ObjectScript シェルの使用法

Version 2024.1
2024-06-03

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

ObjectScript シェルの使用法
InterSystems IRIS Data Platform　Version 2024.1　2024-06-03

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

目次

1

ObjectScript シェルの概要

InterSystems IRIS® データ・プラットフォームは、学習、開発、デバッグに役立つコマンド行インタフェース (ObjectScript シェル) を提供します。これは、ObjectScript コマンドの実行 (およびその結果の確認) だけでなく、その他のさまざまなシェルへのアクセスに使用できます。また、ObjectScript シェルには、広範な行呼び出し機能および複数のカスタマイズ・オプションが用意されています。

注釈　ObjectScript シェルはターミナルやターミナル・プロンプトと呼ばれることもありますが、これらは完全に正確ではありません。それは、ターミナルという Windows アプリケーション (このシェルにアクセスできるようにするアプリケーション) もあるからです。必要に応じて、このドキュメントでは、この Windows アプリケーションを指す場合、より具体的なターミナル・アプリケーションというフレーズを使用します。

1.1 ObjectScript シェルの起動

ObjectScript シェルを起動する方法はいくつかあります。

- 任意のオペレーティング・システムで、iris terminal コマンドを使用して、オペレーティング・システムのプロンプトから起動できます。
- Windows では、ターミナル・アプリケーションを使用できます。これは、ウィンドウに ObjectScript シェルを表示し、追加オプションのメニュー・バーも提供する Windows アプリケーションです。ターミナル・アプリケーションについては、個別に詳しく説明しています。

関連する InterSystems IRIS サーバのセキュリティ設定に基づいて、ログインするよう求められるか、既定のユーザ名でログインされます。これにより、(既定では) プロンプトに現在のネームスペースの名前が表示されます。例えば、USER ネームスペースにいる場合、既定のプロンプトは以下のようになります。

Terminal

```
USER>
```

別のネームスペースに切り替えるには、以下の例のように、$namespace 変数を使用します。

Terminal

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

これは、コード内でネームスペースを切り替えるのと同じ方法です。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。

1.2 一般的な使用

プロンプトに単一行の ObjectScript コマンドを入力できます。ルーチンやクラス内のコード行とは異なり、先頭にスペース文字は不要です。以下に例を示します。

Terminal

```
do ^myroutine
```

Terminal

```
set dirname = "c:\test"
```

Terminal

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

シェルでは、入力した各行の後に Use 0 コマンドが暗黙的に発行されるため、出力は自動的にシェルに書き込まれます。

コマンド入力に複数行モードはないため、コマンド全体を 1 行に入力する必要があります (ただし、シェル内で ルーチンを定義することはできます)。

1.3 プロンプトの変化

特定のシナリオで、プログラム・スタック・レベルまたはトランザクション・レベルを示すために、プロンプトが変化して追加情報が表示されます。"利用可能なシェル" も参照してください。

1.3.1 プログラム・スタック・レベル (デバッグ・プロンプト)

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。

Terminal

```
USER 5d3>
```

Quit コマンドを入力すると、デバッグ・プロンプトを終了できます。または、エラーをデバッグします。"コマンド行ルーチンのデバッグ" を参照してください。

1.3.2 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。

Terminal

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

ObjectScript シェルを終了すると、これによりトランザクションがロールバックされます。

1.4 コマンドの繰り返し

前のコマンドを繰り返すには、目的のコマンドがシェルに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、Enter キーを押します。ObjectScript シェルでは、より広範な行呼び出し機能も、前のコマンドを検索するオプションと共にサポートされています。

1.5 コピーと貼り付け

テキストをコピーするには、そのテキストを選択して、Ctrl-Insert を押します。次にそれを貼り付けるには、Shift-Insert を押します。

ターミナル・アプリケーションを使用している場合は、その他のオプションも利用できます。

1.6 実行の中断

フォアグラウンドの実行を中断するには、Ctrl-Shift-c または Ctrl-c を押します。

1.7 スクロール、一時停止、および再開

アクティブなテキストが到着したときは常に、ObjectScript シェルによって、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することもできます。

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。
Ctrl-s	スクロールを一時停止します。スクロールが一時停止されている間、ObjectScript シェルは、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)。
Ctrl-q	スクロールを再開します。

1.8 終了

ObjectScript シェルを終了するには、以下の手順を実行します。

1. 現在、使用可能ないずれかの (セカンダリ) シェル内にいる場合は、そのシェルを終了します (詳細はさまざまです)。
2. HALT または H と入力します (大文字/小文字の区別なし)。

iris terminal コマンドから ObjectScript シェルにアクセスしていた場合、プロンプトはオペレーティング・システムのプロンプトに戻ります。または、ターミナル・アプリケーションにアクセスしていた場合は、そのアプリケーション・ウィンドウが閉じます。

1.9 関連項目

- 利用可能なシェル
- コマンドの履歴とエイリアス
- ObjectScript シェルのカスタマイズ

2

利用可能なシェル

ObjectScript シェルでは、さまざまな方法でデータのクエリまたはコードを実行できる他のシェルを起動して、使用することができます。オペレーティング・システム・シェルを起動して使用することもできます。

2.1 Python シェル

Python シェルを起動するには、以下のコマンドを入力します。

Terminal

```
do ##class(%SYS.Python).Shell()
```

プロンプトが >>> になります。

このようになったら、Python コマンドを入力して、その結果を確認できます。

このシェルを終了するには、コマンド quit() (大文字/小文字の区別なし) を入力します。

詳細は、"Python シェルから実行" を参照してください。

2.2 SQL シェル

SQL シェルを起動するには、以下のコマンドを入力します。

Terminal

```
Do $SYSTEM.SQL.Shell()
```

プロンプトが変化して、SQL シェル内で実行していることが示されます。具体的には、[SQL] 接頭語が追加され、末尾に > が追加されます。以下に例を示します。

Terminal

```
[SQL]USER>>
```

このようになったら、SQL クエリを入力して、その結果を確認できます。

注釈　　SQL シェル内では、Ctrl-c コマンド (通常は、中断に使用される) は無効です。

このシェルを終了するには、q または quit (大文字/小文字の区別なし) を入力します。

複数行モードや SQL 言語の指定も含め、詳細は、"SQL シェル・インタフェースの使用法" を参照してください。

2.3 TSQL シェル

TSQL シェルを起動するには、以下のコマンドを入力します。

Terminal

```
Do $system.SQL.TSQLShell()
```

プロンプトが変化して、TSQL シェル内で実行していることが示されます。具体的には、末尾に > が追加されます。

このようになったら、Transact-SQL クエリを入力して、その結果を確認できます。MSSQL 言語または Sybase 言語のいずれかを使用できます。

このシェルを終了するには、q または quit (大文字/小文字の区別なし) を入力します。

詳細は、"SQL を使用した TSQL の操作" を参照してください。

2.4 MDX シェル

MDX シェルを起動するには、以下のコマンドを入力します。

Terminal

```
Do ##class(%DeepSee.Utils).%Shell()
```

プロンプトが >> になります。

このようになったら、MDX クエリを入力して、その結果を確認できます。

このシェルを終了するには、q または quit (大文字/小文字の区別なし) を入力します。

詳細は、"MDX シェル" を参照してください。

2.5 オペレーティング・システム・シェル

ObjectScript シェルでは、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。プロンプトに作業ディレクトリが表示されます。以下に例を示します。

Terminal

```
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。

3

コマンドの履歴とエイリアス

ObjectScript シェルでは、広範な行呼び出し機能が、前のコマンドを検索するオプションと共にサポートされているだけでなく、コマンドのエイリアスを定義する機能もサポートされています。これらのオプションは ObjectScript の拡張機能ではないため、ルーチンやメソッドで使用することはできません。

3.1 行呼び出し機能

ObjectScript シェルでは、上矢印以外にも、履歴バッファに保存されたコマンドのリストを使用する、より広範な行呼び出し機能もサポートされています。この機能により、以前に入力した特定のコマンドを呼び出すことができます。

行呼び出しを有効化するには、コロン (:) に続けて、コマンドを明確に特定する文字を 1 つ以上入力します。

さまざまな特殊なコマンドを発行して、コマンド・プロンプトで行呼び出しを操作できます。まず、:? は、クイック・ヘルプを表示します。

Terminal

```
USER>:?
 :<number>    Recall command # <number>
 :alias       Create/display aliases
 :clear       Clear history buffer
 :history     Display command history
 :unalias     Remove aliases
```

:history コマンドは、履歴バッファに保存されているコマンドの番号付きリストを生成します。以下の例のように、その番号を使用して特定のコマンドを呼び出すことができます。

Terminal

```
USER>:h[istory]
 1: zn "%SYS"
 2: Do ^SYSLOG
 3: write "Hello"," World!"
 4: write $J
 5: :h

$SYS>:3
write "Hello"," World!"
Hello World!
```

:alias コマンドによって、プロンプトでよく入力する ObjectScript コマンドのショートカットが作成され、:unalias <name> によって、その名前に関連付けられた定義が削除されます。"エイリアス" を参照してください。

:clear コマンドは、呼び出しバッファの内容をすべて削除しますが、~¥.iris_history は削除しません。

3.2 逆インクリメンタル検索

逆インクリメンタル検索 (RIS) は、ObjectScript シェル呼び出し機能の追加の動作モードです。このモードにするには、Ctrl-R を入力します。RIS モードを使用するには、呼び出しバッファにコマンドが保存されている必要があります。バッファが空の場合、ビープ音が鳴る以外には何も実行されません。プロンプトの後にある 2 つの円記号が RIS モードを示します。

Terminal

```
USER>\\
```

検索文字列はこの 2 つの円記号の間に表示され、一致するコマンドがあれば右側に表示されます。検索文字列に文字を入力すると、コマンド履歴バッファが履歴と同じ方向にスキャンされます。上矢印を入力すると、最も新しいコマンドから最も古いコマンドへスキャンされます。入力した文字列が含まれる最初のコマンドまで来ると、スキャンは停止します。コマンドを実行するには Enter を押します。

例えば、呼び出しバッファにコマンド W $J がある場合、'$' キーを入力すると、ObjectScript シェルに以下のように表示されます。

Terminal

```
USER>\$\ W $J
```

代わりにコマンド W $ZV を見つけたい場合は、その次の文字 Z を入力します。

Terminal

```
USER>\$Z\ W $ZV
```

ANSI 属性をサポートするオペレーティング・システムでは、呼び出されたコマンドに一致する文字が太字属性で表示されます。

以下を入力すると、RIS モードは終了します。

- Enter – コマンドを選択してただちに実行します。
- Ctrl-G – RIS モードをキャンセルして、コマンドを実行せずに前のプロンプトに戻ります。
- 編集キー (左矢印や Ctrl-A など) – 編集モードを起動して、呼び出されたコマンド行を編集できます。
- 上矢印または下矢印 – 検索によって選択されたコマンドを基準にして、その前または次のコマンドを選択します。

以下のいずれかを入力すると、検索モードで続行します。

- 非制御文字 – 入力した文字が検索文字列に付加され、その新しい文字列を含む以前のコマンドが検索されます。
- Ctrl-R – 同じ検索文字列を含む以前のコマンドを検索します。
- DEL または BS – 検索文字列の右端の文字を消去し、対応するコマンドに戻ります。

3.3 エイリアス

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。セッションの残りの部分では、: 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。

Terminal

```
%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。

Terminal

```
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。

Terminal

```
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。

Terminal

```
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。

各セッションの起動時に ObjectScript シェルが自動的に設定するエイリアス定義のリストを提供することもできます。こういったエイリアスは、ホーム・ディレクトリ内の .iris_init という名前のファイルに (1 行に 1 つずつ) 定義します。このディレクトリは、具体的には、オペレーティング・システム内の現在のユーザのホーム・ディレクトリとなります。(例えば、Windows の場合、現在のユーザに対応する C:¥Users¥ サブディレクトリです。)

4

ObjectScript シェルのカスタマイズ

ここでは、ObjectScript シェルをカスタマイズする方法を説明します。

4.1 シェル内でのルーチンの定義

外部エディタを使用せずに、ルーチンを定義 (およびそれを実行) できます。以下のトピックを参照してください。

- Tab キーの省略表現を使用したルーチンの作成
- ターミナルからのルーチンの作成 (この手法では、ZLOAD、ZINSERT、および ZSAVE コマンドを使用します)

4.2 開始ネームスペースの変更

ObjectScript シェルを開始する際、そのユーザは、特定のネームスペースへのアクセス権を持ち、そのネームスペースでコードを実行することができます。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。"ユーザ・アカウント" を参照してください。

4.3 プロンプトのカスタマイズ

既定の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、構成オプション TerminalPrompt を設定します。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、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。

4.4 ^ZWELCOME ルーチン

ObjectScript シェルが起動すると、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、シェルのログイン・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。

以下はその簡単な例です。

Terminal

```
^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
```

^ZWELCOME ルーチン (**%SYS** ネームスペースに配置する必要あり) を作成するには、管理者特権と **IRISSYS** データベースへの書き込み権限が必要です。

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。