



# Caché ^%ZSTART および ^%ZSTOP ルーチンの使用 法

Version 5.1

2006-03-14

Caché ^%ZSTART および ^%ZSTOP ルーチンの使用法

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 ^%ZSTART および ^%ZSTOP ルーチンの使用法.....	1
1 はじめに .....	1
2 設計に関する考慮事項 .....	1
3 %ZSTART および %ZSTOP を有効にする .....	3
4 ^%ZSTART および ^%ZSTOP のデバッグ .....	4
5 実装例 .....	5
5.1 ルーチン: ^%ZSSUtil .....	5
5.2 ルーチン: ^%ZSTART .....	8
5.3 ルーチン: ^%ZSTOP .....	10



# Caché ^%ZSTART および ^%ZSTOP ルーチンの使用法

## 1 はじめに

Caché は、多くのシステムと同じように、特定のイベントが発生したときに、ユーザが記述したルーチン呼び出すメカニズムを提供します。各イベントの処理は、通常システム自体のサブルーチンとして扱われます。Caché には、^%ZSTART および ^%ZSTOP という 2 つのルーチンがあります。それぞれが、システムが事前に定義した 4 組の中の 1 つ以上のラベルを定義します。これらのラベルは、イベントのクラスが起きた際に呼び出される各ルーチンへのエントリ・ポイントになります。

- ・ SYSTEM: Caché がシステムとして開始または停止します
- ・ LOGIN: ユーザがログインまたはログアウトを実行します
- ・ JOB: JOB が開始または終了します
- ・ CALLIN: 外部プログラムが開始または CALLIN が完了します

適切なラベルは、その動作が開始ならば ^%ZSTART 内で、終了ならば ^%ZSTOP 内で呼び出されます。

これらのエントリ・ポイントは、Caché のサブルーチンとして機能しますが、複雑な計算をしたり、長時間実行するためのものではありません。ネットワーク・アクセスなどのように長期的に計算したり、潜在的に長時間わたって実行すると、呼び出されたルーチンが返されるまで動作の完了が遅れます。この場合、ユーザのログインに長い時間がかかったり、時間がかかりすぎるために JOB の処理能力が削減される場合があります。

## 2 設計に関する考慮事項

^%ZSTART および ^%ZSTOP の実行環境には、ある程度制約があります。設計者は以下のような点に注意してください。

- ・ ルーチンは Caché ObjectScript で記述しなければなりません。
- ・ 任意のルーチン・エントリ・ポイントが呼び出されたときに、引数として渡される値はありません。さまざまな状況で異なるアルゴリズムが適用できる場合は、呼び出されたエントリ・ポイントはルー

チンの外部、つまりグローバルやシステム変数や \$ZUTIL ルーチンからの返り値などのデータを調査して、何を実行するかを決定する必要があります。

- ・ あらゆる条件下で、ルーチンがうまく動作することを確認してください。ルーチンは防御的に記述します。つまり、タスクの完了に必要なすべてのリソースがすぐ使用できる場所にあり、可能であれば演算処理が始まる前にルーチンに予約されることを、これらのルーチンが確認します。発生したエラーはそのシステム関数の失敗として報告されるので、エラー抑制と処理の観点から設計を考えることが重要です。リソースが見つからない、あるいはエラーが起こる際に適切に回復できなければ、Cacheの開始の失敗、スタジオなどの主要な機能の動作の不具合、またはすぐには検知されないようなわずかな内部的な影響など、さまざまな結果を引き起こします。これらのルーチンは、細心の注意を払って記述し、シミュレート条件下でデバッグし、製品に使用する前にシミュレート環境下でテストすることを推奨します。
  - ・ 以前の呼び出しの間に見つかった条件や、異なるエン트리・ポイントが有効であるとは考えないでください。例えば、JOB^%ZSTART への連続呼び出し間に、次の呼び出しが発生する前に前回使用したファイルが削除される可能性もあります。
  - ・ 各エン트리・ポイントはタスクを効率的に実行します。タスクの一部が、潜在的に長時間実行するもので、完了に十分な情報であれば、ユーザのアプリケーションの別の部分によって後から処理するように、待ち行列に入れられます。
  - ・ エン트리・ポイントが統計的に使用する目的でデータを永続的に保持したい場合、データの保持にはグローバルや外部ファイルなどを使用する必要があります。
  - ・ ルーチンを実行する環境についての条件は、最小限にしてください。例えば、このようなルーチンの開発者は、プログラムが常に特定のジョブ数以下で実行されるとは予測できませんし、ルーチンを実行する環境についての条件は、最小限にしてください。設計者は、特定の命令でさまざまなエン트리・ポイントが呼び出されることを想定できません。Cacheを実装する複数のプロセスを呼び出す配列が、決定性である(次のステップが1つに決まる)ことはほとんどありません。
  - ・ ルーチンは、システムの開始中の特定の時点で呼び出されるとは限りません。開始中のイベントの配列は、リリースの度に、または再開の度に変更する可能性があります。
  - ・ いくつかの例外を除き、^%ZSTART と ^%ZSTOP の中では装置情報を変更してはいけません。例えば、^%ZSTART と ^%ZSTOP の中で実行されるルーチン内で、\$IO の値を待避せずに変更すると、エラーの原因になります。後続のルーチンは、\$IO に変更があったことを知る方法がなく、また、このような変更を防ぐことができません。したがって、^%ZSTART と ^%ZSTOP で実行されるサブルーチンは、装置情報を混乱させるような変更処理をしてはいけません。
- 変更不可という規則の例外として、アプリケーション・プログラムやインストール・プログラムの処理中に使用される、内部的な値は変更できます。例えば、SYSTEM^%ZSTART エン트리・ポイントで1つ以上の \$ZUTIL(69) のサブ関数を呼び出し、システム全体の既定値を設定します。同様に、アプリケーションのテストのために、\$ZUTIL(71,date)を呼び出して任意の日付を設定し、月末処理の妥当性を検証することもできます。
- ・ ^%ZSTOP には、リモート・データベース内のグローバルへの参照を含めることができません。^%ZSTOP の呼び出し時に、これらの参照の一部がアクセスできなくなります。

## 3 %ZSTART および %ZSTOP を有効にする

^%ZSTART と ^%ZSTOP は、%SYS ネームスペースに属さなければ (したがってコンパイルされなければ) なりません。しかし、Cache がこれらを使用して自動的に開始するには、ただそこにあるだけでは不十分です。代わりに、構成マネージャから個々のエントリ・ポイントを有効または無効にします。

一旦ルーチンを設計、開発、コンパイルし、テストができる段階になると、構成マネージャから個々のエントリ・ポイントを有効にすることができます。構成マネージャを開始し、[詳細] タブを選択します。表示されたパネルで [開始] から、[ユーザ・コード有効] を開くと、8 項目のリストが表示されます。このリストは、4 つのイベント・クラスの動作開始または停止の、すべての組み合わせに対応しています。

- ・ システム開始用
- ・ システム停止用
- ・ プロセス開始用
- ・ プロセス停止用
- ・ ジョブ開始用
- ・ ジョブ停止用
- ・ コールイン開始用
- ・ コールイン停止用

これらの各項目を、有効にする (= はい) または無効にする (= いいえ) に指定します。設定を変更するには、項目を強調表示にして [変更] オプションを選択します。項目の有効 (チェックあり)/無効 (チェックなし) を示すチェック・ボックスを持つダイアログ・ボックスが表示されるので、必要に応じて変更し、設定を変更した場合は [OK] をクリックして適用します。

構成マネージャの終了時に、ユーザはこの設定を適用するか、適用しないかを選択することができます。場合によっては、この設定を有効にするために Cache を再起動する必要があります。

例えば、バックグラウンド・プロセスが開始するたびに何かを得たいとします。以下のすべてに該当する場合、適切なエントリ・ポイント JOB%^ZSTART が呼び出されます。

- ・ 構成マネージャで、希望するルーチンのエントリ・ポイントが有効になっていること。つまり、構成マネージャの [詳細] タブの [開始] 項目の [ユーザ・コード有効] リストを展開すると、“ジョブ開始用 = はい” と表示されています。
- ・ 構成マネージャに制御されるローカルの Cache システムに、^%ZSTART がコンパイルされロードされること。
- ・ ルーチンが JOB エントリ・ポイントを含むこと。

## 4 ^%ZSTART および ^%ZSTOP のデバッグ

最終的な環境において ^%ZSTART と ^%ZSTOP をデバッグする機会ほとんどありません。エラーが発生した際にインタラクティブ・セッションでターミナルに記述されるエラー・メッセージは、オペレータのコンソール・ログに転送されます。これは Caché マネージャ・ディレクトリにある “cconsole.log” ファイルです。

メッセージは、失敗の原因とエラーが検出された位置を示します。この位置は、プログラム・ロジックまたはフローで実際にエラーが発生した場所とは異なる場合があります。開発者は、提供された情報からエラーの特性と場所を推定し、ルーチンを修正してください。これにより、今後のテストでは、発生するエラーの特性についてより多くの情報を得ることができるようになります。

極端な失敗が発生した場合でない限り、他の Caché の機能が使用できなくなっても、構成マネージャは開始したり、実行することができます。つまり、原因の関数の使用を無効にし Caché を再起動することにより、そのエラー状態から回復し分析することができます。

1 つ以上のエントリ・ポイント呼び出しを無効にするには、以下を実行します。

1. [構成マネージャ] を起動します。
2. [詳細] タブをクリックします。
3. [開始] の [ユーザ・コード有効] を開きます。
4. それぞれの項目を選択し、[変更] をクリックして、各オプションの設定を変更します。
5. 変更を加えた後、[OK] をクリックします。

場合によっては、この設定を有効にするために Caché を再起動する必要があります。

^%ZSTART および ^%ZSTOP (および、サポートする任意のルーチン) が正確に格納されるように注意してください。これらのトレースをすべて削除するには、以下を実行します。

1. エクスプローラを起動します。
2. %SYS ネームスペースを開きます。
3. [ルーチン] を選択します。
4. 右側のパネル (ルーチンの内容) で、削除するルーチンを探します。
5. 削除したいルーチンを選択します。
6. [ファイル] メニューから [削除] を選択します。次に表示されるダイアログ・ボックスで、[はい] を選択すると、削除を確認します。

注釈: 必ず、ルーチンを削除する前に、Cache 構成マネージャ経由でエントリ・ポイントを無効にしてください。この変更を有効にするために、構成マネージャが Cache を再起動するように警告した場合は、次に進む前に再起動も実行してください。これにより、エントリ・ポイントが削除される間に、実行されることはありません。

## 5 実装例

以下は、システムの動作状況を追跡する単純なログの実例です。これは、`^%ZSSUtil` と呼ばれるログ・エントリを記述する共通のユーティリティと、`^%ZSTART` と `^%ZSTOP` に対する別々のルーチンで構成されます。`^%ZSTOP` ルーチンは、`^%ZSTART` の多くの動作に似ているため、ここでは `^%ZSTART` と `^%ZSSUtil` について主に説明しますが、上記の 3 つのルーチンについて、すべて実例を示しています。

注釈: 以下の 3 つのコードは、ユーザの理解を深めるために記述されたものです。ObjectScript の便利な利用法や簡潔な記述方法を示したものではありません。

### 5.1 ルーチン: `^%ZSSUtil`

このルーチンには、2 つのエントリ・ポイントがあります。一方は、オペレータのコンソール・ログに 1 行記述し、他方は、ローカルのログ・ファイルに名前と値の組み合わせのリストを記述します。オペレータのコンソール・ログもローカルのログ・ファイルも Cache の管理者のディレクトリにあります。このディレクトリの名前は、`ManagerDirectory` クラスの `%Library.File` メソッドにより返されます。

## 実装例

---

```
%ZSSUtil ;
; this routine packages a set of subroutines
; used by the %ZSTART and %ZSTOP entrypoints
;
; does not do anything if invoked directly
quit

#Define Empty ""
#Define OprLog 1

WriteConsole(LineText) PUBLIC ;
; write the line to the console log
; by default the file cconsole.log in the MGR directory
new SaveIO

; save the current device and open the operator console
; set up error handling to cope with errors
; there is little to do if an error happens
set SaveIO = $IO
set $ZTRAP = "WriteConsoleExit"
open $$$OprLog
use $$$OprLog
; we do not need an "!" for line termination
; because each WRITE statement becomes its
; own console record (implicit end of line)
write LineText
; restore the previous io device
close $$$OprLog
; pick up here in case of an error

WriteConsoleExit ;
set $ZTRAP = ""
use SaveIO
quit

WriteLog(rtnname, entryname, items) PUBLIC ;
; write entries into the log file
; the log is presumed to be open as
; the default output device
;
; rtnname: distinguishes between ZSTART & ZSTOP
; entryname: the name of the entrypoint we came from
; items: a $LIST of name-value pairs
new ThisIO, ThisLog
new i, DataString

; preserve the existing $IO device reference
; set up error handling to cope with errors
; there is little to do if an error happens
set ThisIO = $IO
set $ZTRAP = "WriteLogExit"

; construct the name of the file
; use the month and day as part of the name so that
; it will create a separate log file each day
set ThisLog = "ZSS"
_ "-"
_ $EXTRACT($ZDATE($HOROLOG, 3), 6, 10)
_ ".log"

; and change $IO to point to our file
open ThisLog:"AWS":0
use ThisLog

; now loop over the items writing one line per item pair
for i = 1 : 2 : $LISTLENGTH(items)
{
set DataString = $LISTGET(items, i, "MISSING")
if ($LISTGET(items, (i + 1), $$$Empty) '= $$$Empty)
```

```

    {
        set DataString = DataString
                        - ": "
                        - $LISTGET(items, (i + 1))
    }
    write $ZDATETIME($HOROLOG, 3, 1),
        ?21, rtnname,
        ?28, entryname,
        ?35, DataString, !
}

; stop using the log file and switch $IO back
; to the value saved on entry
close $IO
; pick up here in case of an error
WriteLogExit ;
set $ZTRAP = ""
use ThisIO
quit

```

## 5.1.1 説明

### ラベル: ^%ZSSUtil

このルーチンは、他のルーチンと同様に、まず QUIT コマンドを実行して、以下のコマンドから呼び出されると良好な結果が得られます。

```
do ^%ZSSUtil
```

#DEFINE 配列は、外観をそろえるために、プログラムの本文に指定された制約を提供します。このインスタンスでは、空文字列とオペレータのコンソール・ログのデバイス番号を指定します。

### ラベル: WriteConsole^%ZSSUtil

このエントリ・ポイントは非常に単純です。容量の少ない出力用に、またデバッグの出力に使用するための最小限のルーチンとして、設計されたものです。

引数として1つの文字列を持ち、これをオペレータのコンソール・ログに記述します。しかし、現在の呼び出しの \$IO 装置の保存とリストアは、注意して実行しなければなりません。

コンソール・デバイスでは、デバイス各項目が送信された結果、コンソール・ログには別々のレコードが記述されます。以下にコードの例があります。

```
WRITE 1, 2, 3, !
```

上記は、4つのレコードを記述します。最初の3つは1桁の数字からなり、4つめは空白行からなります。1行に複数の項目を記述したい場合は、呼び出し元がこれらを文字列に連結させなければなりません。

### ラベル: WriteLog^%ZSSUtil

これは、上記のルーチンよりも複雑です。^%ZSTART または ^%ZSTOP 内の任意のエントリ・ポイントから呼び出すことができます。最初の2つの引数は、このラベルが何に対して実行しているかについてのレポートに必要な情報を提供します。3番目の引数は、ログに記述される名前と値の組み合わせの \$LIST です。

## 実装例

---

このエントリ・ポイントは、まず使用するファイルの名前を作成します。ログ管理を簡素化するため、この名前にはルーチンが入力された月日が含まれます。したがって、WriteLog への呼び出しは、現地時間が深夜 12 時を過ぎる度に、新規のファイルを作成します。その名前は呼び出し時のみ決定されるため、引数として渡される名前と値の組み合わせはすべて同じファイルに表示されます。

一旦名前が作成されると、\$IO の現在値を後で使用できるように保存し、出力デバイスを指定されたログ・ファイルに切り替えます。OPEN コマンドに使用するパラメータによって、そのファイルがなければ作成するように指定されています。timeout がゼロの場合、Caché がファイルを 1 回だけ開こうとします。もし開くことができなければ失敗します。

そのファイルが一旦開かれると、コードは名前と値の組み合わせをループします。各組み合わせに対し、その行に呼び出しルーチン名とエントリ・ポイント名、続けて名前と値の組み合わせが記述されます (値の部分が空文字列の場合は、名前のみ記述)。組み合わせは 1 行に 1 個ずつ記述されます。読みやすいように、各行の最初の 3 つの値は一行に並ぶようになっています。

すべての組み合わせの記入後、ログ・ファイルを終了し、先ほど保存した \$IO の値がリストアされ、コントロールは呼び出し元に返されます。

## 5.2 ルーチン: ^%ZSTART

このルーチンには、実際に Caché に呼び出されるエントリ・ポイントを含みます。上記の ^%ZSUtil の機能も使用します。すべてのエントリ・ポイントは、事実上同じように動作し情報をログに配置します。SYSTEM エントリ・ポイントは、他に比べるとやや複雑ですが、同様にオペレータのコンソール・ログに情報を配置します。

```

%ZSTART ; User startup routine.

#Define ME "ZSTART"
#Define BgnSet "Start"
#Define Empty ""

    ; cannot be invoked directly
    quit

SYSTEM ;
    ; Cache starting
    new EntryPoint, Items

    set EntryPoint = "SYSTEM"

    ; record the fact we got started in the console log
    do WriteConsole^%ZSSUtil((EntryPoint
        - "^%"
        - $$$ME
        - " called @ "
        - $ZDATETIME($HOROLOG, 3)))

; log the data accumulate results
    set Items = $LISTBUILD($$$BgnSet, $ZDATETIME($HOROLOG, 3),
        "Job", $JOB,
        "Computer", $ZUTIL(110),
        "Version", $ZVERSION,
        "StdIO", $PRINCIPAL,
        "Namespace", $ZUTIL(5),
        "CurDirPath", $ZUTIL(12),
        "CurNSPath", $ZUTIL(12, ""),
        "CurDevName", $ZUTIL(67, 7, $JOB),
        "JobType", $ZUTIL(67, 10, $JOB),
        "JobStatus", $ZHEX($JOB),
        "StackFrames", $STACK,
        "AvailStorage", $STORAGE,
        "UserName", $ZUTIL(67, 11, $JOB))
    do WriteLog^%ZSSUtil($$$ME, EntryPoint, Items)

    quit

LOGIN ;
    ; a user logs into Cache (user account or telnet)
    new EntryPoint, Items

    set EntryPoint = "LOGIN"
    set Items = $LISTBUILD($$$BgnSet, $ZDATETIME($HOROLOG, 3))
    do WriteLog^%ZSSUtil($$$ME, EntryPoint, Items)
    quit

JOB ;
    ; JOB'd process begins
    new EntryPoint, Items

    set EntryPoint = "JOB"
    set Items = $LISTBUILD($$$BgnSet, $ZDATETIME($HOROLOG, 3))
    do WriteLog^%ZSSUtil($$$ME, EntryPoint, Items)
    quit

CALLIN ;
    ; a process enters via CALLIN interface
    new EntryPoint, Items

    set EntryPoint = "CALLIN"
    set Items = $LISTBUILD($$$BgnSet, $ZDATETIME($HOROLOG, 3))
    do WriteLog^%ZSSUtil($$$ME, EntryPoint, Items)
    quit

```

## 5.2.1 説明

### ラベル: `^%ZSTART`

`^%ZSSUtil` の説明でも述べたように、このルーチンはまず `QUIT` コマンドを実行します。エントリ・ポイントの 1 つから実行を開始するのではなく、ルーチンとして呼び出す方が良好な結果を得ることができるためです。

このルーチンは、そのルーチン自体の名前、開始文字列および空文字列に対し、指定された定数 (マクロとして) を定義します。

### ラベル: `SYSTEM^%ZSTART`

`^%ZSSUtil` がログ・ファイルの記述に関する詳細な機能の大部分を担うため、このルーチンは主に情報を収集し、その後 `^%ZSSUtil` で適切なエントリ・ポイントを呼び出します。これは、そのルーチン名、エントリ・ポイント、呼び出された日付時刻を使用した文字列を作成します。その後、`WriteConsole^%ZSSUtil` を呼び出してオペレータのコンソール・ログに配置します。

後で、表示したい名前と値の組み合わせのリストを作成し、これを `WriteLog^%ZSSUtil` に渡してローカルのログ・ファイルに配置します。その後、呼び出し元に戻ります。

### ラベル: `LOGIN^%ZSTART`、`JOB^%ZSTART`、`CALLIN^%ZSTART`

`SYSTEM` エントリ・ポイントとは異なり、これらはオペレータのコンソール・ログに情報を置きません。呼び出す際に識別するための簡単な項目のリストを作成し、`WriteLog^%ZSSUtil` を使用して記録します。

## 5.3 ルーチン: `^%ZSTOP`

このルーチンは、実際に `Caché` に呼び出されるエントリ・ポイントを含みます。`^%ZSTART` と同様に、`^%ZSSUtil` の機能を使用します。`^%ZSTOP` のエントリ・ポイントは、`%ZSTART` のエントリ・ポイントと同じ名前と同様の機能を持ち、`SYSTEM` エントリ・ポイントのみがオペレータのコンソール・ログに記述します。すべてのルーチンは、ローカルのログに動作状況を記録します。

```

%ZSTOP ; User shutdown routine.

#Define ME "ZSTOP"
#Define EndSet "End"
#Define Empty ""

    ; cannot be invoked directly
    quit

SYSTEM ; Cache stopping
new EntryPoint

set EntryPoint = "SYSTEM"
; record termination in the console log
do WriteConsole^%ZSSUtil((EntryPoint
    - "^%"
    - $$$ME
    - " called @ "
    - $ZDATETIME($HOROLOG, 3)))
; write the standard log information
do Logit(EntryPoint, $$$ME)
quit

LOGIN ; a user logs out of Cache (user account or telnet)
new EntryPoint

set EntryPoint = "LOGIN"
do Logit(EntryPoint, $$$ME)
quit

JOB ; JOB'd process exits.
new EntryPoint

set EntryPoint = "JOB"
do Logit(EntryPoint, $$$ME)
quit

CALLIN ; process exits via CALLIN interface.
new EntryPoint

set EntryPoint = "CALLIN"
do Logit(EntryPoint, $$$ME)
quit

Logit(entrypoint, caller) PRIVATE ;
; common logging for exits

new items

set items = $LISTBUILD($$$EndSet, $ZDATETIME($HOROLOG, 3))
do WriteLog^%ZSSUtil(caller, entrypoint, items)
quit

```

### 5.3.1 説明

このルーチンの機能は、^%ZSTART に類似しています。詳細は、^%ZSTART の概要、例、説明を参照してください。

