既定のプロンプトは、現在作業しているネームスペースを示しています。トランザクション・レベルまたはプログラム・スタック・レベルを示すために、プロンプトに追加情報が表示される場合があります。SQL シェル、MDX シェル、またはオペレーティング・システム・シェルを使用する場合は、プロンプトが変更されます。既定のプロンプトをカスタマイズすることもできます。

1.7.1 トランザクション・レベル

トランザクション内で作業している場合、プロンプトには、トランザクション・レベルを示す接頭語が含まれます。接頭語の形式は TLn: です。n はトランザクション・レベルです。例えば、User ネームスペース内で作業している場合に、ObjectScript コマンド TSTART を入力すると、プロンプトは以下のように変更されます。

```
USER>tstart
TL1:USER>
```

ターミナルを終了すると、これによりトランザクションがロールバックされます。

1.7.2 プログラム・スタック・レベル

エラーが発生した場合、プロンプトには、プログラム・スタック・レベルを示す接尾語が含まれます。次に例を示します。

```
USER 5d3>
```

Quit コマンドを入力して、デバッグ・プロンプトを終了します。または、エラーをデバッグします。"ObjectScript の使用法" の "コマンド行ルーチンのデバッグ" の章を参照してください。

1.7.3 SQL シェルと MDX シェル

ターミナルでは、さまざまな方法でデータのクエリを実行できるシェルを使用することができます。これらのいずれかのシェルを使用する場合、プロンプトは以下の表に示すように変化します。

シェル	プロンプトの変化	関連項目
SQL	[SQL] 接頭語が追加される	"InterSystems SQL の使用法" の "SQL シェル・インタフェースの使用法"
TSQL	[SQL] 接頭語が追加される	"Transact-SQL (TSQL) 移行ガイド" の "概要"
MDX	プロンプトが >> になる	"InterSystems IRIS Business Intelligence の概要" の "MDX シェル"

1.7.4 オペレーティング・システム・シェル

ターミナルでは、既定のオペレーティング・システム・シェルを開くこともできます。シェルを開くには、! または $ と入力して、Enter キーを押します。これにより、ターミナルで既定のオペレーティング・システム・シェルが開き、プロンプトに作業ディレクトリが表示されます。次に例を示します。

```
USER>!
c:\intersystems\iris\mgr\user\>
```

注釈　Macintosh では、C シェルをこの方法で開くことはできません。アクセス拒否のエラーが表示されます。ただし、他のシェル (Bash、Bourne、または Korn) は使用できます。

シェルを終了するには、そのシェルに該当する `quit` コマンドか `exit` コマンドを使用します。

1.7.5 ターミナル・プロンプトのカスタマイズ

既定のターミナル・プロンプトをカスタマイズするには、管理ポータルを使用して、ターミナル・プロンプトを制御する構成オプション TerminalPrompt を設定します。このオプションは管理ポータルで使用できます。これを見つけるには、**[システム管理]→[構成]→[追加設定]→[開始]** を選択します。例えば、ターミナル・プロンプトには、使用するシステムの構成名を含めることができます。

1.8 ターミナルでの実行の中断

ターミナルを中断して、フォアグラウンドのすべての実行内容を停止するには、次のいずれかのキー組み合わせを使用します。

- Ctrl-C ― **[Windowsエディットアクセラレータ]** オプションが有効になっていない場合に使用します。
- Ctrl-Shift-C ― **[Windowsエディットアクセラレータ]** オプションが有効になっている場合に使用します。

[Windowsエディットアクセラレータ] オプションの詳細は、“ターミナルの外観および動作の制御” の章の “ユーザ設定” セクションを参照してください。

1.9 ターミナルの終了

ターミナルを終了するには、以下のいずれかの操作を実行します。

- **[ファイル]→[終了]** を選択します
- Alt-F4 を押します
- HALT または H と入力します (大文字/小文字の区別なし)

これらのオプションにより、ターミナルの現行のコピーが終了し、開いているすべてのファイルが閉じられて、フォアグラウンドのすべての実行内容が停止されます。

このターミナルが起動時にサーバに接続されていた場合は、通信チャネルが閉じたときに自動的に終了します。

InterSystems ランチャーの **[InterSystems Telnet]** を使用してこのターミナルにアクセスした場合、ターミナルは通信チャネルが閉じたときに自動的に終了せず、アクティブ状態のままになるため、**[接続]** メニューを使用して再接続できます。

2

ターミナルのインタラクティブな使用法

この章では、インタラクティブ・モードでターミナルを操作する基本的な方法について説明します。

2.1 コマンドの入力

ターミナルでコマンドを入力するには、ObjectScript コマンドとその引数を入力します。または、シェルを使用している場合、そのシェルに適用できるコマンドを入力します。

ObjectScript の場合、コマンドの前にスペースを入力する必要はありません (ルーチンやメソッドを定義する場合とは対照的です)。

また、Tab キーを押してターミナルで行を開始する場合、その行の残りは、ルーチン・バッファに入力され、実行されません。この動作は以前に定義され、ルーチンを作成する便利な方法です (同じ目的で機能する ZINSERT コマンドも参照してください)。誰もこの方法でルーチンを定義しなくなったため、このドキュメントではこの機能について詳しく説明しません。

2.2 スクロール

アクティブなテキストが到着したときは常に、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スクロールされます。右のスクロール・バーを使用して、上下にスクロールします。

以下のように、キーボードを使用して、ターミナル内をスクロールすることもできます。

テーブル 2-1: キーボードを使用したターミナル表示のスクロール

キーの組み合わせ	結果
Ctrl-Home	バッファの先頭にスクロールします。
Ctrl-End	カーソルまで下方にスクロールします。
Ctrl-PageUp	1 ページ単位で上方にスクロールします。
Ctrl-PageDown	1 ページ単位で下方にスクロールします。
Ctrl-LineUp	1 行ごとに上方にスクロールします。
Ctrl-LineDown	1 行ごとに下方にスクロールします。

2.3 ターミナルのスクロールの一時停止と再開

ターミナルのスクロールを一時停止するには、Ctrl-S を押します。スクロールが一時停止されている間、ターミナルは、コマンドを受け入れて処理しますが、画面にコマンドや出力を書き込むことはありません (そのため、応答しないように見えます)。

再開するには、Ctrl-Q を押します。

2.4 前のコマンドの繰り返し

前のコマンドを繰り返すには、そのコマンドが表示されるまで上矢印キーを繰り返し押します。コマンドを入力するには、通常どおり Enter キーを押します。

または、コマンド :history (または :h) を発行すると、現在のターミナル・セッションで発行したコマンドの番号付きリストを取得できます。このリストからコマンドを繰り返すには、コマンド `:<number>` を発行します。〈number〉 はリスト上のコマンドに対応する番号です。

コマンド :clear を実行すると履歴バッファがクリアされます。したがって、上記の方法で前のコマンドを呼び出すことができなくなります。

2.5 頻繁に使用するコマンドのエイリアス作成

コマンドを発行するためのショートカットを定義するには、:alias コマンドに続けて、コマンドに割り当てる別名 (エイリアス)、そのエイリアスに割り当てるコマンドの順に入力します。ここから以降の現在のターミナル・セッションでは、: 文字の後にエイリアスを入力することで該当のコマンドを発行できます。

次のセッション例では、1 行目で :alias コマンドを発行することにより、2 行目で文字列 `:u` を入力するだけで、ネームスペースを USER ネームスペースに切り替えることができます。

```
%SYS>:alias u set $namespace="USER"
%SYS>:u
USER>
```

エイリアスを使用してコマンドに引数を渡すこともできます。エイリアス定義の一部として、先頭に $ 文字を使用した番号付きのホスト変数を、コマンドの各引数のプレースホルダとして指定します。エイリアスに続けてホスト変数の番号の順序で引数の値を指定し、エイリアスを起動します。

例えば、以下のコマンドを発行すると、ファイルをロードするショートカット (load) が作成されます。ファイルのある場所を 1 番目の引数 ($1) で指定し、2 番目の引数 ($2) でフラグを指定します。

```
USER>:alias load Do $System.OBJ.Load("$1","$2")
```

このエイリアスを使用して、パス /directory/foobar.xml にある、c と k のフラグを設定したファイルをロードするには以下を入力します。

```
USER>:load /directory/foobar.xml ck
```

コマンドのプレースホルダ変数に ObjectScript 式の一部を使用することもできます。例えば、以下のコマンドでは $SELECT 関数を使用して、2 番目の引数を指定しない場合の既定の 2 番目の引数として "ck" を設定します。

```
USER>:alias load Do $System.OBJ.Load("$1", $S("$2"="":"ck", 1:"$2"))
```

エイリアス定義を削除するには、:unalias コマンドを、それに続けてエイリアス名を指定して入力します。引数を指定せずに :alias を入力すると、現在のターミナル・セッションで定義されているエイリアスが一覧表示されます。

UNIX® または Linux のシステムを使用している場合は、セッションの開始ごとにターミナルで自動的に設定されるエイリアス定義を列挙したリストを指定できます。ホーム・ディレクトリにあるファイル .iris_init に、これらのエイリアスを 1 行に 1 つずつ定義します。

2.6 テキストのコピーと貼り付け

ターミナルでテキストをコピーして貼り付けることができます。この操作を実行するには、コンテキスト・メニュー (右クリック・メニュー)、[編集] メニュー、または各種のキーボード・ショートカットを使用できます。以下のオプションを使用できます。

- **コピー**— 選択されているテキストを、クリップボードにコピーします。
- **貼り付け**— クリップボードのコンテンツをカーソルの現在位置 (ターミナルのスクロールバック・バッファの最後) に行ごとに貼り付けます。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。
- **コピー + 貼り付け**— 選択したテキストをクリップボードにコピーして、カーソルの現在位置に行ごとに貼り付けます。

2.6.1 キーボードによるショートカット

使用できるキーボードによるショートカットは、以下のとおりです。

テーブル 2-2: コピーおよび貼り付けのショートカット

アクション	基本のショートカット	Windows のショートカット
コピー	Ctrl-Insert	Ctrl-C
貼り付け	Shift-Insert	Ctrl-V
コピーと貼り付け		Ctrl-Shift-V

“基本のショートカット” 列に示したショートカットは、常に有効です。

“Windows のショートカット” 列に示したショートカットは、[Windows edit accelerators] オプションを [はい] に設定した場合にのみ有効になります。この設定の詳細は、“ターミナルの外観および動作の制御” の章の “ユーザ設定” のセクションを参照してください。

2.6.2 コピーと貼り付けに関する注記

- 前述のように、[Windowsエディットアクセラレータ] オプションを [はい] に設定すると、Ctrl-C によって、選択したテキストが Windows クリップボードにコピーされます。ターミナルを中断するには、代わりに Ctrl-Shift-C を押す必要があります。
- 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。
- 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。改行を貼り付けない場合は、“ユーザ設定” を参照してください。
- 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。

2.7 印刷

ターミナルから印刷するには、[ファイル] メニューの以下のオプションを使用します。

- プリンタを選択して、ターミナルでそのプリンタを使用するように設定するには、[ファイル]→[印刷設定] を選択します。
- ターミナル画面の内容を印刷するには、[ファイル]→[印刷] を選択します。
- ログ・ファイル (または他の ASCII ファイル) を印刷するには、[ファイル]→[ログの印刷] を選択します。このオプションでは印刷するファイルを選択でき、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。

2.8 画面のクリア

[編集] メニューには、画面をクリアするための 2 つの異なるオプションが用意されています。

- 画面をリセットするには、[編集]→[リセット] を選択します。このオプションを使用すると、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、ターミナルでウィンドウが再描画されます。
- 画面を再初期化するには、[編集]→[削除] を選択するか、Ctrl-Del を押します。このオプションは、ターミナル・ウィンドウを再初期化し、すべてのセッション・データを消去し、スクロールバック領域を 0 にリセットします。

2.9 ターミナル・セッションのログへの記録

ターミナル・セッションのログ記録を開始するには、以下の手順を実行します。

1. **[ファイル]→[ログ]** を選択するか、Alt-L を押します。

   ターミナルは、ログ・ファイルの場所と名前の入力を促すダイアログ・ボックスを表示します。既定のディレクトリは、install-dir/mgr です。既定のファイル名は TERMINAL.LOG です。

   パスとファイル名の合計長は 126 文字を超える長さにはできません。

2. 必要な場合は、別のディレクトリとファイル名を指定します。

3. [OK] をクリックします。

   ログ・ファイルが存在する場合は、ターミナルでログ・ファイルを上書きするかどうか尋ねるメッセージが表示され、次の 3 つの選択肢が提示されます。

   - **[はい]** は、新しいログ・データでファイルを上書きします。
   - **[いいえ]** は、新しいログ・データをファイルに追加します。
   - **[キャンセル]** は、ファイルをそのまま残します (ロギングは行われません)。

その後、ログへの記録を終了するには、**[ファイル]→[ログ]** を選択するか、Alt-L を押します。ログ・ファイルが閉じられたことを示すダイアログ・ボックスがターミナルに表示されます。[OK] を選択します。

ログ・ファイルには、接続からの出力のみが記録されます (現行のラップ・モードには関係ありません)。

このドキュメントで後述するように、ターミナル・スクリプトからもロギングを実行できます。**[ファイル]→[ログ]** によりロギングを開始した場合、ロギングも実行するスクリプトは開始できないことに注意してください。そのようにした場合の動作は不確定です。

“ターミナル使用に関するその他のトピック”の章の“学習モード”のセクションも参照してください。

3

ターミナルの外観および動作の制御

この章では、インタラクティブ・モードでターミナルの外観と動作を制御するさまざまな方法について説明します。

Tipヒント　ここにリストされているオプションの他に、構成オプション TerminalPrompt を使用できます。このオプションはターミナル・プロンプトを制御します。この値は管理ポータルで使用できます。これを見つけるには、**[システム管理]→[構成]→[追加設定]→[開始]** を選択します。例えば、ターミナル・プロンプトには、使用するシステムの構成名を含めることができます。

3.1 フォントの指定

使用するフォント・サイズを指定するには、**[編集]→[フォント]** を選択します。これによりダイアログ・ボックスが表示されます。このダイアログ・ボックスで、使用しているモニタと解像度に適した書体、サイズ、およびスタイルを選択できます。

注釈　画面の枠を越えてウィンドウが拡張するようなフォント・サイズを選択した場合は、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。

また、異なるサイズ画面に切り替えたときは常に、そのサイズに対して事前選択されているフォントが使用されます。

3.2 色の指定

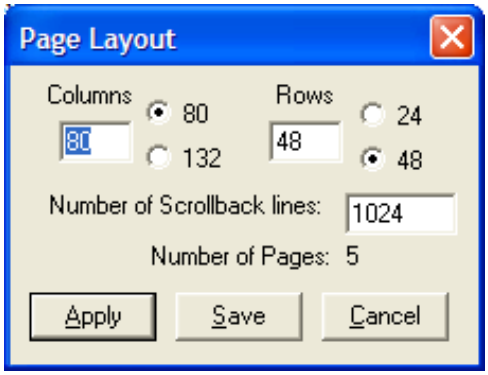
使用する色を指定するには、**[編集]→[色]** を選択します。これにより、ターミナルの既定の前景色と背景色を選択できるダイアログ・ボックスが表示されます。以下のいずれかを選択します。

- **[適用]** をクリックすると、現在のターミナルのみが変更されます。
- **[保存]** をクリックすると、現在のインスタンスは変更されず、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。

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調整できます。これらの色は、前景色および背景色と共に保存されます。既定の色を選択するには、**[デフォルト]** を選択してから色を選択します。

3.3 ウィンドウ・サイズの指定

ターミナル・ウィンドウのサイズを指定するには、**[編集]→[ウィンドウサイズ]** を選択します。これにより、ウィンドウ・サイズを指定できるダイアログ・ボックスが表示されます。



列の最大数は 132 で、行の最大数は 64 です。変更を行うと、ダイアログ・ボックスでは、使用可能なスクロールバックの行数とページ数が更新されます。以下のいずれかを選択します。

- **[適用]** をクリックすると、現在のターミナルのみが変更されます。
- **[保存]** をクリックすると、現在のインスタンスは変更されず、ターミナルの新規インスタンス用にウィンドウ・サイズ情報が保存されます。

注釈　ウィンドウのサイズを変更すると、ターミナルでは、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、そのフォントが選択されます。

3.4 カスタム・キーの組み合わせの定義

カスタム・キーの組み合わせを定義するには、**[編集]→[ユーザキー]** を選択します。これによりダイアログ・ボックスが表示されます。このダイアログ・ボックスでは、ObjectScript コマンドを、Alt-Shift-F1 から Alt-Shift-F10 までのいずれかのキーの組み合わせと関連付けることができます。

[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。

出力できない文字をコマンドに含めるには、その文字と同等の 10 進数値 (nnn) を使用します。また、<CR>、<F10>、<F7>、<DO>、<TAB>、<LF>、<ESC>、<CSI>、<NL> (= <CR><LF>) のいずれかを使用できます。

<P1>、<P2> などのコマンド行パラメータも使用できます。

注釈　[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。

3.5 その他のユーザ設定の指定

ユーザ設定を指定するには、[編集]→[ユーザ設定] を選択します。これによりダイアログ・ボックスが表示されます。このダイアログ・ボックスでは、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指定できます。この設定は以下のとおりです。

テーブル 3-1: ユーザ設定

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改行を (キャリッジ・リターンと共に) 送信するように設定します。
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。
Windows edit accelerators	基本的なターミナルの編集ショートカット (Ctrl-Insert と Shift-Insert) に加えて、ターミナルで一般的な Windows の編集ショートカット (Ctrl-C、Ctrl-V、Ctrl-Shift-V) を有効にするかどうかを指定します。 これらの Windows ショートカットが有効化されていない場合は、文字がデータ・ストリームで InterSystems IRIS サーバ・プロセスに渡されます。
Paste burst size (in bytes)	一度の送信に貼り付け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、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指定し、[Paste pause time] で送信間の一時停止時間を指定します。いずれかの設定が 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。

3.6 ネットワーク・エンコードの指定

ターミナルのネットワーク・エンコードでは、以下の場合に文字を変換する方法を制御します。

- キーボード入力が、Unicode を使用してターミナルのディスプレイ・メモリに変換される場合。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、複数言語の混在入力が可能となり、混在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。複数言語を混在入力する場合は、ネットワーク・エンコードおよび InterSystems IRIS $ZMODE 入出力変換テーブルとして [UTF8] を選択します。

- ターミナルがピア・サーバと通信する場合。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。

既定のネットワーク・エンコードは UTF8 です。

ネットワーク・エンコードを指定するには、**[編集]→[ネットワーク・エンコーディング]** を選択します。これにより、ターミナルで使用するネットワーク・エンコードを選択できるダイアログ・ボックスが表示されます。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。

3.6.1 UTF8 エンコード

UTF8 オプションを選択すると、ターミナルで、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 UTF8 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 InterSystems IRIS 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。円記号 (\) で区切られた 4 番目のフィールドにあります。

3.6.2 Windows エンコード

Windows オプションを選択すると、ターミナルでは、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に、現行の Windows 入力コード・ページが使用されます。Windows エンコードを使用するときは、InterSystems IRIS 入出力変換 ($ZMODE) を、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する必要があります。

3.6.3 ISO エンコード

ISO オプションを選択すると、ターミナルでは、ピア・サーバとの入出力変換に、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。

テーブル 3-2: ISO エンコード

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、$ZMODE で表示される InterSystems IRIS 入出力変換が、ターミナルによって使用されるアクティブな ISO コード・ページで示す文字セットと矛盾しないように確認する必要があります。

3.6.4 EUC エンコード

EUC エンコードは、極東地域の言語に関連するエンコードであり、一部の UNIX® システムとの通信に使用します。EUC オプションを選択すると、ターミナルでは、サーバとの入出力変換に、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。

テーブル 3-3: EUC エンコード

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。

3.7 表示の物理文字設定の指定

物理文字設定を指定するには、**[編集]→[物理文字表示の設定]** を選択し、**[論理]** または **[物理]** を選択します。このオプションでは、ターミナル・ウィンドウに表示される文字の形態を制御できます。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。

4

ターミナル・スクリプトの使用法

この章では、ターミナルでスクリプト・ファイルを作成して使用する方法について説明します。

また、次の章“スクリプト・コマンドのリファレンス”、および“ターミナル使用に関するその他のトピック”の章の“学習モード”のセクションも参照してください。

注釈　ターミナル・スクリプトが役立つ場合もありますが、通常はルーチンを作成して使用する方が（各ルーチン・プログラミング言語が非常に豊富なオプション・セットを提供しているので）はるかに簡単です。

ユーザ環境変数および共有ドライブ文字の指定は、InterSystems IRIS コントロール・サービスを実行するユーザ・アカウントによって定義されます。

4.1 スクリプトの実行の開始

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が、任意の場所に配置可能です。

スクリプトを実行するには、**[ファイル]→[スクリプト]** を選択するか、**Alt-S** を押します。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・モードをロックしてい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確立され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、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確立後まで延期されます。

4.2 スクリプトの実行の一時停止

スクリプトの実行を停止するには、**[ファイル]→[一時停止]** を選択するか、**Alt-P** を押します。現在のスクリプトを一時停止することを確認するプロンプトが表示されます。

4.3 スクリプトの実行の停止

スクリプトを停止するには、[ファイル]→[スクリプト] を選択するか、Alt-S を押します。現在のスクリプトを停止することを確認するプロンプトが表示されます。

4.4 スクリプト・ファイルのコンテンツ

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見なされます。読みやすさを向上させるために空白行を自由に使用できます。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形式は以下のとおりです。引数は文字列または数値と解釈されます。

```
ScriptCommand: ScriptArguments
```

ここで ScriptCommand は、ターミナル・スクリプト・コマンドの 1 つで、ScriptArguments は、そのコマンドの引数リストです (具体的なコマンドの詳細を参照)。スクリプト・コマンドが 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。また、コマンドとコロンの間にはスペースを入れません。

引数のないコマンドを以下に示します。

```
ScriptCommand
```

制御の移動箇所の定義には、ラベルを使用できます。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。

“ターミナル使用に関するその他のトピック”の章の“学習モード”のセクションも参照してください。

4.5 スクリプト・コマンドの概要

次のテーブルは、使用可能なスクリプト・コマンドの一覧です。

テーブル 4-1: ターミナル・スクリプト・コマンド

コマンド	アクション
break	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。
call script	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。
case match	″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。
closelog	ログ・ファイルを閉じます。
connect	ホストに未接続の場合に、ホスト接続を強制します。
debug	スクリプトのデバッグを有効化または無効化します。
disconnect	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。
display	ディスプレイにテキストを送信します。
echo	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
execute	Windows プログラムを実行します。
exit	スクリプトを終了します。
goto	制御をスクリプト内の別の場所に移します。
if empty	最後のテスト文字列が空の場合に、制御を移動します。
key_starttime	キーの時間計測の開始をシミュレートします。
key_stoptime	キーの時間計測の停止をシミュレートします。
key_timer	キーの時間計測のオン/オフを切り替えます。
logfile	ログ・ファイルを開始します。
multiwait for	通信デバイスからの任意の複数文字列を待機します。
notify	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。
on error	タイマが起動された場合の分岐先ラベルを指定します。
pause	スクリプトを一時停止します。
return	スクリプト・ファイルのサブルーチンから戻ります。
send	通信デバイスにテキストを送信します。
subroutine	スクリプト・ファイル内のサブルーチンを呼び出します。
terminate	エミュレータを完全に終了します。
test	テスト対象の文字列を構築します。
timer	"wait for" コマンド用のタイマを制御します。
title	ウィンドウ・タイトルを設定します。
wait for	通信デバイスからの特定の文字列を待機します。

これらのコマンドのリファレンス情報は、後述の“スクリプト・コマンドのリファレンス”を参照してください。

4.6 スクリプト・コマンドの引数

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>

これで、<Pn> が、n 番目のコマンド行パラメータに置換されます。

操作を簡略化するために、特定の ASCII 文字には、この後のテーブルに示す同等のショートカット表記が用意されています。

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。

テーブル 4-2: 特殊文字

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>

4.7 スクリプト例

以下は、ターミナル・スクリプトのサンプルです。

```
; 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:<<< Terminal Example >>><CR><LF>
display:------------------------------<CR><LF>
; wait a second
pause:10

; switch to the USER namespace
send: set $namespace="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
```

5
スクリプト・コマンドのリファレンス

この章では、ターミナル・スクリプト・コマンドのリファレンス情報を示します。

注釈　ターミナル・スクリプトが役立つ場合もありますが、通常はルーチンを作成して使用する方が（各ルーチン・プログラミング言語が非常に豊富なオプション・セットを提供しているので）はるかに簡単です。

5.1 break

ブレークをサポートする通信ノードにブレークを送信します。それ以外には何も実行しません。引数は取りません。使用例は以下のとおりです。

```
break
```

5.2 call script

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、ターミナルで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。使用例は以下のとおりです。

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

この例は、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。サンプル・スクリプト（login.scr）の最初のパラメータは `fred` です。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。既定のファイル拡張子は .scr と見なされます。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。

5.3 case match

wait for コマンドで大文字/小文字の照合を有効化または無効化します。使用例は以下のとおりです。

```
case match: off
```

大文字/小文字の照合を無効化すると、個々の文字の大文字/小文字が異なっていても、文字列が一致していると見なされます。このスイッチの既定は on です。

5.4 closelog

現在開いているログ・ファイルがあれば、それを閉じます。使用例は以下のとおりです。

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

5.5 connect

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。使用例は以下のとおりです。

```
connect
```

5.6 debug

デバッグ・モードを有効にします。デバッグ・モードでは、通常はターミナルが無視する無効なスクリプト・コマンドをトラップします。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。使用例は以下のとおりです。

```
debug: on
```

5.7 disconnect

ホストとの接続を切断します。未接続の場合は、何の動作もしません。使用例は以下のとおりです。

```
disconnect
```

5.8 display

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。使用例は以下のとおりです。

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

この例を実行すると、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:` というテキストが出力され、テキストの末尾にカーソルが置かれます。

5.9 echo

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。これは、出力を非表示にする必要がある場合 (ユーザにとって情報価値がないためなど) に役立ちます。例は、″wait for″ を参照してください。

5.10 execute

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。使用例は以下のとおりです。

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

この例では、Windows のメモ帳プログラムを起動し、そのアプリケーション内でファイル myfile.not を開きます。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。

5.11 exit

スクリプトを終了します。通常、スクリプトは最後の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。使用例は以下のとおりです。

```
    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
```

5.12 goto

制御をスクリプト・ファイル内の別の場所に移します。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。使用例は以下のとおりです。

```
    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 ...
```

5.13 if empty

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。使用例は以下のとおりです。

```
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。

5.14 key_starttime

キー・シーケンスの時間計測を開始します。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 を押したときに、統計が蓄積されます。それ以外の場合は、F10 を押したときに、統計が蓄積されます。使用例は以下のとおりです。

```
key_starttime: 0
```

時間計測を停止するには、key_stoptime コマンドを使用します。

5.15 key_stoptime

時間計測が現在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。使用例は以下のとおりです。

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

5.16 key_timer

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。または、Alt-Shift-T を使用して、タイマを開始または停止できます。使用例は以下のとおりです。

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

ファイル (KEYTIMER.LOG) がシステム・マネージャ・ディレクトリに作成され、キーの時間計測のヒストグラムが記録されます。時間計測シーケンスは現行の統計ファイルに追加されるのではなく上書きされるため、使用可能な時間計測シーケンスは 1 つのみです。

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には、<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。

5.17 logfile

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。使用例は以下のとおりです。

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

既定のディレクトリはスクリプトの存在するディレクトリとなります。これはフル・パス名を与えることで変更できます。

通常、ログ・ファイルは上書き方式で開きます。つまり、ログ・ファイルが既に存在する場合は、新規データで既存のものが置換されます。

5.18 multiwait for

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、引数に指定された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。使用例は以下のとおりです。

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

この例では、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機能します。したがって、このコマンドは、次のいずれかのシーケンスが到着するまでサンプル・スクリプトを待機させます。

```
USER>
***ERROR,
DONE
```

タイマを使用することで、このコマンドを終了できます。

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>
```

5.19 notify

Windows メッセージ・ボックスを表示し、ユーザが [OK] ボタンを押すまで待機します。スクリプトを実行するユーザへのメッセージに使用できます。使用例は以下のとおりです。

```
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あるため、他のユーザが割り込むことはできません。

5.20 on error

タイマが時間切れになった場合（通常はテキストの到着を待っている最中）に実行される暗黙の goto のターゲット・ラベルを設定します。厳密な正確さを確保するには、このコマンドは timer の使用前に使用する必要がありますが、実際にはこの順序は重要ではないことがあります。

例は、"wait for"、"exit"、"goto"、および "subroutine" を参照してください。

また、"timer" コマンドも参照してください。

5.21 pause

実行中のスクリプトを一時停止します。停止時間は 10 分の 1 秒単位で指定します。使用例は以下のとおりです。

```
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。引数が 0 の場合は永久停止となります。永久停止から再開するには、Alt–P を使用します。

5.22 return

subroutine コマンドと共に使用して、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。使用例は、subroutine コマンドを参照してください。

5.23 send

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。使用例は以下のとおりです。

```
send: set $namespace="%SYS"<CR>
```

この行はネームスペースを %SYS に変更します。send コマンドはキャリッジ・リターンを暗黙的に追加しないため、末尾の <CR> が必要です。

もう 1 つの使用例は以下のとおりです。

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

このコマンドは、以下の順で入力したことと同等です。

- 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
- A1234 という文字列

- F10 キー
- 1 つの空白文字

<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。普通に入力した場合、これらの文字はコマンド・インタープリタによって削除されます。

重要　wait for コマンドを各 send コマンドの後ろに組み込むことをお勧めします。wait for コマンドにより、スクリプト内のコマンドをターミナルからの入力と後で同期させることができます。ターミナル・スクリプト・メカニズムは、wait for コマンドを検出した場合を除き、InterSystems IRIS から返される入力に関係なく、コマンドを順次送信します。

wait for コマンドを各 send コマンドの後ろに組み込まず、ログ・ファイルを生成する場合、そのログ・ファイルには後続の send コマンドの情報は含まれません。

5.24 subroutine

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。このコマンドを使用すると、記憶域を節約できると同時に、異なる多くのラベルを保持する必要がなくなります。このコマンドは return コマンドと共に使用します。使用例は以下のとおりです。

```
    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。それより深くサブルーチンをネストしようとすると、スクリプトは失敗します。

5.25 terminate

終了して Windows に戻るようターミナルに命令します。開いているすべてのファイルが閉じ、不要なウィンドウが消去され、接続が閉じます。使用例は以下のとおりです。

```
    terminate
```

5.26 test

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。使用例は、if empty コマンドを参照してください。

5.27 timer

wait for コマンドと連携して使用されるタイマを設定します。timer コマンドは、Windows の SetTimer() コマンドを実行します。このタイマが始動すると、スクリプト・プロセッサは on error で指定されたラベルに移動します (指定されている場合)。このスクリプトは、on error によってラベルが指定されていない場合には直ちに終了します。

使用例は以下のとおりです。

```
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。

タイマをオフにするには、次のコマンドを使用します。

```
timer: 0
```

timer の使用例は、"wait for" を参照してください。

5.28 title

ターミナル・ウィンドウのタイトルを指定された文字列に設定します。使用例は以下のとおりです。

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

タイトルの設定は、拡張エミュレータ・コマンドを使用してリモートで実行することもできます。

5.29 wait for

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。使用例は以下のとおりです。

```
wait for:  USER>
```

この例では、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ターミナルからの既定のプロンプトです。つまり、このコマンドは、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。

timer を使用することで wait for コマンドを終了できます。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。timer を使用する際は、そのタイマが時間切れになった場合にフローを受け取るラベルを指定でき

ます (″on error″ を参照してください)。wait for で検索対象のテキストが見つかった場合は、タイマが強制終了されて、on error によって設定されたラベルがクリアされます。使用例は以下のとおりです。

```
    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
```

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、[OK] を選択すると終了します。

wait for コマンドでテキストの大文字と小文字が区別されるかどうかは、case match コマンドの使用状況によって異なります。

6

バッチ・モードでのターミナルの使用法

一部のオペレーティング・システムでは、コマンド行 (DOS ウィンドウなど) からターミナルを実行できます。

6.1 バッチ・コマンド行

Windows システムでは、DOS のコマンド行 (正確に言えば、cmd.exe) からターミナルを起動することができます。コマンド行の一般的な形式は次のとおりです。

```
iristerm Arg1 Arg2 ... ArgN ScriptFilePath
```

以下はその説明です。

テーブル 6-1: バッチ・モードでのターミナルの起動

アイテム	意味
`iristerm`	ターミナル・アプリケーションを起動します。Windows 環境変数の PATH に InterSystems IRIS バイナリの場所が設定されている場合は、コマンド名 `iristerm` または `iristerm.exe` を使用します。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使用する必要があります。InterSystems IRIS の既定のインストールでのバイナリの場所は、install-dir ディレクトリです。¥Bin
Arg1 ... ArgN	制御引数 (次のセクションで説明します)。
ScriptFilePath	スクリプト・ファイルの保存場所。

6.2 制御引数

ターミナル・セッションの開始環境を制御する引数は数種類あります。これらのいくつかは内部用に予約されているため、ここでは説明しません。最も有用な引数は以下のとおりです。

注釈　ターミナルを起動する際に /console または /server という制御引数を指定した場合は、[接続] メニュー項目は表示されません。

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。接続タイプは、TELNET 接続とローカル・コンソール・アプリケーションへの接続の 2 つです。

注釈　/console および /server 引数は両方とも指定できません。

6.2.1.1 /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。これは、ローカル・マシン上でスクリプトを実行するのに便利です。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。次に例を示します。

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

6.2.1.2 /console=cn_ap:<Instance>[<Namespace>]

ターミナル・セッションを開くことができます。以下はその例です。

```
iristerm /console=cn_ap:iris[USER]
```

この行により、ターミナル・セッションが開始され、特定のネームスペースに切り替えられます。

この場合、インスタンス名は iris で、ネームスペース名は USER となります。

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が使用されます。

6.2.1.3 /console=cn_ap:<Instance>[<Namespace>]:<Routine>

バッチ・ファイルからルーチンを実行することもできます。例えば、Windows 7 で使用されるバッチ・ファイル (.bat) に、以下の行が記述されているとします。

```
iristerm /console=cn_ap:iris[USER]:^^%%D
```

この行により、(未実行の場合) 指定されたインスタンスが開始され、ターミナル・セッションが開かれた後に、指定されたネームスペースへ切り替わり、指定されたルーチンが実行されます。ルーチンが終了したら、ターミナル・セッションは閉じます。

この場合、インスタンス名は iris で、ネームスペース名は USER となります。ルーチン名は **^%D** (現在の日付を出力する) となります。

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が使用されます。

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。

```
iristerm /server=ServerName
```

ServerName には InterSystems IRIS サーバを指定します。使用できるサーバのリストを確認するには、[InterSystems ランチャー] を選択してから、**[優先接続サーバ]** を選択します。サーバのリストが表示されます。

UNIX® 以外のプラットフォームでは、以下を確認してください。

- 所望のサーバにおいて、Telnet サービス (**%Service_Telnet**) が有効であること。(既定ではこのサービスが有効ではないことに注意してください。)

詳細は、"セキュリティ管理ガイド" の "サービス" を参照してください。

- サーバが起動していること。

UNIX® では、サーバが起動している必要はありませんが、InterSystems IRIS に直接ではなく、シェルにログインすることになります。

注釈　`/console` および `/server` 引数は両方ともに指定できません。

6.2.3 /size=RowsxCols

この引数は、ターミナル画面の初期サイズを行数と列数で指定します。Rows と Cols はどちらも必ず符号なし整数で指定します。行数と列数の間に入る x は上記のとおりに挿入する必要があります。この制御引数ではスペースは使用できません。

Rows および Cols は次の範囲で指定します。

- 5 <= Rows <= 64
- 20 <= Cols <= 132

6.2.4 /pos=(X,Y)

この引数は、ディスプレイ・デバイス・ウィンドウに表示されるターミナル画面の初期原点をピクセル単位で指定します。X と Y はどちらも必ず符号なし整数で指定します。X と Y を囲む括弧、およびこれらを区切るコンマは省略できません。この制御引数ではスペースは使用できません。

注釈　ディスプレイ・デバイスのサイズよりも大きな値を X と Y に指定すると、表示領域の外側にウィンドウを配置することができます。

6.2.5 /ppos=(Xpct,Ypct)

この引数は、ディスプレイ・デバイス・ウィンドウに表示されるターミナル画面の原点の初期値を表示領域のパーセンテージで指定します。Xpct と Ypct はどちらも必ず符号なし整数で指定します。X と Y を囲む括弧、およびこれらを区切るコンマは省略できません。この制御引数ではスペースは使用できません。

XPct および Ypct は次の範囲で指定します。

- 0 <= Xpct <= 40
- 0 <= Ypct <= 40

つまり、ウィンドウの原点はデバイス原点の上と左側には配置できません。ディスプレイ・デバイスの 40% よりも下または右側にも配置できません。

6.2.6 /UnbufferedLogging

この引数により、ログがアクティブである場合は、バッファされる代わりに、ログ・ファイルに出力が直ちに書き込まれます。これは、別のプロセスがログ・ファイルの出力を検査しているときに便利です。

6.3 例

このセクションでは、このドキュメントで前述したスクリプト例を使用して、それを実行する 2 つの方法を示します。

6.3.1 バッチ・モードでのスクリプトの実行

この例では、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。スクリプト・コマンドの格納場所が C:¥TestScript.scr である場合は、次のサンプル・コマンドを DOS コマンド・ウィンドウに入力して実行できます。

```
C:\InterSystems\iris\bin\iristerm.exe /console=cn_iptcp:127.0.0.1[23] C:\TestScript.scr
```

6.3.2 スクリプトのインタラクティブな実行

前述のルーチンはターミナル・セッションで、次のように呼び出すことができます。

```
C:\InterSystems\iris\bin\iristerm.exe /console=cn_ap:iris[USER]:^^%STACK
```

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。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・ウィンドウが閉じます。

7

ターミナル使用に関するその他のトピック

この章では、さまざまな高度なトピックについて説明します。

7.1 ターミナルの閉じるボタンの無効化

ターミナルの閉じるボタン (X) を無効化する必要がある場合は、以下のようにレジストリ・キーを追加します。

- 32 ビット Windows マシンの場合 : HKEY_LOCAL_MACHINE¥SOFTWARE¥InterSystems¥Terminal¥NoExit
  "Terminal" にあるスペースに注意してください。
- 64 ビット Windows マシンの場合 :
  HKEY_LOCAL_MACHINE¥SOFTWARE¥ Wow6432Node ¥InterSystems¥Terminal¥NoExit=1

どちらの場合も、NoExit 値は REG_SZ 型です。

7.2 拡張キーボードのマッピング

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。

テーブル 7-1: キー・マッピング

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。

テーブル 7-2: キーボード・マッピング

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、単独の XON/XOFF トグル・キーとして機能します。

7.3 特殊モード

ターミナルには、キーボード・アクションからのみアクセス可能な 2 つの特殊モードが用意されています。

7.3.1 キーの時間計測モード

キーの時間計測モードに入る、または終了するには、Alt-Shift-T を押します。

このモードは、さまざまな負荷状況におけるホスト・システムのパフォーマンスの測定に役立ちます。時間計測の実行結果の出力先は、InterSystems IRIS システム・マネージャのディレクトリにある KEYTIMER.LOG ファイルです。

7.3.2 学習モード

学習モードでは、わずかな編集を行っただけで、ターミナルにより、簡単にスクリプト・ファイルに変換できるログ・ファイルが生成されます。このモードが有効な場合は、ログ・ファイルが一連の wait for および send スクリプト・コマンドになります。wait for コマンドは、送信したデータに先行する文字を 16 文字まで表示します。

学習モードに入るには、以下の手順を実行します。

1. Alt-L を押してロギングを有効にします。次に、このドキュメントで前述した“ターミナル・セッションのログへの記録”の説明のように、ログ・ファイル名とディレクトリを指定します。
2. Alt-Shift-L を押します。

学習モードを終了するには、Alt-Shift-L を押します。

7.4 DDE を使用したターミナルの使用法

ターミナルは DDE (Dynamic Data Exchange) リンクをサポートすることによって、他のアプリケーションがリモート・ホストとやり取りすることを実現しています。このセクションでは、ユーザが DDE に精通していることが前提となっています。ここで説明する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 タスクでは、DDE の使用時にターミナルの複数インスタンスを区別できません。このため、実行されているターミナルが 1 つの場合に限り、DDE を使用します。

7.4.1 DDE Layout 接続

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

テーブル 7-3: DDE Layout 接続

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。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。

7.4.2 DDE Screen 接続

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。

テーブル 7-4: DDE Screen 接続

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。

7.4.3 DDE Message 接続

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。

テーブル 7-5: DDE Message 接続

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。