



# Caché \$ZOBJxxx 内部関 数の使用法

Version 5.1

2006-03-14

Caché \$ZOBJxxx 内部関数の使用法  
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 \$ZOBJxxx 内部関数の使用法.....	1
\$ZOBJCLASSMETHOD 内部関数 .....	1
\$ZOBJMETHOD 内部関数 .....	3
\$ZOBJPROPERTY 内部関数 .....	4



# Caché \$ZOBJxxx 内部関数の使用法

このドキュメントでは、Caché が提供する関数 (内部関数) について説明します。この関数は、クラスや、そのメソッドまたはプロパティへの参照を実行時に処理することにより、オブジェクトの処理の一般化を図ります。

この資料では、以下の内部関数について説明します。

- ・ [\\$ZOBJCLASSMETHOD](#)
- ・ [\\$ZOBJMETHOD](#)
- ・ [\\$ZOBJPROPERTY](#)

ObjectScript 関数に関する詳細は、Caché ObjectScript の使用法 の “関数” を参照してください。

ここでは、関数名はすべて大文字で表記されていますが、実際は大文字と小文字は区別しません。

## \$ZOBJCLASSMETHOD 内部関数

```
$ZOBJCLASSMETHOD (Classname, Methodname, Arg1, Arg2, Arg3, ... )
```

### パラメータ

Classname	アクセスできる既存のクラスの名前
Methodname	指定されたクラス内のクラス・メソッドの名前
Arg1, Arg2, Arg3, ...	classmethod が呼び出されたときに、その引数と置換される値

### 概要

この関数は、Caché ObjectScript プログラムが任意のクラス内の任意のクラス・メソッドを呼び出すことを許可します。クラス名とメソッド名の両方は、実行時に処理されるか、文字列制約として提供されます。

メソッドが引数を持つ場合は、methodname の後の引数リストが引数を提供します。最大 255 の引数値がメソッドに渡されます。

## パラメータ

### Classname

有効な文字列として評価する式。文字列の内容は、既存のアクセスできる、すでにコンパイルされたクラスの名前と正確に一致する必要があります。Cache クラスへの参照の場合、その名前はキャノニック形式 (%Library.String) か省略形式 (%String) のどちらかです。

### Methodname

有効な文字列として評価する式。文字列の値は、Classname により特定される、クラス内の既存の ClassMethod の名前と一致する必要があります。

### Arg1, Arg2, Arg3, ...

指定されたメソッドの引数に対し、順次置換される一連の式。式の値は、任意のタイプです。作成者は、提供された式のタイプはメソッドが予期するものと一致すること、また宣言された限界内に値があることを確認してください。

指定されたメソッドが引数を予期していない場合、この関数呼び出しで MethodName 以外の引数を与える必要はありません。余分な引数は無視されます。

## 備考

関数またはプロシージャとして \$ZOBJCLASSMETHOD を呼び出すと、ターゲット・メソッドの呼び出しを確定します。詳細は、以下の例を参考にしてください。

存在しないメソッドの呼び出しや、ClassMethod であると宣言されていないメソッドの呼び出しを実行すると、<METHOD DOES NOT EXIST> というエラーを生じます。

## 例

以下は、\$ZOBJCLASSMETHOD を関数として使用する例です。

```
set ClassName = "%Dictionary.ClassDefinition"  
set ClassmethodName = "NormalizeClassname"  
set SingleArgument = "%String"  
write $ZOBJCLASSMETHOD(ClassName, ClassmethodName, SingleArgument), !
```

# \$ZOBJMETHOD 内部関数

```
$ZOBJMETHOD ( Instance, Methodname, Arg1, Arg2, Arg3... )
```

## パラメータ

Instance	メソッド名を含むクラスのインスタンス
Methodname	指定されたクラス内のメソッドの名前
Arg1, Arg2, Arg3, ...	methodname が呼び出されたときに、その引数と置換される値

## 概要

この関数は、Caché ObjectScript プログラムが、あるクラスの既存のインスタンス内の任意のメソッドを呼び出すことを許可します。最初の引数はオブジェクトへの参照でなければならぬため、実行時に算出されます。メソッド名は、実行時に算出されるか、文字列リテラルとして提供されます。メソッドが引数を持つ場合、これらは methodname の後の引数リストが提供します。最大 255 の引数値がメソッドに渡されます。メソッドに予期されていない余分な引数は、無視されます。

## パラメータ

### Instance

オブジェクト参照に評価する式。式の値は、希望するクラスのインメモリ・インスタンスの式でなければなりません。

### Methodname

有効な文字列として評価する式。文字列の値は、最初の引数で指定されたクラスのインスタンス内の、既存のメソッドの名前と一致する必要があります。

### Arg1, Arg2, Arg3, ...

指定されたメソッドの引数に対し、順次置換される一連の式。式の値は、任意のタイプです。作成者は、提供された式のタイプはメソッドが予期するタイプと一致すること、また限界内に値があることを確認してください。

指定されたメソッドが引数を予期していない場合、この関数呼び出しで Classname と MethodName 以外の引数を与える必要はありません。余分な引数は無視されます。

## 備考

関数またはプロシージャとして \$ZOBJMETHOD を呼び出すと、ターゲット・メソッドの呼び出しを確認します。詳細は、以下の例を参考にしてください。

## \$ZOBJPROPERTY 内部関数

クラス・インスタンスの 1 つのメソッド内で、そのインスタンスの別のメソッドを参照するために \$ZOBJMETHOD を使用するときには、Instance を省略できます。しかし、通常 Instance の後に続くコンマは必要です。

存在しないメソッドの呼び出しや、ClassMethod であると宣言されているメソッドの呼び出しを実行すると、<METHOD DOES NOT EXIST> というエラーを生じます。

### 例

以下は、\$ZOBJMETHOD を関数として使用する例です。

```
set ListOfStuff = ##class(%Library.ListOfDataTypes).%New()
for i = "First", "Second", "Third", "Fourth"
{
    do ListOfStuff.Insert((i _ "-Element"))
}
set MethodName = "Count"
set Elements = $ZOBJMETHOD(ListOfStuff, MethodName)
write "Elements: ", Elements, !
set i = $RANDOM(Elements) + 1
write "Element #", i, " = ", $ZOBJMETHOD(ListOfStuff, "GetAt", i), !
```

## \$ZOBJPROPERTY 内部関数

```
$ZOBJPROPERTY (Instance, Propertyname, Index1, Index2, Index3... )
```

### パラメータ

Instance	メソッド名を含むクラスのインスタンス
Propertyname	指定されたクラス内のプロパティの名前
index1, index2, index3, ...	2 番目の引数として指定されたプロパティが多次元である場合、index<n> は、プロパティを構成する配列へのインデックスとして扱われます。

### 概要

この関数は、Caché ObjectScript プログラムが、あるクラスの既存のインスタンス内の任意のプロパティの値を選択することを許可します。最初の引数はクラスのインスタンスでなければならないため、実行時に算出されます。プロパティ名は、実行時に算出されるか、文字列リテラルとして提供されません。文字列の内容は、クラスで宣言されたプロパティの名前と正確に一致する必要があります。

プロパティが多次元であると宣言されている場合は、propertyname の後の引数は、多次元配列へのインデックスとして扱われます。最大で 255 の引数をインデックスに使用できます。

\$ZOBJPROPERTY は割り当ての左側にも表示されます。このインスタンス内では、値が割り当てられる位置を提供します。

## パラメータ

### Instance

orefに評価する式。式の値は、希望するクラスのインメモリ・インスタンスの値でなければなりません。

### Propertyname

有効な文字列として評価する式。文字列の値は、Classname により特定される、クラス内で定義された既存のプロパティ名と一致する必要があります。

### Arg1, Arg2, Arg3, ...

Propertyname が多次元値である場合、この一連の式はプロパティが表す配列へのインデックスとして扱われます。

指定されたプロパティが多次元でない場合、余分な引数があると実行時にエラーが生じます。

## 備考

\$ZOBJPROPERTY が割り当て演算子の左にある場合、これは設定される位置で、右側にある場合は、計算で使用する値です。

メソッド内で、現在のインスタンスのプロパティを参照するために \$ZOBJPROPERTY を使用する場合は、Instance を省略できます。しかし、通常 Instance の後に続くコンマは必要です。

多次元と宣言されていないプロパティから多次元値を取得したり、そのプロパティへ多次元値を設定したりしようとすると、<FUNCTION> エラーが生じます。

## 例

以下は、\$ZOBJPROPERTY を関数として使用する例です。

```

set TestName = "%Library.File"
set ClassDef = ##class(%Library.ClassDefinition).%OpenId(TestName)
for i = "Name", "Super", "Persistent", "Final"
{
    write i, ": ", $ZOBJPROPERTY(ClassDef, i), !
}

```

以下は、割り当て演算子の右と左の両方で使用する例です。

```

set TestFile = ##class(%Library.File).%New("AFile")
write "Initial file name: ", $ZOBJPROPERTY(TestFile, "Name"), !
set $ZOBJPROPERTY(TestFile, "Name") = $ZOBJPROPERTY(TestFile, "Name")
    _ "Renamed"
write "File name afterward: ", $ZOBJPROPERTY(TestFile, "Name"), !

```

