HP OpenView AssetCenter

ソフトウェアバージョン：5.0

プログラマーズリファレンス

ビルド番号：328

利用規約

保証

HP製品およびサービスに対する保証は、当該製品またはサービスに付帯する明示的保証条項でのみ規定されます。

本規定のいかなる部分も、他の保証を構成すると解釈されるものではありません。

HPは本書の技術上または編集上の誤謬、欠落についての責任は負わないものとします。

本書に含まれる内容は、予告なく変更される場合があります。

限定保証条項

機密コンピュータソフトウェア。

所有、使用、コピーには、HPによる有効なライセンスが必要です。

FAR12.211および12.212準拠。商用コンピュータソフトウェア、コンピュータソフトウェアマニュアル、技術データは、ベンダの標準商用ライセンスに基づき、米国政府にライセンス供与されています。

著作権

商標

- Adobe®, Adobe Photoshop® and Acrobat® are trademarks of Adobe Systems Incorporated.
- Corel® and Corel logo® are trademarks or registered trademarks of Corel Corporation or Corel Corporation Limited.
- Java™ is a US trademark of Sun Microsystems, Inc.
- Linux is a U.S. registered trademark of Linus Torvalds
- Microsoft®, Windows®, Windows NT® and Windows® XP are U.S. registered trademarks of Microsoft Corporation.
- Oracle® is a registered US trademark of Oracle Corporation, Redwood City, California.
- UNIX® is a registered trademark of The Open Group.

目次

I はじめに

1 プログラミングの基本

この章では、AssetCenterで使用できるBASIC言語によるプログラミングの基本を説明します。プログラミングの経験があり、他の言語を使用したことがある方にとっては、この章の内容のほとんどは既知の事項のはずです。ただし、AssetCenterのBASICでは従来の機能のいくつかが意図的に廃止されていたり制限されていたりするので、この章全体に目を通しておくことをお勧めします。

変数の概要

変数は、プログラムの実行中にデータを格納しておくために用いられます。変数は以下の情報によって識別されます。

- 変数名。変数に格納された値を参照するために用いられます。
- 変数の型。変数に格納できるデータの種類を決定します。

一般に、変数は以下の2種類に分けられます。

- 配列。
- スカラ変数。これは配列以外のすべての変数を指します。

変数の宣言

変数は、使用する前に明示的に宣言する必要があります。宣言のシンタックスは次の通りです。

```
Dim <変数名> [As <変数の型>]
```

注意:

AssetCenterのBASICで変数の明示的宣言が必要なのは、Microsoft Visual BasicでOption Explicitキーワードを使用した場合と同じです。

変数名は以下の制約を満たす必要があります。

- 先頭は英大文字または英小文字。
- 長さは40文字以内。
- 使用できる文字は、英字A～Z、a～z、数字0～9、および下線文字（「_」）です。

  注意:

  アクセント付きの文字は使用可能ですが、なるべく使用しないでください。

- 予約されたキーワードは使用できません。例えば、BASICの関数や句の名前は予約されたキーワードです。

オプションのAs句により、定義する変数の型を指定できます。型は変数に格納できる情報の種類を示します。使用できるデータ型には、String（文字列）、Integer（整数）、Variant（バリアント）などがあります。

As句を省略した場合、変数はVariant型と見なされます。

単一宣言

単一宣言とは、1つの宣言文で1つの変数を宣言するものです。次の例を参照してください。

```
Dim I As Integer
Dim strName As String
Dim dNumber As Double
```

複合宣言

複合宣言とは、1つの宣言文で任意の数の変数を宣言するものです。次の例を参照してください。

```
Dim I As Integer, strName As String, dNumber As Double
Dim A, B, C As Integer
```

注意:

すでに説明したように、変数の型を指定しないと、変数はデフォルトでVariant型と見なされます。このため、上の例の2行目で、変数AとBの型はVariantであり、Cの型はIntegerです。

データ型

次の表は、関数またはパラメータで使用可能な型の一覧です。

型	意味
Integer	-32768～+32767の範囲の整数。
Long	-2147483647～+2147483646の範囲の整数。
Single	4バイトの浮動小数点数（単精度）。
Double	8バイトの浮動小数点数（倍精度）。
String	テキスト。すべての文字が使用可能。
Date	日付または日付+時刻
Variant	他のすべての型の代わりとなる汎用の型。

注意:

これらの型は外部ツールからは使用できません。使用できるのはLong、Doubleおよびstringだけです。Variantは存在せず、Integer型とDate型のオブジェクトはLongで表されます。

数値型

AssetCenterのBASIC言語では、数値型としてInteger（整数）、Long（長整数）、Single（単精度浮動小数点数）、Double（倍精度浮動小数点数）を用意しています。数値型はVariant型よりも使用メモリ量が少ないのが普通です。

変数に整数（123など）だけを格納し、小数（3.14など）を決して格納しない場合には、IntegerまたはLongとして宣言するのが適しています。これらのデータ型に対する演算は、他のデータ型よりも高速で、使用メモリ量も少なくてすみます。特にループのカウンタとしては、これらのデータ型が最適です。

変数に小数を格納する場合には、SingleまたはDoubleとして宣言します。

注意:

浮動小数点数（SingleまたはDouble）の演算では、丸め誤差が発生する場合があります。

文字列型

変数に文字列だけを格納する場合は、Stringとして宣言します。

```
Dim MyString As String
```

これにより、この変数に文字列を格納し、専用の文字列処理関数を使ってその内容を操作できるようになります。

```
MyString = "This is a string"
MyString = Right(MyString,6)
```

デフォルトでは、String型の変数は可変サイズです。文字列を格納するために割り当てられるサイズは、変数に代入されたデータのサイズによって変化します。ただし、次のシンタックスを使ってString型の変数を宣言することもできます。

```
Dim <変数名> As String * <格納する文字列のサイズ>
```

次の例では、20文字の文字列変数を宣言しています。

```
Dim MyString As String * 20
```

この変数に20文字よりも短い文字列を格納した場合、宣言したサイズになるまで文字列の末尾にスペースが追加されます。20文字よりも長い文字列を格納した場合、21文字目から後は切り捨てられます。

バリアント型

Variant型は、他のすべての型の代わりとなる汎用的な型です。他のデータ型とVariant型との間の変換は自動的に行われます。下の例を参照してください。

```
Dim MyVariant As Variant
MyVariant = "123"
MyVariant = MyVariant - 23
MyVariant = "Top " & MyVariant
```

変換は自動的に行われますが、以下の規則を守る必要があります。

- Variant型に対して算術演算を実行する場合、変数の内容は数値（文字列で表現されるものを含む）でなければなりません。
- Variant型に対して文字列連結演算を行いたい場合は、+演算子でなく&演算子を使用します。

Variantには2つの特殊な値を格納できます。空値とNull値です。

空値

Variant変数に初めて値を代入するまでは、変数の内容は空値です。これは特別な値であり、0、空文字列、Nullのどの値とも異なります。Variant変数の内容が空値かどうかをテストするには、BASIC関数**IsEmpty()**を使用します。次の例を参照してください。

```
Dim MyFirstVariant As Variant
Dim MySecondVariant As Variant
If IsEmpty(MyFirstVariant) Then MyFirstVariant = 0
MySecondVariant = 0
If IsEmpty(MySecondVariant) Then MySecondVariant = 123
```

空値を持つVariant変数を式で使用することは可能です。状況に応じて、値は0または空文字列と解釈されます。Variant変数に空値を再代入するには、キーワードEmptyを使用します。次の例を参照してください。

```
Dim MyVariant As Variant
MyVariant = 123
MyVariant = Empty
```

Null値

Null値は、データベースにおいて、存在しない値または未知の値を表すためによく用いられます。この値には以下のような特別な性質があります。

- Null値を含む式の値は常にNull値になります。これを、式の中でNull値が伝搬するといいます。式の一部がNullなら、式全体がNullとなります。
- 一般的に、関数のパラメータをNullに設定すると、関数の戻り値はNullになります。

データ配列

配列は、複数の変数を1つの名前で格納して参照するためのものです。配列の中の1つの変数を識別するには、番号（添字）を使います。配列の要素はすべて同じ型になります。StringとDoubleの両方の型を含む配列を作成することはできません。このような場合、Variant型を使用すれば、両方の値を格納する配列を作成できます。

配列の宣言

配列は変数の集合です。

配列に関して以下の規約が採用されています。

- 配列の下限：最初の要素の添字。

  注意:

  デフォルトでは、配列の下限は0です。

- 配列の上限：最後の要素の添字。

  注意:

  配列の上限は、Long型の最大値（2147483646要素）を超えることはできません。

配列の宣言は、変数の宣言に似ています。

```
Dim <配列名>(<配列の上限>) [As <配列に含まれる変数のデータ型>]
```

例：

```
Dim MyFirstArray(30) As String ' 31 elements
Dim MySecondArray(9) As Double ' 10 elements
```

次の宣言を使って、配列の下限を指定することもできます。

```
Dim <配列名>(<配列の下限> To <配列の上限>) [As <配列に含まれる変数のデータ型>]
```

例：

```
Dim MyFirstArray(1 To 30) As String ' 30 elements
Dim MySecondArray(5 To 9) As Double ' 5 elements
```

制限

AssetCenterのBASICでは、配列に以下の制限があります。

- 可変サイズの配列はサポートされません。特に、実行中に配列のサイズを変更することはできません。
- 多次元配列はサポートされません。

制御構造

制御構造とは、その名の示すとおり、プログラムの実行を制御するためのものです。制御構造には以下の2種類があります。

- 決定構造：何らかの条件に基づいてプログラムの行き先を決定するもの。
- ループ構造：何らかの条件に基づいてプログラムの特定の部分を反復するもの。

決定構造

決定構造は、テストの結果に基づいて特定の命令を条件付きで実行します。以下の決定構造が使用できます。

- **If...Then**
- **If...Then...Else...End If**
- **Select Case**

If...Then

この構造は、1つまたは複数の命令を条件付きで実行するために使用します。この構造のシンタックスでは、1行または複数行の文が使用できます。1行の文は1つの命令だけを実行できます。

```
If <条件> Then <命令>
```

```
If <条件> Then
<命令群>
End If
```

条件は一般に比較式ですが、結果が数字で表される式はどれでも使用できます。この値は次にBasicによりTrueまたはFalseとして解釈されます。Falseは数値0に該当し、その他の値はTrueとみなされます。

条件がTrueに評価されると、キーワードThenのあとの命令または命令群が実行されます。

If...Then...Else...End If

この構造は、複数の条件付き命令ブロックを定義するために使用します。最初にTrueに評価されたブロックだけが実行されます。

```
If <条件1> Then
<命令群1>
ElseIf <条件2> Then
<命令群2>
...
Else
<命令群N>
End If
```

最初の条件がテストされ、結果がFalseに評価された場合、2番目の条件がテストされ、以下同様に、どれかの条件がTrueに評価されるまで続きます。その条件のThenキーワードの後の命令群が実行されます。

Elseキーワードは省略可能です。これは、すべての条件がFalseに評価された場合に実行される命令群を指定するために使います。

注意:

この決定構造のElseIf命令群はいくつでも重ねることができます。ただし、同じ式を連続的に異なる値と比較する場合、決定構造が無意味に複雑になり、読みにくくなるおそれがあります。このような場合は、Select...Case決定構造を使用することをお勧めします。

Select...Case

この構造は上の決定構造と働きは同じですが、一般的にコードが読みやすくなる効果があります。Select...Case機能は、構造の開始時に1回だけテストを実行し、テスト結果を各Caseで指定された値と比較します。値が一致すると、そのCaseに対応する命令群が実行されます。

```
Select Case <テスト>
[Case <値リスト1>
<命令群1>]
[Case <値リスト2>
<命令群2>]
...
```

```
[Case Else
<命令群n>]
End Select
```

値リストは、カンマで区切った値のリストです。複数のCaseキーワードにテスト結果に一致する値がある場合、最初に一致したCaseに対応する命令群だけが実行されます。

CaseElseキーワードに対応する命令群は、Caseキーワード内に一致する値が見つからなかった場合に実行されます。

ループ構造

ループ構造は、一覧の命令を繰り返し実行するために使用します。以下のループ構造が使用できます。

- **Do...Loop**
- **For...Next**

Do...Loop

この構造は、一連の命令を不定の回数繰り返す場合に使用します。条件が満たされたとき、あるいは満たされなかったときにループが終了します。この条件は、False（0）またはTrue（0以外）に評価される値または式です。

注意:

命令群の中でExitDoを実行することにより、ループを強制的に終了させることができます。

この構造にはいくつかの形式がありますが、最も普通に用いられるのは次のものです。

```
Do While <Condition>
<Instructions>
Loop
```

この場合、最初に条件が評価されます。値がTrueなら、命令群が実行され、プログラムはDoWhileキーワードに戻り、もう一度条件をテストし、以下同様に続きます。条件がFalseに評価されるとループは終了します。

次の例は、カウンタの値をテストし、ループの1回の反復ごとにカウンタを1ずつ増やします。カウンタが20になるとループは終了します。

```
Dim iCounter As Integer
iCounter = 0
Do While iCounter < 20
iCounter = iCounter +1
Loop
```

次の例は前の例と似ていますが、カウンタの値が10になったときにExit Doキーワードでループを強制終了します。

```
Dim iCounter As Integer
iCounter = 0
Do While iCounter < 20
iCounter = iCounter +1
If iCounter = 10 Then Exit Do
Loop
```

この形式のDo...Loop構造では、命令群が実行される前に条件が評価されます。命令を実行してから条件をテストしたい場合は、次のDo...Loop構造を使用します。

```
Do
<命令群>
Loop While <条件>
```

注意:

この形式の構造では、命令群が少なくとも1回は必ず実行されます。

前の2つのDo...Loop構造では、条件がTrueである間反復が行われます。これに対して、条件がFalseの間反復したい場合、以下の構造を使用します。

```
Do Until <条件>
<命令群>
Loop

Do
<命令群>
Loop Until <条件>
```

この形式の構造を使うと、前の例は以下のように書くことができます。

```
Dim iCounter As Integer
iCounter = 0
Do Until iCounter = 20
iCounter = iCounter +1
Loop
```

For...Next

この構造は、一連の命令を不定の回数繰り返す場合に使用します。Do...Loopと異なり、For...Nextループではカウンタと呼ばれる変数を使用し、反復のたびにカウンタの値を加算または減算します。

注意:

命令群の中でExit Forを実行することにより、ループを強制的に終了させることができます。

```
For <カウンタ> = <初期値> To <最終値> [Step <増分値>]
<命令群>
Next [<カウンタ>]
```

重要項目:

引数カウンタ、初期値、最終値、増分値は、すべて数値です。

注意:

増分値は正または負の値です。増分値が正の場合、初期値が最終値以下でないと命令群は実行されません。増分値が負の場合、初期値が最終値以上でないと命令群は実行されません。増分値を指定しないとデフォルトで1に設定されます。

For...Nextループが実行されると、以下の動作が行われます。

1. **カウンタが初期化され、初期値が格納されます。**
2. **カウンタの値が最終値より大きいかどうかがテストされます。結果が真なら、ループは終了します。**

   注意:

   増分値が負の場合は、カウンタの値が最終値より小さいかどうかがテストされます。

3. **命令群が実行されます。**
4. **カウンタに1または指定された増分値が加算されます。**
5. **動作2から4までが繰り返されます。**

次の動作は、1000までのすべての偶数を合計します。

```
Dim iCounter As Integer, lSum As Long
For iCounter = 0 To 1000 Step 2
lSum = lSum + iCounter
Next
```

次の例は前の例と似ていますが、カウンタの値が500になったときにExit Forキーワードでループを強制終了します。

```
Dim iCounter As Integer, lSum As Long
For iCounter = 0 To 1000 Step 2
lSum = lSum + iCounter
```

```
If iCounter = 500 Then Exit For
Next
```

演算子

演算子とは、単純な演算（加算、乗算等）を変数に対して実行したり、変数を比較したりするときに使用する記号です。演算子には以下の種類があります。

- 代入演算子
- 算術演算子
- 関係演算子（比較演算子とも呼ぶ）
- 論理演算子

代入演算子

この種類の演算子は、変数に値を代入するために使用します。AssetCenterのBASICでは、代入演算子は「=」の1つだけです。代入のシンタックスは次の通りです。

```
<変数> = <値>
```

算術演算子

算術演算子は、変数の値を算術的に変更したり、2つの式に対して単純な算術演算を実行したりするために使用します。

+演算子

この演算子は、2つの値を加算するために用いられます。シンタックスは次の通りです。

```
<結果> = <式1> + <式2>
```

注意:

この演算子は、2つの値の加算と、2つの文字列の連結の両方に用いられます。あいまいさを避けるため、この演算子は加算だけに使い、文字列の連結には&演算子を使用することをお勧めします。

-演算子

この演算子は、2つの値の差を求めるため、または1つの値の符号を反転するため（単項演算子）に用いられます。この演算子には2通りのシンタックスがあります。

```
<結果> = <式1> - <式2>
```

または

```
- <式>
```

*演算子

この演算子は、2つの値を乗算するために用いられます。シンタックスは次の通りです。

```
<結果> = <式1> * <式2>
```

/演算子

この演算子は、2つの値を除算するために用いられます。シンタックスは次の通りです。

```
<結果> = <式1> / <式2>
```

^演算子

この演算子は、値の累乗を求めるために用いられます。シンタックスは次の通りです。

```
<結果> = <式1> ^ <式2>
```

注意:

このシンタックスでは、式2（指数）が整数の場合は式1が負の値を取ることはできません。式の中で複数の指数演算が連続して行われる場合、左から右の順番で論理的に解釈されます。

Mod演算子

この演算子は、2つの値のユークリッド除算の剰余を計算します。シンタックスは次の通りです。

```
<結果> = <式1> Mod <式2>
```

注意:

浮動小数点数は自動的に整数に丸められます。

次の例は4を返します（6.8は最も近い整数である7に丸められます）。

```
Dim iValue As Integer
iValue = 25 Mod 6.8
```

関係演算子

関係演算子は、2つの値を比較するために使用します。下の表は関係演算子の一覧です。

演算子	名称	説明	シンタックス
=	等価演算子	2つの値を比較し、等しいかどうかを調べます。	<式1> = <式2>
<	「小なり」演算子	ある値が別の値より厳密に小さいかどうかをテストします。	<式1> < <式2>
<=	「小なりまたは等しい」演算子	ある値が別の値より小さいかまたは等しいかどうかをテストします。	<式1> <= <式2>
>	「大なり」演算子	ある値が別の値より厳密に大きいかどうかをテストします。	<式1> > <式2>
>=	「大なりまたは等しい」演算子	ある値が別の値より大きいかまたは等しいかどうかをテストします。	<式1> >= <式2>
<>	不等価演算子	ある値が別の値と異なるかどうかをテストします。	<式1> <> <式2>

論理演算子

論理演算子は、複数の条件を評価するために使用します。

And演算子

この演算子は、2つの式の論理積（両方の条件が真）演算を実行します。シンタックスは次の通りです。

```
<結果> = <式1> And <式2>
```

すべての式（オペランド）がTrueに評価された場合、結果はTrueです。どちらかの式がFalseに評価された場合、結果はFalseです。

Or演算子

この演算子は、2つの式の論理和（どちらかの条件が真）演算を実行します。シンタックスは次の通りです。

```
<結果> = <式1> Or <式2>
```

どちらかの式がTrueに評価された場合、結果はTrueです。

Xor演算子

この演算子は、2つの式の排他的論理和（2つの条件が一方だけが真）演算を実行します。シンタックスは次の通りです。

```
<結果> = <式1> Xor <式2>
```

一方の式だけがTrueに評価された場合、結果はTrueになります。

Not演算子

この演算子は、式の論理否定を実行するために用いられます。シンタックスは次の通りです。

```
<結果> = Not <式1>
```

式がTrueに評価された場合、結果はFalseです。式がFalseに評価された場合、結果はTrueです。

演算子の優先順位

式の評価で複数の演算子を結合する場合、以下の優先順位が用いられます。演算子は優先順位の高いものから低いものへという順番で記載されています。

1 ()
2 ^
3 -, +
4 /, *
5 Mod
6 =, >, <, <=, >=
7 Not
8 And
9 Or
10 Xor

ファイル管理

AssetCenterのBASICには、簡単なファイル管理機能があります。一般的な操作（読取り、書込みなど）が標準で用意されています。

ファイルに関する注意事項

ファイルとは、プログラムから外部のオブジェクトを参照する手段です。ファイルは論理的なレコードの集合です。レコードは構造化されている場合もされてい

ない場合もあります。プログラムはレコードに対して基本的な操作（読取り、書込みなど）を実行できます。論理レコードは、1回の基本操作で処理できるデータの最小単位に相当します。

AssetCenterで処理できるのは、シーケンシャルファイルと呼ばれるものだけです。シーケンシャルファイルに対して実行できる主な操作は、次のレコードの読取りと、ファイルの末尾への新しいレコードの追加です。レコードの読取りと書込みを同時に行うことはできません。

読取りの場合、シーケンシャルファイルの最初の論理レコードにカーソルが置かれます。読取り操作を行うたびに、プログラムの内部領域（通常は変数）に1個のレコードが読み取られ、ファイル中の次のレコードにカーソルが移ります。読み取るレコードがまだ残っているかどうかを判定する操作（**EOF**（End Of File)句）が用意されています。

書込みの場合、シーケンシャルファイルは空か、ファイルの最後のレコードの後ろにカーソルが置かれます。書込み操作を行うたびに、プログラムの内部領域（通常は変数）のデータがファイル中のレコードに転送され、そのレコードの後ろにカーソルが移動します。

注意:

シーケンシャルファイルの主な特徴の1つは、レコードが書き込まれた順番で読み取られることです。

ファイルのオープンとクローズ

Open句

これはファイルを操作するための主要な機能です。ファイルの読取り、作成、書込みを実行できます。シンタックスは次の通りです。

```
Open <ファイルのパス> For <モード> [Access <アクセスタイプ>] As [#]<ファイル番号>
```

この句のパラメータを下の表に示します。

パラメータ	説明
<ファイルのパス>	操作の対象となるファイルを指定する文字列。この文字列にはファイルのフルパスを指定できます。

パラメータ	説明
<モード>	ファイルの処理モードを指定します。以下のいずれかの値を取ります。 ■ Input：ファイルは読取りモードでオープンされます。 ■ Output：ファイルは書込みモードでオープンされます。ファイルがすでに存在する場合、既存の内容は上書きされます。 ■ Append：ファイルは書込みモードでオープンされます。ファイルがすでに存在する場合、新しい内容はファイルの末尾に追加されます。 ■ Binary：ファイルはバイナリ読取りモードでオープンされます。
<アクセスタイプ>	オープンしたファイルに対して実行できる操作を指定します。ファイルが別のプロセスでオープンされており、指定したアクセスが許可されない場合、ファイルオープンコマンドは失敗します。このパラメータは以下のいずれかの値を取ります。 ■ Read：ファイルは読取り専用アクセスのためにオープンされます。 ■ Writeファイルは書込み専用アクセスのためにオープンされます。 ■ ReadWrite：ファイルは読取り／書込みモードでオープンされます。このアクセスタイプは、アクセスモードがBinaryおよびAppendのときだけ使用できます。
<ファイル番号>	ファイルを識別する1～511の固有の番号を指定します。**FreeFile()**関数を使えば、使用可能な次のファイル番号を知ることができます。

注意:

以下の点に注意してください。

- ファイルに対して読取りまたは書込み操作を行う前に、必ず**Open**句でファイルをオープンする必要があります。
- Append、Binary、またはOutputモードでは、参照したファイルが存在しない場合、新規作成されます。
- BinaryまたはInputモードの場合、ファイルをクローズせずに、別の番号を使って同じファイルをオープンすることが可能です。AppendまたはOutputモードの場合、ファイルをクローズしてから別の番号でオープンする必要があります。

Close句

Open()でオープンしたファイルをクローズします。シンタックスは次の通りです。

```
Close [<ファイルのリスト>]
```

オプションの<ファイルのリスト>引数は、1つまたは複数のファイル番号のリストです。このオプション引数のシンタックスは次の通りです。

```
[[#]<ファイル番号>][,[#]<ファイルのリスト>]...
```

注意:

このパラメータを省略すると、Open()句でオープンされたすべてのアクティブなファイルがクローズされます。

ファイルからのデータの読取り

ファイルからデータを読み取るには2つの句が使用できます。どちらを使用するかは、ファイルに対して指定されたアクセスモードによります。2つの句は以下の通りです。

- **Input**
- **Line Input**

Input句

BinaryまたはInputモードでオープンしたファイルから一定数の文字を読み取るために使用します。シンタックスは次の通りです。

```
Input (<読取り文字数>,[#]<ファイル番号>)
```

Line Input句

シーケンシャルファイルから1行を読取り、StringまたはVariant型の変数に格納します。シンタックスは次の通りです。

```
Line Input #<ファイル番号>, <変数名>
```

重要項目:

ファイル中の文字が1文字ずつ読み取られ、復帰文字または復帰文字+改行文字が読み取られると終了します。

ファイルへのデータの書込み

ファイルにデータを書き込むには、Print句だけが使用できます。シンタックスは次の通りです。

```
Print #<ファイル番号>, [<データ>]
```

2 関数の分類

関数は、次の3通りの方法で分類することができます。

関数の種類

AssetCenter環境で使用できる関数は、複数のグループに分類されます。

- AssetCenterで認識される関数：主にソフトウェアのスクリプト（BASIC）を記述できる部分で使います。
- AssetCenter APIで認識される関数：外部のツールから呼び出される関数、または高度言語で書くプログラムで使う関数です。

これらの関数は、それぞれ完全に独立しているわけではありません。例えば、AssetCenter API関数の中には、ソフトウェアのBASICスクリプトで使えるものがあります。このようにAssetCenterの内部BASICスクリプトでも使えるAssetCenter API関数のことを「公開されている」関数といいます。このような関数のシンタックスが変わることはありますが、関数で実行される処理は同じです。

関数の用途

本マニュアルで説明されている関数は、少なくとも次の場所の1つで使用可能です。

- AssetCenter APIライブラリ。これらの関数は、特にGet-Itアプリケーションの開発で使用可能です。
- フィールドやリンクの設定スクリプト（ポップアップメニューで［オブジェクトの設定］またはAssetCenter Database Administratorを選択します）、または特殊フィールドの［計算スクリプト］（SQL名：memScript）内。
  - デフォルト値
  - 必須属性
  - 履歴の保持
  - 読み取り専用属性
  - ...
- 「スクリプト」タイプのアクション
  - スクリプトアクションの［アクションのスクリプト］（SQL名：Script）フィールドで定義されるスクリプト
- AssetCenterウィザード
  - ウィザードの「FINISH.DO」スクリプト
  - ノードのプロパティの値を定義するスクリプト

アプリケーションモジュール

各関数は1つまたは複数のアプリケーションモジュールに関連付けられています。アプリケーションモジュールは、関数が実行する処理の性質を表します。以下のアプリケーションモジュールがあります。

- 組み込み：標準のBasic関数や変換関数、文字列の操作など
- テクニカル：データベースへの接続、テーブル、フィールド、リンク、インデックス、レコード、クエリの管理
- 業務：専門分野に関する一般的な関数
- ケーブル
- 調達
- 経費付替え
- ウィザード
- アクション
- グラフィックス

3 表記法と形式

本章では、次の事項について説明します。

表記法

関数のシンタックスと用例の表記法は以下の表の通りです。

[]	大括弧は、オプションのパラメータを示します。実際にコマンドのパラメータを入力するときは、大括弧は必要ありません。 ただし、BASICスクリプトでは、次の例のように、データへのパスを大括弧で囲みます。 [Link.Link.Field]
<>	山形括弧は、パラメータの短い説明（英語の略語）を示します。実際にパラメータを入力する時は、山形括弧を使わずに、括弧内にあるテキストに該当する情報のみを入力してください。
{}	中括弧は、ノードの定義または1プロパティ用の複数言語のスクリプトブロックの定義を囲みます。
|	縦線（パイプ文字）は、中括弧に囲まれた複数のパラメータを区切る場合に使います。

特別な意味のあるテキストの表記法は以下の通りです。

固定幅の文字	DOSコマンド、関数のパラメータ、および日付形式
例	コードやコマンドの例
...	コードまたはコマンドの省略
［オブジェクト名］	フィールド、タブ、メニュー、キー名など

スクリプト内での「日付+時間」型定数のフォーマット

ユーザが定義している表示形式に関係なく、スクリプトで日付を記述するときは、次のような国際標準形式を使います。

yyyy/mm/dd hh:mm:ss

例

```
RetVal="1998/07/12 13:05:00"
```

注意:

年月日の区切りに、ハイフン（-）を使うこともできます。

日付について

内部BASICでの日付と外部ツールからの日付は形式が異なります。

- **BASICでは、国際標準形式、またはDouble（倍精度）型の浮動小数点数で記述します。後者の場合、整数部分は1899年12月30日の午前0時から数えて現在まで経過した日数を表し、小数部分は本日の午前0時から現在まで経過した時間の１日（86400秒）に対する割合（現在までの経過秒数を86400で割ったもの）を表します。**
- **UNIXでは、Long（倍長）型の整数で記述します。現地時間に関係なく、UTC（協定世界時）で1870年1月1日の午前0時から現在までに経過した秒数を示します。**

Duration（時間）定数の形式

スクリプトでは、時間を秒単位で記述して保存します。例えば、Duration型フィールドのデフォルト値を3日に設定するには、次のスクリプトを使います。

```
RetVal=259200
```

同様に、AmWorkTimeSpanBetween()などの時間を計算する関数も、秒単位の時間を返します。

注意:

AssetCenterの会計計算では、最も一般的な計算方法を使います。この方法では、1年は12ヶ月、1月は30日として計算するので、1年は360日になります。

4 定義

本章では、主要な用語の定義について説明します。

以下の用語の定義を説明します。

関数の定義

「関数」とは、なんらかの処理を実行し、値をユーザに返すプログラムです。返される値を「戻り値」または「戻りコード」といいます。

AssetCenterの内部関数を呼び出すシンタックスの例は次の通りです。

AmConvertCurrency(strSrcName As String, strDstName As String, dVal As Double) As Double

上記と同じ関数をAssetCenter APIから呼び出すシンタックスは次の通りです。

double AmConvertCurrency(long hApiCnxBase,long ltm, const char *pszSrcName, const char *pszDstName,double dVal)

CurrentUser仮想リンクの定義

定義

CurrentUserは、すべてのテーブルから出発し、現在のユーザに対応する部署と従業員テーブル内のレコードに到達するリンクと考えられます。

- **CurrentUserという形式を使うと、仮想リンクは現在のユーザのレコードを検索し、［会社の部署と従業員テーブル］の記述文字列を返します。**
- **CurrentUser.Fieldという形式を使うと、現在のユーザのフィールドの値を返します。**

注意:

仮想リンクは、フィールドとリンクのリストには表示されません。そのため、CurrentUser**は、**AssetCenter**内のスクリプトの表示／編集用ウィンドウには直接表示されません。手動で「**CurrentUser**」と入力する必要があります。**

等価関数

AmLoginName()関数とAmLoginId()関数は、それぞれ現在のユーザの「名前（SQL名：Name）」と「ID（SQL**名：**lPersId）」を返し、CurrentUserから派生した関数と見なされます。これらの関数には、次のような関係が成り立ちます。**

- AmLoginName()=[CurrentUser.UserLogin]
- AmLoginId()=[CurrentUser.lEmplDeptId]

制限

CurrentUserはコンテキストが定義されている場合のみ機能します（コンテキストはテーブルであること）。

コンテキストがない場合、別の関数を使用してください。

例：

コンテキスト外アクションを作成し、AssetCenter**データベースに接続するユーザに依存するパスを持つファイルを実行することができます。**

状況依存アクションの場合、［フォルダ］フィールドが例えば c:\scripts\[CurrentUser.Name]**に設定された実行可能なタイプアクションを作成できます。**

ただし、実行可能なタイプアクションにコンテキストがない場合、[CurrentUser.Name]**は固定テキストとみなされます。**

したがって、スクリプトを使用してスクリプトタイプのコンテキスト外アクションを作成するなど、別の解決策を見つける必要があります。

```
RetVal = amActionExec("program.exe","c:\scripts\" + amLoginName())
```

ハンドルの定義

「ハンドル」とは、オブジェクト固有の識別子です。AssetCenter**では、フィールド、リンク、インデックス、クエリ、レコード、テーブル、または接続のことをオブジェクトと呼びます。ハンドルは、**32**ビット整数型（**Long**型）の値を取ります。**

注意:

NULL**は、有効なハンドルの値ではありません。**

外部ツールからも（データベースの）接続ハンドルにアクセスできます。

エラーコードの定義

関数を正しく実行できなかった場合は、エラーコードが返ります。

外部ツールの場合

外部ツールでは、AmLastError()**関数と**AmLastErrorMsg()**関数を使って、それぞれエラーコードとそのコードに付いているエラーメッセージを取得することができます。エラーをクリアするには、**AmClearLastError()**関数を使います。**

注意:

新しい関数を呼び出すと、常にその前に発生したエラーとエラーメッセージがクリアされます。

内部

内部（Basic**スクリプト内など）では、最後のエラーコードとエラーの説明を、Err.Number関数とErr.Description関数を使って取得します。**

注意:

内部的にはエラーの管理をプログラムする必要はありません。問題のあるスクリプトは停止し、必要に応じてデータベースのロールバックが実行されます。

Err.Raise関数を使用すると、エラーメッセージを故意に発生させることができます。シンタックスは以下の通りです。

```
Err.Raise (<エラー番号>, <エラーメッセージ>)
```

注意:

レコードの作成や変更が、テーブルの「有効性」に関するフィールド値により無効になる場合は、Err.Raise関数でエラーメッセージを発生させ、ユーザに知らせる方が賢明です（コード12006または12007）。エラーメッセージがないと、ユーザはレコードの変更や作成が不可能な理由を理解できません。

頻繁に使用されるエラーコードは以下の表の通りです。

エラーコード	説明
12001	未定義のエラー
12002	関数用の間違ったパラメータ
12003	無効なハンドルまたは削除されるオブジェクト
12004	使用可能なデータがない場合。このエラーは特にクエリの実行時に発生します。クエリの結果がデータを返さない場合に、このエラーが発生します。
12005	データベースサーバの内部エラー
12006	無効な値（パラメータに不適切なタイプ、など）
12007	無効なレコード（必須フィールドに値が入力されていない、など）
12008	データベースへのアクセス権限上の問題
12009	古い関数または実施されていない関数
12010	データベースへの接続最大数を超えた場合

5 関数とパラメータのデータ型

本章では、次の事項について説明します。

データ型のリスト

関数とパラメータで使用可能なデータ型とその説明は、以下の表の通りです。

データ型	説明
Integer	-32,768～+32,767の整数
Long	-2,147,483,647～+2,147,483,646の整数
Single	4バイトの浮動小数点数（単精度）。
Double	8バイトの浮動小数点数（倍精度）。
String	任意の文字からなるテキスト
Date（日付）	日付または日付+時刻

データ型	説明
Variant（可変）	汎用の任意の型

注意:

これらのデータ型の一部は外部ツールからは使用不可能です。Long、DoubleとString型のみが使用可能です。Variant型は存在しません。また、IntegerとDate型のオブジェクトはLong型で表現されます。

関数の型

関数の型は、その関数の返す値の型に対応しています。プログラムのコンパイルエラーやランタイムエラーが発生するのを防ぐために、正しい型の関数を適切な場所で使う必要があります。

例えば、フィールドのデフォルト値を定義する時に、そのフィールドのデータ型と異なる型の値を返す関数を使うことはできません。例えば、下の関数を使ってDate（日付）型やDate+Time（日付+時刻）型のフィールドのデフォルト値を定義すると、エラーが発生します。

```
RetVal=AmLoginName()
```

これは、「AmLoginName()」関数は、接続されているユーザの名前を文字列（String（文字列）型）で返すためです。文字列型の戻り値は、「Date（日付）」型や「Date+Time（日付+時刻）」型のフィールドでは使えないため、エラーメッセージが表示されます。

パラメータの型

関数で使うパラメータには型が決まっており、関数を正しく実行するためには、正しい型のパラメータを使用する必要があります。関数のシンタックス内では、パラメータの型を示す接頭辞がパラメータの先頭に付いています。このリファレンスに記載されている接頭辞は、関数のシンタックス（APIまたはBASIC）によって異なることに注意してください。それぞれAPIシンタックスとBASICシンタックスで使われている接頭辞は、次の表の通りです。

型	API関数のシンタックスで使われている接頭辞	BASIC関数のシンタックスで使われている接頭辞
Integer（整数）	"i"	"i"
Long（倍長整数）	h（ハンドル）l（数値）\n	"l"
Double（倍精度）	"d"	"d"
String（文字列）	"char*psz"	"str"

型	API関数のシンタックスで使われている接頭辞	BASIC関数のシンタックスで使われている接頭辞
Date（日付）	"ltm"	"dt"
Variant（可変）	"v"	"v"

II　APIの使用

6 はじめに

AssetCenter APIは、Windows 95/98、2000、XP、およびServer 2003上で使用可能な32ビットDLLとして用意されています。

次の環境がサポートされています。

- Visual Basic 4.0、5.0、および6.0
- Visual C++ 4.0、5.0、および6.0
- Visual Basic .NET 2002および2003
- Visual Studio .NET 2002および2003
- Visual C# .NET 2002および2003
- VBA（Visual Basic for Applications）を使用するMicrosoft全製品

警告:

ライブラリ（.dll）のエントリポイントは、.NET環境には用意されていません。これらの開発環境を使用する場合は、これらのエントリポイントをユーザ自身で定義する必要があります。

注意:

APIは、外部DLLの使用を許可するすべてのツールと互換性があります。

注意事項

AssetCenter APIを使う前に、AssetCenterの概念と用語を理解しておくことをお勧めします。特に、データベースの構造に関する知識が必要です。

データベースの構造の詳細は、AssetCenterのインストール先フォルダの「doc\infos」サブフォルダにあるマニュアル『管理』の、「データベースの標準記述ファイル」の章、および「Database.txt」と「Tables.txt」ファイルを参照してください。

インストール

AssetCenter APIを使用する前に、AssetCenterの完全インストールを実行してください。これにより、使用しているコンピュータからデータベースにアクセスできるかどうかを簡単に確認できるようになり、またデータベースの作成や設定が可能になります。APIは、AssetCenterと同じデータベース層と設定情報を使って、データソースにアクセスします。このため、APIで問題が発生した場合は、AssetCenterのGUIで問題の原因を探すことができます。

AssetCenterでの開発環境を設定するための一般的な手順は、次の通りです。

- AssetCenterの32ビットバージョンとAssetCenter APIパッケージをインストールします。
- AssetCenterを使って、データソースを設定し、データベースを開きます。
- 開発環境で、AssetCenter API関数を呼び出します。

デモ用データベースか、エラーが発生してデータが壊れても差し支えないデータソースを使って、AssetCenter APIを試用してください。

DLLに関連付けられた「.ini」構成ファイル

▶『AssetCenter; - インスタレーション』マニュアルの「.iniおよび .cfgファイル」の章

特に以下のセクションを参照してください。

- 使用可能な「.ini」および「.cfg」ファイル
- .iniファイルの変更を管理する

7 操作手順

AssetCenter APIを使用した一連のな処理手順は以下の通りです。

ステップ 1 AQL文を使ってクエリを作成します。

```
SELECT AssetTag, User.Name, Supervisor.Name FROM
amPortfolio
```

注意:

AssetCenter Exportを使ってAQLクエリを作成することもできます。

ステップ 2 クエリの結果を検索し、特定の項目の必要なハンドルを取得します。

ステップ 3 取得したハンドルを使って、レコード情報を更新します。

ステップ 4 トランザクション全体をコミット（完了）するか、ロールバック（キャンセル）します。

8　基本概念とプログラムの例

本章では、次の事項について説明します。

基本概念

AssetCenterは、オブジェクト指向で設計されており、このためAPIもオブジェクト指向型の構造です。Windows DLLには、単層構造の「C言語型」APIを使用するという制約点がありますが、AssetCenter APIは、この問題を回避するためにハンドル（32ビット整数）を使ってユーザが作成した各オブジェクトを識別します。このため、オブジェクト指向言語以外のプログラムからでも、AssetCenterオブジェクトモデルにアクセスできるようになっています。

使用するプログラムは、AssetCenter DLLを初期化するために、最初にAmStartUp()関数を呼び出さなければなりません。また、最後にAmCleanUp()関数を呼び出して終了する必要があります。

データベースオブジェクトにアクセスする前に、ユーザとデータベースの接続を確立する必要があります。この接続は、「接続」オブジェクトの「ハンドル」で識別されます。このハンドルは、その後データベースと相互作用するすべてのAPI関数で使うことができます（ハンドルは「hApiCnxBase」パラメータで指定しま

す。）「接続」オブジェクトは、クエリを作成し、レコードにアクセスするのに使用されます。

注意:

すべてのデータベースオブジェクトが接続オブジェクトにリンクされるので、ユーザのアクセス権限情報をチェックできます。

接続が確立したら、まず有効なデータソース名とログイン名／パスワードを使って、接続を開きます。

警告:

AssetCenter APIを介してAssetCenterデータベースに接続するときは、スロットを使います。

日付と時刻の処理

Date（日付）型とDate+Time（日付+時刻）型のフィールドの値を読み取るには、次のいずれかの関数を使います。

- AmGetFieldLongValue()：UNIXで使うLong（倍長整数）型のUTC（協定世界時）を返します。
- AmGetFieldStrValue()：Windowsのコントロールパネルで設定されている形式と同じ文字列を返します。Windowsで指定しているタイムゾーンに合わせた日付と時刻が返ります。戻り値を表示する必要がある場合には、この関数を使うことをお勧めします。

プログラムの例（1）

デモ用データベースへの接続を作成するC言語プログラムの例は、次の通りです。

```
long lCnx ;
lCnx = AmOpenConnection(ACDemo351ENG, Admin , ) ;
```

この例のlCnxは、新しく作成する接続の識別に使用される接続ハンドルです。

次に、作成した接続を使ってクエリを作成したり、データベースにアクセスしたりできます。特定の資産を検索するC言語プログラムの例は、以下の通りです。

```
#include apiproto.h
#define SZ_MODEL_LEN 200
long lCnx ;
```

```
long lQuery ;
long lStatus ; /* to store error code */
char szModel[SZ_MODEL_LEN] ;
/* dll initialization */
AmStartup();
/* Open a connection */
lCnx = AmOpenConnection("ACDemo300Eng","Admin" ,"") ;
if( lCnx != 0 )
{
/* Creation of a query object */
lQuery = AmQueryCreate (lCnx)
if( lQuery != 0 )
{
/* Construction of  the result set : all assets from Compaq*/
lStatus = AmQueryExec(lQuery, "select AssetTag where brand = 'Compaq'")
/* Navigates through the result set */
while( !lStatus )
{
/* Read the first field (AssetTag) of the current item in the query */
lStatus = AmGetFieldStrValue(lQuery,0,szModel,SZ_MODEL_LEN-1);
if( lStatus == 0 )
{
printf(' Compaq AssetTag=%s\n',szModel);
lStatus = AmQueryNext(lQuery);
}
}
/* clean things up */
AmReleaseHandle(lQuery);
}
AmCloseConnection(lCnx);
}
AmCleanup();
```

プログラムの例（2）

データベース内のオブジェクトを見つけるにはクエリを使います。レコードを更新するには、まずクエリを使って"record"**オブジェクトのハンドルを取得する必要があります。次にそのレコードを他の**AssetCenter API**関数で処理します。**

特定のレコードハンドルを取得して、フィールドを変更するプログラムの例は、以下の通りです。

```
/* Handles for objects */
long lCnx ;
long lQuery ;
long lStatus ;
long lRecord ;
AmStartup();
lCnx = AmOpenConnection("ACDemo300Eng","Admin" ,"") ;
/* Creation of a query object attached to lCnx */
lQuery = AmQueryCreate(lCnx);
/* Mark the starting point of the current transaction */
AmStartTransaction(lCnx);
/* Use a query that matches a single object */
lStatus = AmQueryGet(lQuery, "select model, AssetId where brand = 'Compaq' and bar
code='34234'") ;
/* Get a record handle to the matching object */
lRecord = AmGetRecordHandle(lQuery) ;
/* Change the field Field1 with new value spam */
lStatus = AmSetFieldStrValue(lRecord, "Field1", "Spam");
/* Update the change for the current session */
lStatus = AmUpdateRecord(lrecord);
/* Commit all modifications to the database */
lStatus = AmCommit(lCnx) ;
/* you can release here query and record objects */
/* but closing connection will do it */
/* Close the connection to the database */
AmCloseConnection(lCnx);
AmCleanup();
```

この例では、クエリを使ってレコード特有の識別子（ハンドル）を取得する方法を示しています。ここで使用したクエリでは、オブジェクトを1つだけ検索しますが、AmQueryExec()を使うと、複数のレコードを検索して1つまたは複数のレコードハンドルが返るようにすることもできます。

注意:

この例を簡単にするため、発生する可能性のあるエラーコードは含まれていません。

III 関数の説明

9 関数の説明

Abs()

数値の絶対値を返します。

内部Basicシンタックス

Function Abs(dValue As Double) As Double

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **dValue：絶対値を取得する数値**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

例

```
Dim iSeed as Integer
iSeed = Int((10*Rnd)-5)
RetVal = Abs(iSeed)
```

AmActionDde()

DDEサーバアプリケーションにDDE要求を送ります。この関数を使うと、DDEリンクにより、AssetCenterから他のアプリケーションをコントロールできます。この関数はDDEタイプのアクションに相当します。

APIシンタックス

long AmActionDde(long hApiCnxBase, char *strService, char *strTopic, char *strCommand, char *strFileName, char *strDirectory, char *strParameters, char *strTable, long lRecordId);

内部Basicシンタックス

Function AmActionDde(strService As String, strTopic As String, strCommand As String, strFileName As String, strDirectory As String, strParameters As String, strTable As String, lRecordId As Long) As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓

	使用
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strService：呼び出す実行可能ファイルのDDEサービスの名前。実行可能ファイルのDDEサービスの一覧については、その実行可能ファイルのマニュアルを参照してください。**
- **strTopic：実行するDDEアクションのトピック（コンテキスト）**
- **strCommand：実行する外部 アプリケーションのコマンド。外部 アプリケーションで定義されているシンタックスに従って指定する必要があります。**
- **strFileName：サービスがメモリにない場合は、サービスを起動する実行可能ファイルの名前（または、Windowsのファイルマネージャで実行可能ファイルとして設定されているファイルの名前）をこのパラメータで指定して、サービスを起動する必要があります。**
- **strDirectory：strFileNameで指定したファイルのパス**
- **strParameters：サービスの起動時にサービスをアクティブにする実行可能ファイルに渡す様々なパラメータ**
- **strTable：アクションのコンテキストを指定する時に追加するパラメータ。アクションを適用するレコードが入っているテーブルのSQL名を示します。**
- **lRecordId：アクションが状況依存である場合に使用する、オプションのパラメータ。アクションを適用するレコードの識別子を指定します。**

戻りコード

- **0：成功**
- **0以外：エラーコード**

AmActionExec()

「.exe」、「.com」、「.bat」、「.pif」のいずれかのアプリケーションを起動します。また、Windowsのファイルマネージャで実行可能ファイルとして設定している拡張子の文書を参照できます。この関数は、「実行可能ファイル」タイプのアクションに相当します。

APIシンタックス

long AmActionExec(long hApiCnxBase, char *strFileName, char *strDirectory, char *strParameters, char *strTable, long lRecordId);

内部Basicシンタックス

Function AmActionExec(strFileName As String, strDirectory As String, strParameters As String, strTable As String, lRecordId As Long) As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **strFileName：実行可能ファイルまたはドキュメント（ファイルマネージャで実行可能ファイルとして設定されている場合）の名前**
- **strDirectory：strFileNameに指定されているファイルへのパス。**
- **strParameters：実行可能ファイルの起動時に実行可能ファイルに提供する各種のパラメータ（必要な場合のみ）**
- **strTable：アクションのコンテキストを指定する時に追加するパラメータ。アクションを適用するレコードが入っているテーブルの**SQL**名を指定します。**
- **lRecordId：アクションが状況依存である場合に使用する、オプションのパラメータ。アクションを適用するレコードの識別子を指定します。**

戻りコード

- 0**：成功**
- 0**以外：エラーコード**

例

「C**」ドライブの「**WinNT**」ディレクトリにある**Windows NT Explorer**を実行します。**

```
RetVal = AmActionExec("explorer.exe", "c:\winnt\")
```

AmActionMail()

AssetCenterが管理する以下のメッセージシステムのうちのいずれか1つを介してメッセージを送信します。

- 内部メッセージシステム
- VIM規格をベースとする外部のメッセージシステム（Lotus Notesなど）
- MAPI規格をベースとする外部のメッセージシステム（Microsoft Exchange、Microsoft Outlookなど）
- SMTP規格をベースとする外部のメッセージシステム（インターネット規格）

APIシンタックス

long AmActionMail(long hApiCnxBase, char *strTo, char *strCc, char *strCcc, char *strSubject, char *strMessage, long iPriority, long bAcknowledge, char *strRefObject, char *strTable, long lRecordId);

内部Basicシンタックス

Function AmActionMail(strTo As String, strCc As String, strCcc As String, strSubject As String, strMessage As String, iPriority As Long, bAcknowledge As Long, strRefObject As String, strTable As String, lRecordId As Long) As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strTo：このパラメータには、メッセージの受取人のアドレスのリストが、「メッセージシステム：アドレス」の形で列挙されています。セミコロンが区切り文字として使用されています。**
- **strCc：メッセージのCarbon Copyを送信するアドレスのリスト。アドレスをコンマ（:）で区切ります。**
- **strCcc：メッセージのBlind Carbon Copyを送信するアドレスのリスト（受取人のリストには表示されません）。アドレスをコンマで区切ります。**
- **strSubject：メッセージの件名**
- **strMessage：メッセージの本文**
- **iPriority：メッセージ送信の優先度を指定します。**
  - 0：低
  - 1：普通
  - 2：高
- **bAcknowledge：メッセージの送信者が、受信者から通知を受け取るかどうかを指定します。**
  - 0：送信者は通知を受け取りません。
  - 1：送信者は通知を受け取ります。
- **strRefObject：AssetCenterの内部メッセージシステムを介して送信されたメッセージにのみ使用します。参照されたオブジェクトに達するため、実行コンテキストに対応するレコードから追跡するリンクのSQL名が、このパラメータには含まれています。CurrentUser仮想リンクを使用することも可能です。**
- **strTable：アクションのコンテキストを指定する時に追加するパラメータ。アクションを適用するレコードが入っているテーブルのSQL名を指定します。**
- **lRecordId：アクションが状況依存である場合に使用する、オプションのパラメータ。アクションを適用するレコードの識別子を指定します。**

戻りコード

- 0：成功
- 0以外：エラーコード

AmActionPrint()

データベースで指定したレコードのレポートを印刷します。

内部Basicシンタックス

Function AmActionPrint(lReportId As Long, lRecordId As Long) As Long

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **lReportId：印刷するレポートの識別子。**
- **lRecordId：レポートを印刷するレコードの識別子。デフォルトでは、このパラメータは「**0**」に設定されています。関連するテーブルはレポートが暗黙に定義しています。**

戻りコード

- 0**：成功**
- 0**以外：エラーコード**

AmActionPrintPreview()

データベースのレコードに関するレポートの印刷プレビューを起動します。

内部Basicシンタックス

Function AmActionPrintPreview(lReportId As Long, lRecordId As Long) As Long

用途

バージョン：3.60

使用	
AssetCenter API	
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lReportId：関連するレポートの識別子。**
- **lRecordId：レポートに関するレコードの識別子。このパラメータのデフォルト値は「0」です。**

戻りコード

- 0：**成功**
- 0**以外：エラーコード**

AmActionPrintTo()

データベースレコードと、プリンタについてのレポートを印刷します。

内部Basicシンタックス

Function AmActionPrintTo(strPrinterName As String, lReportId As Long, lRecordId As Long) As Long

用途

バージョン：3.5

使用	
AssetCenter API	
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strPrinterName：印刷を実行するプリンタの名前。**
- **lReportId：印刷するレポートの識別子。**
- **lRecordId：レポートに関するレコードの識別子。このパラメータのデフォルト値は「0」です。**

戻りコード

- 0：**成功**
- 0**以外：エラーコード**

AmAddAllPOLinesToInv()

既存の請求書に1件の発注を追加します。

APIシンタックス

long AmAddAllPOLinesToInv(long hApiCnxBase, long lPOrdId, long lInvId);

内部Basicシンタックス

Function AmAddAllPOLinesToInv(lPOrdId As Long, lInvId As Long) As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lPOrdId：請求書に追加する発注の識別子。**
- **lInvId：発注を追加する請求書の識別子。**

戻りコード

- **0：成功**
- **0以外：エラーコード**

AmAddCatRefAndCompositionToPOrder()

ある発注にカタログリファレンスの内容を全部追加します。

APIシンタックス

long AmAddCatRefAndCompositionToPOrder(long hApiCnxBase, long lPOrderId, long lCatRefId, double dCatRefQty, long lRequestId, double dUnitPrice, char *strCur);

内部Basicシンタックス

Function AmAddCatRefAndCompositionToPOrder(lPOrderId As Long, lCatRefId As Long, dCatRefQty As Double, lRequestId As Long, dUnitPrice As Double, strCur As String) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- lPOrderId：記入する発注の識別子。
- lCatRefId：カタログリファレンスの識別子。
- dCatRefQty：追加する数量（製品に関連付けられた単位で）
- lRequestId：この発注の元となる依頼の識別子。
- dUnitPrice：カタログリファレンスの製品の単価
- strCur：単価を表記するための通貨のコード

戻りコード

- 0：成功
- 0以外：エラーコード

注

注意:

発注に記入するために、カタログリファレンスの製品構成を使用します。

AmAddCatRefToPOrder()

カタログリファレンスを既存の発注に追加します。

APIシンタックス

```
long AmAddCatRefToPOrder(long hApiCnxBase, long lRequestLineId, long lCatRefId, long lPOrderId, double dQty, long bCanMerge);
```

内部Basicシンタックス

```
Function AmAddCatRefToPOrder(lRequestLineId As Long, lCatRefId As Long, lPOrderId As Long, dQty As Double, bCanMerge As Long) As Long
```

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lRequestLineId：発注に関連する依頼明細の識別子。**
- **lCatRefId：追加するカタログリファレンスの識別子。**
- **lPOrderId：演算の適用先の発注の識別子。**
- **dQty：追加する数量（製品に関連付けられた単位で）**
- **bCanMerge：追加明細を発注内に既存する明細と結合するかどうかを指定します。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献 301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmAddEstimLinesToPO()

既存の発注に、1件の見積の全見積明細を追加します。

APIシンタックス

long AmAddEstimLinesToPO(long hApiCnxBase, long lEstimId, long lPOrdId, long bMergeLines);

内部Basicシンタックス

Function AmAddEstimLinesToPO(lEstimId As Long, lPOrdId As Long, bMergeLines As Long) As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **lEstimId：発注に追加する見積の識別子。**
- **lPOrdId：**1**件の見積の全見積明細を追加する発注の識別子。**
- **bMergeLines：同じ依頼明細（bMergeLines**=1**）を**1**つの明細に統合するかどうかを指定できます。統合する明細の数量を合計して、**1**つの明細を作成します。**

戻りコード

- 0**：成功**
- 0**以外：エラーコード**

AmAddEstimLineToPO()

既存の発注に、1**個の見積明細を追加します。**

APIシンタックス

long AmAddEstimLineToPO(long hApiCnxBase, long lEstimLineId, long lPOrdId, long bMergeLines);

内部Basicシンタックス

Function AmAddEstimLineToPO(lEstimLineId As Long, lPOrdId As Long, bMergeLines As Long) As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **lEstimLineId：発注に追加する見積明細の識別子。**
- **lPOrdId：見積明細を追加する発注の識別子。**
- **bMergeLines：同じ依頼明細（bMergeLines**=1**）を**1**つの明細に統合するかどうかを指定できます。統合する明細の数量を合計して、**1**つの明細を作成します。**

戻りコード

- 0**：成功**
- 0**以外：エラーコード**

AmAddLicContentToRequest()

この関数は、ライセンスの対象となるすべてのソフトウェアインストールを購入依頼に追加します。

APIシンタックス

long AmAddLicContentToRequest(long hApiCnxBase, long lRequestId, long lLicModelId, long lParent, long bExternalParent);

内部Basicシンタックス

Function AmAddLicContentToRequest(lRequestId As Long, lLicModelId As Long, lParent As Long, bExternalParent As Long) As Long

用途

バージョン：?

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **lRequestId：操作に関連する購入依頼の識別子。**
- **lLicModelId：ライセンスモデルの識別子。**
- **lParent：作成した依頼明細の親になるポートフォリオ品目または依頼明細の識別子。**
- **bExternalParent：このパラメータを「**1**」に設定した場合は、作成された明細の親は依頼明細の既存のポートフォリオ品目です。**

戻りコード

- 0**：成功**
- 0**以外：エラーコード**

注

注意:

この関数は互換性のためだけに用意されています。この関数の使用は推奨されません。

AmAddPOLineToInv()

1つの発注明細の数量のうち、指定した数量だけを1件の請求書に追加します。

APIシンタックス

```
long AmAddPOLineToInv(long hApiCnxBase, long lPOrdLineId, long lInvId, double dQty);
```

内部Basicシンタックス

```
Function AmAddPOLineToInv(lPOrdLineId As Long, lInvId As Long, dQty As Double) As Long
```

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- lPOrdLineId：請求書に追加する発注明細の識別子。
- lInvId：発注明細の物件を追加する請求書の識別子。
- dQty：請求書に追加する発注明細の物件の数量

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

AmAddPOrderLineToReceipt()

受領に発注明細を追加します。これにより、既存の受領内で発注明細を受領できます。

APIシンタックス

long AmAddPOrderLineToReceipt(long hApiCnxBase, long lPOrderLineId, long lRecptId, double dQty, long bCanMerge);

内部Basicシンタックス

Function AmAddPOrderLineToReceipt(lPOrderLineId As Long, lRecptId As Long, dQty As Double, bCanMerge As Long) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lPOrderLineId：発注明細の識別子。**
- **lRecptId：影響を受ける受領の識別子。**
- **dQty：受領する数量。発注した数量の内、受領する数量を指定できます（製品の単位で）。**
- **bCanMerge：このパラメータでは、受領内に既存の明細に明細を結合するかどうかを指定できます。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**

- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

AmAddReceiptLineToInvoice()

請求書に受領明細を追加します。これにより、既存の請求書内で受領明細を請求できます。

APIシンタックス

long AmAddReceiptLineToInvoice(long hApiCnxBase, long lRecptLineId, long lInvoiceId, double dQty, long bCanMerge);

内部Basicシンタックス

Function AmAddReceiptLineToInvoice(lRecptLineId As Long, lInvoiceId As Long, dQty As Double, bCanMerge As Long) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lRecptLineId：受領明細の識別子。**
- **lInvoiceId：影響を受ける請求書の識別子。**
- **dQty：請求する数量。受領した数量の内、請求する数量を指定できます（製品の単位で）。**
- **bCanMerge：このパラメータでは、請求書の既存明細に明細を結合するかどうかを指定できます。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmAddReqLinesToEstim()

既存の見積に1**件の依頼の全依頼明細を追加します。**

APIシンタックス

long AmAddReqLinesToEstim(long hApiCnxBase, long lReqId, long lEstimId, long bMergeLines);

内部Basicシンタックス

Function AmAddReqLinesToEstim(lReqId As Long, lEstimId As Long, bMergeLines As Long) As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **lReqId：見積に追加する依頼の識別子。**
- **lEstimId：依頼の全依頼明細を追加する見積の識別子。**

- **bMergeLines：同じ依頼明細（bMergeLines=1）を1つの明細に統合するかどうかを指定できます。統合する明細の数量を合計して、1つの明細を作成します。**

戻りコード

- **0：成功**
- **0以外：エラーコード**

AmAddReqLinesToPO()

既存の発注に1件の依頼のすべての依頼明細を追加します。依頼で指定するサプライヤと発注で指定するサプライヤが同じでなければなりません。

APIシンタックス

long AmAddReqLinesToPO(long hApiCnxBase, long lReqId, long lPOrdId, long bMergeLines);

内部Basicシンタックス

Function AmAddReqLinesToPO(lReqId As Long, lPOrdId As Long, bMergeLines As Long) As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lReqId：発注に追加する依頼の識別子。**
- **lPOrdId：依頼明細を追加する発注の識別子。**

- **bMergeLines：同じ依頼明細（bMergeLines=1）を1つの明細に統合するかどうかを指定できます。統合する明細の数量を合計して、1つの明細を作成します。**

戻りコード

- 0：**成功**
- 0**以外：エラーコード**

AmAddReqLineToEstim()

既存の見積に1つの依頼明細を追加します。

APIシンタックス

long AmAddReqLineToEstim(long hApiCnxBase, long lReqLineId, long lEstimId, long bMergeLines);

内部Basicシンタックス

Function AmAddReqLineToEstim(lReqLineId As Long, lEstimId As Long, bMergeLines As Long) As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **lReqLineId：見積に追加する依頼明細の識別子。**
- **lEstimId：依頼明細を追加する見積の識別子。**

- **bMergeLines：同じ依頼明細（bMergeLines=1）を1つの明細に統合するかどうかを指定できます。統合する明細の数量を合計して、1つの明細を作成します。**

戻りコード

- **0：成功**
- **0以外：エラーコード**

AmAddReqLineToPO()

既存の発注に1つの依頼明細を追加します。

APIシンタックス

long AmAddReqLineToPO(long hApiCnxBase, long lReqLineId, long lPOrdId, long bMergeLines);

内部Basicシンタックス

Function AmAddReqLineToPO(lReqLineId As Long, lPOrdId As Long, bMergeLines As Long) As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lReqLineId：発注に追加する依頼明細の識別子。**
- **lPOrdId：依頼明細を追加する発注の識別子。**

- **bMergeLines：同じ依頼明細（bMergeLines=1）を1つの明細に統合するかどうかを指定できます。統合する明細の数量を合計して、1つの明細を作成します。**

戻りコード

- **0：成功**
- **0以外：エラーコード**

AmAddRequestLineToPOrder()

発注に1つの依頼明細を追加します。

APIシンタックス

long AmAddRequestLineToPOrder(long hApiCnxBase, long lRequestLineId, long lPOrderId, double dQty, long bCanMerge);

内部Basicシンタックス

Function AmAddRequestLineToPOrder(lRequestLineId As Long, lPOrderId As Long, dQty As Double, bCanMerge As Long) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lRequestLineId：依頼明細の識別子。**
- **lPOrderId：影響を受ける発注の識別子。**

- **dQty：発注する数量。依頼された数量の内、発注する数量を指定できます（モデルの単位で）。**
- **bCanMerge：このパラメータでは、発注の既存明細に明細を結合するかどうかを指定できます。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献** 300**]関数（必要に応じて**AmLastErrorMsg() **[献**301**]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmAddTemplateToPOrder()

ある発注に標準発注の内容を全部追加します。

APIシンタックス

```
long AmAddTemplateToPOrder(long hApiCnxBase, long lRequestId, long lPOrderId,
long lTemplateId, long lQty, long bCanMerge);
```

内部Basicシンタックス

```
Function AmAddTemplateToPOrder(lRequestId As Long, lPOrderId As Long,
lTemplateId As Long, lQty As Long, bCanMerge As Long) As Long
```

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lRequestId：追加される発注明細の元となる依頼明細の識別子。**
- **lPOrderId：影響を受ける発注の識別子。**
- **lTemplateId：追加する標準発注の識別子。**
- **lQty：追加する数量（製品の単位で）**
- **bCanMerge：このパラメータでは、発注の既存明細に明細を結合するかどうかを指定できます。**

戻りコード

- 0：**成功**
- 0**以外：エラーコード**

AmAddTemplateToRequest()

ある依頼に標準依頼の内容を全部追加します。

APIシンタックス

long AmAddTemplateToRequest(long hApiCnxBase, long lReqId, long lTemplateId, long lQty, long bCanMerge);

内部Basicシンタックス

Function AmAddTemplateToRequest(lReqId As Long, lTemplateId As Long, lQty As Long, bCanMerge As Long) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓

使用
ウィザードスクリプト
ウィザードの「FINISH.DO」スクリプト 

パラメータ

- lReqId：影響を受ける依頼明細の識別子。
- lTemplateId：追加する標準依頼の識別子。
- lQty：追加する数量（製品の単位で）
- bCanMerge：このパラメータでは、依頼の既存明細に明細を結合するかどうかを指定できます。

戻りコード

- 0：成功
- 0以外：エラーコード

AmArchiveRecord()

データベース中のレコードをアーカイブします。レコードは元のテーブルから削除され、対応するアーカイブテーブルに移動されます。

APIシンタックス

long AmArchiveRecord(long hApiRecord);

内部Basicシンタックス

Function AmArchiveRecord(hApiRecord As Long) As Long

用途

バージョン：4.3.0

使用
AssetCenter API 
リンクまたはフィールドの設定スクリプト
「スクリプト」タイプのアクションの設定 
ウィザードスクリプト

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiRecord：操作に関連するレコードのハンドル。**

戻りコード

- 0：成功
- 0以外：エラーコード

注

注意:

リンクされたレコードの処理は、リンクのタイプによって異なります。「所有」タイプのリンクの場合、リンクされたレコードは同一の方法で処理されます。「定義」または「普通」タイプのリンクの場合、リンクされたレコードの外部キーが0にリセットされ、アーカイブされるレコードの識別子と説明文字列がアーカイブフィールドに入力されます。

重要項目:

この関数は標準テーブルからのレコードに対して使用できます。

AmAttribCmdAvailability()

ヘルプデスクチケットに対して「割当」または「割当解除」ボタンが使用できるかどうかを識別します。

内部Basicシンタックス

Function AmAttribCmdAvailability(bAttrib As Long, lGroupID As Long, lInChargeID As Long, bInChargeIsReadOnly As Long) As Long

用途

バージョン：4.3.0

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **bAttrib：「割当」ボタンが使用可能かどうかをテストするには、このパラメータを「1」に設定します。「割当解除」ボタンが使用可能かどうかをテストするには、このパラメータを「0」に設定します。**
- **lGroupID：関連するヘルプデスクグループに関連付けられたヘルプデスクグループの識別子。**
- **lInChargeID：チケット担当者の識別子。**
- **bInChargeIsReadOnly：このパラメータを「1」に設定すると担当者はチケットの参照だけが可能であり、「0」に設定すると変更の権限があります。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmBackupRecord()

レコードのバックアップコピーを作成します。レコードは対応するアーカイブテーブルにコピーされ、元のテーブルからは削除されません。

APIシンタックス

```
long AmBackupRecord(long hApiRecord);
```

内部Basicシンタックス

```
Function AmBackupRecord(hApiRecord As Long) As Long
```

用途

バージョン：4.3.0

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiRecord：操作に関連するレコードのハンドル。**

戻りコード

- 0：**成功**
- 0**以外：エラーコード**

注

注意:

リンクされたレコードの処理は、リンクのタイプによって異なります。「所有」タイプのリンクの場合、リンクされたレコードは同一の方法で処理されます。「定義」または「普通」タイプのリンクの場合、リンクされたレコードの外部キーが0にリセットされ、アーカイブされるレコードの識別子と説明文字列がアーカイブフィールドに入力されます。

重要項目:

この関数は標準テーブルからのレコードに対して使用できます。

AmBuildNumber()

アプリケーションのビルド番号を返します。

内部Basicシンタックス

Function AmBuildNumber() As Long

用途

バージョン：4.3.0

使用	
AssetCenter API	
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	
ウィザードスクリプト	
ウィザードの「FINISH.DO**」スクリプト**	

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献** 300**]関数（必要に応じて**AmLastErrorMsg() **[献**301**]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmBusinessSecondsInDay()

カレンダに従って、1日の業務時間を秒単位で計算します。

APIシンタックス

long AmBusinessSecondsInDay(long hApiCnxBase, char *strCalendarSqlName, long tmDate);

内部Basicシンタックス

Function AmBusinessSecondsInDay(strCalendarSqlName As String, tmDate As Date) As Date

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strCalendarSqlName：計算に使用するカレンダのSQL名**
- **tmDate：計算を実行する日付**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmCalcConsolidatedFeature()

SQL名でテーブルを識別し、そのテーブルに含まれる合計された任意管理項目の値を計算します。

APIシンタックス

```
long AmCalcConsolidatedFeature(long hApiCnxBase, long lCalcFeatId, char *strSQLTableName);
```

内部Basicシンタックス

```
Function AmCalcConsolidatedFeature(lCalcFeatId As Long, strSQLTableName As String) As Long
```

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lCalcFeatId： 合計された任意管理項目の識別子。**
- **strSQLTableName：合計された任意管理項目を計算するテーブルのSQL名。任意管理項目は、このテーブルの任意管理項目でなければなりません。**

戻りコード

- 0：**成功**
- 0**以外：エラーコード**

AmCalcDepr()

指定した日付の資産の減価償却の金額を計算できます。この日付の減価償却の値を返します。

APIシンタックス

double AmCalcDepr(long hApiCnxBase, long iType, long lDuration, double dCoeff, double dPrice, long tmStart, long tmDate);

内部Basicシンタックス

Function AmCalcDepr(iType As Long, lDuration As Long, dCoeff As Double, dPrice As Double, tmStart As Date, tmDate As Date) As Double

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **iType：減価償却の属性を指定します。次のいずれかの値になります。**
  - 0：減価償却なし
  - 1：定額法
  - 2：定率法
- **lDuration：資産が減価償却される期間。この期間は秒単位で示されます。**
- **dCoeff：定率法の減価償却に適用する償却率。定額法ではこのパラメータは使用されませんが、値は設定しておく必要があります。**
- **dPrice：減価償却を計算する資産の初期価額**
- **tmStart：資産の減価償却を開始する日付**
- **tmDate：資産の減価償却と残存価額を計算する日付**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmCalculateCatRefQty()

購入依頼を満たすために発注するカタログリファレンスの数量を計算します。

APIシンタックス

double AmCalculateCatRefQty(long hApiCnxBase, long lSetQty, long lUseUnitId, long lPurchUnitId, char *strModelDesc, char *strCatRefDesc, char *strPurchUnit, char *strUseUnit, double dPkgQty, double dUnitConv, double dReqLineQty);

内部Basicシンタックス

Function AmCalculateCatRefQty(lSetQty As Long, lUseUnitId As Long, lPurchUnitId As Long, strModelDesc As String, strCatRefDesc As String, strPurchUnit As String, strUseUnit As String, dPkgQty As Double, dUnitConv As Double, dReqLineQty As Double) As Double

用途

バージョン：?

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lSetQty：製品の各品目の対応する数量（例えば1リットル・ボトルの水6本を含む製品の場合は1）。**
- **lUseUnitId：モデルに用いられる単位の識別子。**
- **lPurchUnitId：製品に用いられる単位の識別子。**
- **strModelDesc：モデルの説明。**
- **strCatRefDesc：カタログリファレンスの説明。**
- **strPurchUnit：製品に用いられる単位の説明。**
- **strUseUnit：モデルに用いられる単位の説明。**
- **dPkgQty：製品の各品目の対応する数量（例えば1リットル・ボトルの水6本を含む製品の場合は1）。**
- **dUnitConv：製品の単位の変換率。**
- **dReqLineQty：購入依頼で規定されたモデルの数量。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献 301]関数も）を呼び出し、エラーが発生し**

たかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

AmCalculateReqLineQty()

購入依頼の作成に必要なモデルの数量を計算します。

APIシンタックス

double AmCalculateReqLineQty(long hApiCnxBase, long lSetQty, long lUseUnitId, long lPurchUnitId, char *strModelDesc, char *strCatRefDesc, char *strPurchUnit, char *strUseUnit, double dPkgQty, double dUnitConv, double dCatRefQty);

内部Basicシンタックス

Function AmCalculateReqLineQty(lSetQty As Long, lUseUnitId As Long, lPurchUnitId As Long, strModelDesc As String, strCatRefDesc As String, strPurchUnit As String, strUseUnit As String, dPkgQty As Double, dUnitConv As Double, dCatRefQty As Double) As Double

用途

バージョン：?

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lSetQty**：製品の各品目の対応する数量（例えば1リットル・ボトルの水6本を含む製品の場合は1）。
- **lUseUnitId**：モデルに用いられる単位の識別子。
- **lPurchUnitId**：製品に用いられる単位の識別子。
- **strModelDesc**：モデルの説明。
- **strCatRefDesc**：カタログリファレンスの説明。

- **strPurchUnit：製品に用いられる単位の説明。**
- **strUseUnit：モデルに用いられる単位の説明。**
- **dPkgQty：製品の各品目の対応する数量（例えば1リットル・ボトルの水6本を含む製品の場合は1）。**
- **dUnitConv：製品の単位の変換率。**
- **dCatRefQty：発注されたカタログリファレンスで規定されたモデルの数量。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmCbkReplayEvent()

この関数はイベントの起点にあるレコードを訂正した後、イベントの起点で経費付替え規則を再適用します。

APIシンタックス

long AmCbkReplayEvent(long hApiCnxBase, long lCbkEventId);

内部Basicシンタックス

Function AmCbkReplayEvent(lCbkEventId As Long) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lCbkEventId：関連する経費付替えイベントの識別子。**

戻りコード

- 0：**成功**
- 0**以外：エラーコード**

AmCheckTraceDone()

AmCheckTraceDone API**は、ポート（**lPortId**）またはバンドル（**lBundleId**）が既存のトレースに関連付けられるかを決めます。トレースの方向（**iTraceDir**）はトレースが、ユーザからホストへの方向（**iTraceDir = 1**）で検証されるべきか、またはホストからユーザへの方向（**iTraceDir = 0**）で検証されるべきかを指定します。**

APIシンタックス

long AmCheckTraceDone(long hApiCnxBase, long lPortId, long lBundleId, long iTraceDir);

内部Basicシンタックス

Function AmCheckTraceDone(lPortId As Long, lBundleId As Long, iTraceDir As Long) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

使用	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lPortId：確認するポートのID**
- **lBundleId ：確認するバンドルのID**
- **iTraceDir：このパラメータは検証する方向を指定します。**
  - 1：ホストの方向で検証する
  - 0：ユーザの方向で検証する

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

AmCleanup()

この関数は、データベースの変更関数を使うスクリプトの最後に呼び出されます。使用したすべてのリソースを解放します。

APIシンタックス

```
void AmCleanup();
```

用途

バージョン：2.52

使用	
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	
ウィザードスクリプト	

使用
ウィザードの「FINISH.DO」スクリプト

AmClearLastError()

最後に関数を呼び出した際に発生した最後のエラーメッセージに関する情報を消去します。

APIシンタックス

```
long AmClearLastError(long hApiCnxBase);
```

内部Basicシンタックス

```
Function AmClearLastError() As Long
```

用途

バージョン：2.52

使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

戻りコード

- 0：成功
- 0以外：エラーコード

AmCloseAllChildren()

現在の接続で作成されたすべてのオブジェクトを破棄します。

APIシンタックス

long AmCloseAllChildren(long hApiCnxBase);

内部Basicシンタックス

Function AmCloseAllChildren() As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

戻りコード

- 0：**成功**
- 0**以外：エラーコード**

AmCloseConnection()

指定した接続のAssetCenter**セッションを終了します。接続中に作成したすべてのオブジェクト（クエリ、レコード、テーブル、フィールドなど）は自動的に破棄されます。ハンドルは無効になります。接続ハンドルは存在しません。**

APIシンタックス

long AmCloseConnection(long hApiCnxBase);

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	

戻りコード

- 0：成功
- 0以外：エラーコード

AmCommit()

接続に関連付けられているデータベースに加えられたすべての変更を有効にします。

APIシンタックス

long AmCommit(long hApiCnxBase);

内部Basicシンタックス

Function AmCommit() As Long

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

戻りコード

- 0：成功
- 0以外：エラーコード

AmComputeAllLicAndInstallCounts()

すべてのレコードについて、ソフトウェアライセンス数とインストール数をカウントします。

APIシンタックス

long AmComputeAllLicAndInstallCounts(long hApiCnxBase);

内部Basicシンタックス

Function AmComputeAllLicAndInstallCounts() As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO**」スクリプト**	✓

戻りコード

- 0：**成功**
- 0**以外：エラーコード**

AmComputeLicAndInstallCounts()

1つのレコードについて、ソフトウェアライセンス数とインストール数をカウントします。

APIシンタックス

long AmComputeLicAndInstallCounts(long hApiCnxBase, long lSLCountId);

内部Basicシンタックス

Function AmComputeLicAndInstallCounts(lSLCountId As Long) As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lSLCountId：ソフトウェアライセンスカウンタの識別子。**

戻りコード

- 0：**成功**
- 0**以外：エラーコード**

AmConnectionName()

現在のデータベース接続名を返します。

APIシンタックス

long AmConnectionName(long hApiCnxBase, char *return, long lreturn);

内部Basicシンタックス

Function AmConnectionName() As String

用途

バージョン：4.3.0

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献 301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

例

```
RetVal=amConnectionName()
```

AmConnectTrace()

AmConnectTrace APIはソースデバイス／ケーブルをターゲットデバイス／ケーブルへ接続し、トレース履歴とトレースの処理を作成します。

APIシンタックス

long AmConnectTrace(long hApiCnxBase, long iSrcLinkType, long lSrcPortBunId, long lSrcLabelRuleId, long iDestLinkType, long lDestPortBunId, long lDestLabelRuleId, long iTraceDir, long lDutyId, char *strComment, long lCabTraceOutId);

内部Basicシンタックス

Function AmConnectTrace(iSrcLinkType As Long, lSrcPortBunId As Long, lSrcLabelRuleId As Long, iDestLinkType As Long, lDestPortBunId As Long, lDestLabelRuleId As Long, iTraceDir As Long, lDutyId As Long, strComment As String, lCabTraceOutId As Long) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **iSrcLinkType：このパラメータはソースデバイス／ケーブル用のトレースのタイプを決めます。**
  - 8：ケーブル
  - 9：デバイス
- **lSrcPortBunId ：ソース側で接続するポートまたはバンドルのID**
- **lSrcLabelRuleId ：ソースケーブルリンク用のラベル付け規則のID**
- **iDestLinkType：このパラメータはターゲットデバイス／ケーブル用のトレースのタイプを決めます。**
  - 8：ケーブル
  - 9：デバイス
- **lDestPortBunId ：ターゲット側で接続するポートまたはバンドルのID**
- **lDestLabelRuleId ：ターゲットケーブルリンク用のラベル付け規則のID**
- **iTraceDir：このパラメータは接続の方向を指定します。**
  - 1：ユーザからホストへ
  - 0：ホストからユーザへ
- **lDutyId ：ケーブルタイプのリンクの用途ID**
- **strComment：トレースの処理のラベル**
- **lCabTraceOutId：ケーブルのトレースの説明のID**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

AmConvertCurrency()

指定した日付の金額を、別の通貨に換算します。

APIシンタックス

double AmConvertCurrency(long hApiCnxBase, long tmDate, char *strSrcName, char *strDstName, double dVal);

内部Basicシンタックス

Function AmConvertCurrency(tmDate As Date, strSrcName As String, strDstName As String, dVal As Double) As Double

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **tmDate：換算日。この日付現在の有効な為替レートを取得するために使います。**
- **strSrcName：換算する金額の元の通貨**
- **strDstName：換算通貨。金額の換算に使う通貨です。**
- **dVal：換算する金額（元の通貨による金額）**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献301]関数も）を呼び出し、エラーが発生し**

たかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

注

注意:

この関数の通貨パラメータ（strSrcNameとstrDstName）は、必ずAssetCenterで定義しておく必要があります。また、換算日（tmDateパラメータ）現在の有効な為替レートがなければなりません。

例

1998年11月2日に、5,000フランをドルに換算します。

```
AmConvertCurrency("1998/11/02 00:00:00", "FRF", "$", 5000)
```

AmConvertDateBasicToUnix()

BASIC形式の日付（Date型）をUNIX形式の日付（Long型）に変換します。外部ツールからこの関数を使用しても、2つのタイプは同等なため機能しません。

APIシンタックス

long AmConvertDateBasicToUnix(long hApiCnxBase, long tmTime);

内部Basicシンタックス

Function AmConvertDateBasicToUnix(tmTime As Date) As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **tmTime：変換する日付**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmConvertDateIntlToUnix()

国際標準形式の日付（Date型）をUNIX形式の日付（Long型）に変換します。

APIシンタックス

long AmConvertDateIntlToUnix(long hApiCnxBase, char *strDate);

内部Basicシンタックス

Function AmConvertDateIntlToUnix(strDate As String) As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strDate：国際標準形式（yyyy-mm-dd hh:mm:ss）変換する日付**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

AmConvertDateStringToUnix()

文字列の日付（Windowsのコントロールパネルに表示される形式）をUNIXのLong型に変換します。

APIシンタックス

long AmConvertDateStringToUnix(long hApiCnxBase, char *strDate);

内部Basicシンタックス

Function AmConvertDateStringToUnix(strDate As String) As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strDate：変換する文字列の日付**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmConvertDateUnixToBasic()

UNIX形式の日付（Long型）をBASIC形式の日付（Date型）に変換します。外部ツールからこの関数を使用しても、2つのタイプは同等なため機能しません。

APIシンタックス

```
long AmConvertDateUnixToBasic(long hApiCnxBase, long lTime);
```

内部Basicシンタックス

```
Function AmConvertDateUnixToBasic(lTime As Long) As Date
```

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lTime：変換する日付**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmConvertDateUnixToIntl()

UNIX形式の日付（Long型）を国際標準形式の日付（yyyy-mm-dd hh:mm:ss）に変換します。

APIシンタックス

```
long AmConvertDateUnixToIntl(long hApiCnxBase, long lUnixDate, char *pstrDate, long lDate);
```

内部Basicシンタックス

```
Function AmConvertDateUnixToIntl(lUnixDate As Long) As String
```

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓

	使用
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lUnixDate：変換する日付**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献 301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmConvertDateUnixToString()

Long型のUNIX形式の日付を文字列形式の日付（Windowsのコントロールパネルに表示される形式）に変換します。

APIシンタックス

```
long AmConvertDateUnixToString(long hApiCnxBase, long lUnixDate, char
*pstrDate, long lDate);
```

内部Basicシンタックス

```
Function AmConvertDateUnixToString(lUnixDate As Long) As String
```

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓

	使用
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lUnixDate：変換するLong型のUNIX形式の日付**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmConvertDoubleToString()

倍精度の数字を文字列に変換します。文字列は、Windows**のコントロールパネルで設定した地域のオプション（数字）に応じてフォーマットされます。**

APIシンタックス

```
long AmConvertDoubleToString(double dSrc, char *pstrDst, long lDst);
```

内部Basicシンタックス

```
Function AmConvertDoubleToString(dSrc As Double) As String
```

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓

	使用
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **dSrc：変換する倍精度の数字**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

AmConvertMonetaryToString()

金額値を文字列に変換します。文字列は、Windowsのコントロールパネルで設定した地域のオプション（通貨）に応じてフォーマットされます。

APIシンタックス

long AmConvertMonetaryToString(double dSrc, char *pstrDst, long lDst);

内部Basicシンタックス

Function AmConvertMonetaryToString(dSrc As Double) As String

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓

	使用
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **dSrc：変換する金額値**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

AmConvertStringToDouble()

文字列（Windowsのコントロールパネルで設定したフォーマット）を倍精度の数字に変換します。

APIシンタックス

double AmConvertStringToDouble(char *strSrc);

内部Basicシンタックス

Function AmConvertStringToDouble(strSrc As String) As Double

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓

	使用
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strSrc：変換する文字列**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

AmConvertStringToMonetary()

文字列（Windowsのコントロールパネルで設定したフォーマット）を金額値に変換します。

APIシンタックス

double AmConvertStringToMonetary(char *strSrc);

内部Basicシンタックス

Function AmConvertStringToMonetary(strSrc As String) As Double

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓

	使用
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strSrc：変換する文字列**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmCounter()

strCounterNameカウンタの値よりも1つ大きい値を返します。iWidthがカウンタの桁数よりも大きい場合は、ゼロが先頭に追加されます。iWidthの値よりもカウンタの桁数の方が大きい場合は、関数が返す結果は決して切り捨てられません。

内部Basicシンタックス

Function AmCounter(strCounterName As String, iWidth As Long) As String

用途

バージョン：2.52

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	
ウィザードスクリプト	

使用
ウィザードの「FINISH.DO」スクリプト

パラメータ

- **strCounterName**：AssetCenterで定義されているカウンタの名前（［管理／カウンタ］メニューからアクセスできます。）
- **iWidth**：この関数の出力を何桁で表示するかを指定します。このパラメータは、カウンタのサイズがパラメータ値よりも小さい場合のみ使用されます。

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献 301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

注

注意:

この関数をスクリプトタイプのアクションで使用する場合、アクションのコンテキストを指定する必要があります。指定しないとエラーが発生します。

例

「納入」カウンタの値を5桁で返します。

```
Dim strCounterName As String
strCounter = AmCounter("納入", 5)
```

例えば、「納入」カウンタが18の場合は次の値を返します。

```
00019
```

AmCreateAssetPort()

AmCreateAssetPort APIはデバイス（lAssetId）用に新規のポートを作成します。新規のポートには、あるケーブルコネクタタイプ（lCabCnxTypeId）のピン（iPinCount）が付属しています。ピンのステータスは「使用可能」でなければなりません。ポートに追加されるピンは連続番号で並べ替えられます。ポートの方向（bPinPortDir）に従って、使用可能なピンは昇順（bPinPortDir=0）または降順（bPinPortDir=1）で並べ替えられます。この関数は新規のポートに用途（lDutyId）を割り当てます。

APIシンタックス

long AmCreateAssetPort(long hApiCnxBase, long lAssetId, long lCabCnxTypeId, long lDutyId, long iPinCount, long bPinPortDir, long iConnStatus, long bConsecutivePins, long iPrevPinSeq, long bLogError);

内部Basicシンタックス

Function AmCreateAssetPort(lAssetId As Long, lCabCnxTypeId As Long, lDutyId As Long, iPinCount As Long, bPinPortDir As Long, iConnStatus As Long, bConsecutivePins As Long, iPrevPinSeq As Long, bLogError As Long) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lAssetId：デバイスの**ID
- **lCabCnxTypeId：ケーブルの接続タイプの**ID
- **lDutyId：ポートの用途タイプの**ID
- **iPinCount：新規のポートで使用されるピンの数**
- **bPinPortDir：このパラメータはポートの方向を指定します。**

- **iConnStatus**
- **bConsecutivePins**
- **iPrevPinSeq**
- **bLogError**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmCreateAssetsAwaitingDelivery()

この関数は納品待ちの資産を作成します。

APIシンタックス

long AmCreateAssetsAwaitingDelivery(long hApiCnxBase, long lPOrdId);

内部Basicシンタックス

Function AmCreateAssetsAwaitingDelivery(lPOrdId As Long) As Long

用途

バージョン：3.61

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lPOrdId：関連する発注の識別子。**

戻りコード

- 0：**成功**
- 0**以外：エラーコード**

AmCreateCable()

AmCreateCable API**は新規ケーブルを作成します。ケーブルはモデル（**lModelId**）、ケーブルの役割（**strCableRole**）、ラベル付け規則（**lLabelRuleId**）、ユーザの場所（**lUserId**）とホストの場所（**lHostId**）に基づいて作成されます。プロジェクト（**lProjectId**）と作業指示（**lWorkOrderId**）に値が入力されると、新規のケーブルは、コメント（**strComment**）と共にプロジェクトや作業指示に追加されます。このコメントはケーブル上に実行されるアクションを説明します（例：新規ケーブルの設置）。**

APIシンタックス

long AmCreateCable(long hApiCnxBase, long lModelId, long lUserId, long lHostId, char *strCableRole, long lProjectId, long lWorkOrderId, char *strComment, long lLabelRuleId, char *strLabel);

内部Basicシンタックス

Function AmCreateCable(lModelId As Long, lUserId As Long, lHostId As Long, strCableRole As String, lProjectId As Long, lWorkOrderId As Long, strComment As String, lLabelRuleId As Long, strLabel As String) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lModelId：ケーブルのモデルの**ID
- **lUserId：ユーザ側の場所の**ID
- **lHostId：ホスト側の場所の**ID
- **strCableRole：ケーブルの役割**
- **lProjectId：このパラメータはケーブルの設置に関連するプロジェクトを定義します。**
- **lWorkOrderId：ケーブル設置に関連する作業指示の**ID
- **strComment：作業指示（**lWorkOrderId**で指定）に添付されるコメント**
- **lLabelRuleId：ケーブル用のラベルの作成時に適用されるラベル付け規則の**ID
- **strLabel：このパラメータはケーブルに貼るラベルを指定します。**

戻りコード

エラーの場合は、次の2**つの処理方法を用意します。**

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmCreateCableBundle()

AmCreateCableBundle API**はケーブル（**lCableId**）に新規バンドルを作成します。新規バンドルには、あるケーブルペアタイプ（**lPairType**）のケーブルペア（**iPairCount**）が付属しています。ペアのステータスは「使用可能」でなければなりません。この関数は用途（**lDutyId**）を新規バンドルに割り当てます。**

APIシンタックス

long AmCreateCableBundle(long hApiCnxBase, long lCableId, long lPairTypeId, long lDutyId, long iPairCount, long iStartPairSeq, long bLogError);

内部Basicシンタックス

Function AmCreateCableBundle(lCableId As Long, lPairTypeId As Long, lDutyId As Long, iPairCount As Long, iStartPairSeq As Long, bLogError As Long) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **lCableId：ケーブルの**ID**（このID**はケーブルのテーブルに存在しなければなりません**）。**
- **lPairTypeId：ケーブルペアタイプの**ID
- **lDutyId ：バンドルの用途の**ID
- **iPairCount：このパラメータはバンドルのペア数を定義します。**
- **iStartPairSeq**
- **bLogError**

戻りコード

エラーの場合は、次の2**つの処理方法を用意します。**

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmCreateCableLink()

AmCreateCableLink APIはケーブル（lCableId）とバンドル（lBundleId）用に新規のケーブルリンクを作成します。ケーブルリンクの用途はある機能（lDutyId）から設定されます。ケーブルリンクのラベル付け規則は、あるラベル付け規則（lLabelRuleId）に基づいて設定されます。

注意:

ラベル付けがあるラベル付け規則により更新されることはありません。これにはAmRefreshLabel()関数を別に呼び出す必要があります。

以前のケーブルリンク（lPrevLinkId）が指定されている場合、親ケーブルリンクが2レコード間に作成されます。以前のケーブルリンクは、これら2つのレコードの従属ケーブルリンクになります。

APIシンタックス

long AmCreateCableLink(long hApiCnxBase, long lCableId, long lDutyId, long lBundleId, long lPrevLinkId, long iTraceDir, long lLabelRuleId);

内部Basicシンタックス

Function AmCreateCableLink(lCableId As Long, lDutyId As Long, lBundleId As Long, lPrevLinkId As Long, iTraceDir As Long, lLabelRuleId As Long) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lCableId：**接続用ケーブルID
- **lDutyId：**接続の機能のID

- **lBundleId：接続するケーブルバンドルのID**
- **lPrevLinkId：接続に使用されるケーブルリンクのID。値「0」を使用するとこのパラメータは必須ではなくなります。**
- **iTraceDir： 接続の方向**
  - **0=ホストからユーザへの方向**
  - **1=ユーザからホストへの方向**
- **lLabelRuleId：使用されるラベル付け規則のID**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmCreateDelivFromPO()

発注した製品を受領し、それによって作成された受領伝票の識別子を返します。

APIシンタックス

long AmCreateDelivFromPO(long hApiCnxBase, long lPOrdId);

内部Basicシンタックス

Function AmCreateDelivFromPO(lPOrdId As Long) As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lPOrdId：受領する発注の識別子。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmCreateDevice()

AmCreateDevice APIは新規デバイスを作成します。デバイスはモデル（lModelId）と場所（lLocationId）を基に作成されます。ラベル付け規則は、ある規則（lLabelRuleId）に応じて設定されます。

注意:

ラベル付け規則はラベルを更新しません。更新にはAmRefreshLabel()を別に呼び出す必要があります。

プロジェクト（lProjectId）と作業指示（lWorkOrderId）に値が入力されると、新規資産はstrComment内のコメントと共にプロジェクトと作業指示に追加されます。コメントは資産に実行されるアクションを説明します（例：新規資産の設置）。

APIシンタックス

long AmCreateDevice(long hApiCnxBase, long lModelId, long lLocationId, long lProjectId, long lWorkOrderId, long lLabelRuleId, char *strComment);

内部Basicシンタックス

Function AmCreateDevice(lModelId As Long, lLocationId As Long, lProjectId As Long, lWorkOrderId As Long, lLabelRuleId As Long, strComment As String) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **lModelId：新規デバイスのモデルの**ID
- **lLocationId：新規デバイスの場所の**ID
- **lProjectId：プロジェクトの**ID。「0」の値をとることも可能です。
- **lWorkOrderId：作業指示の**ID。「0」の値をとることも可能です。
- **lLabelRuleId：資産に使用されるラベル付け規則の**ID
- **strComment：作業指示に添付されるコメント**

戻りコード

エラーの場合は、次の2**つの処理方法を用意します。**

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmCreateDeviceLink()

AmCreateDeviceLink API**は、あるデバイス（**lAssetId**）とポート（**lPortId**）用にデバイスタイプのケーブルリンクを作成します。ケーブルリンクのラベル付け規則は、あるラベル付け規則（**lLabelRuleId**）を元に設定されます。**

注意:

ラベル付け規則はラベルを更新しません。別にAmRefreshLabel()を呼び出す必要があります。

以前のケーブルリンク（lPrevLinkId）が指定される場合は、親ケーブルリンクが2つのレコード間に作成されます。トレースの方向が「ユーザからホスト」の場合（iTraceDir＝1）、以前のケーブルリンクは従属になります。トレースの方向が「ホストからユーザ」の場合 （iTraceDir＝0）以前のケーブルリンクは親になります。

APIシンタックス

long AmCreateDeviceLink(long hApiCnxBase, long lAssetId, long lPortId, long lPrevLinkId, long iTraceDir, long lLabelRuleId);

内部Basicシンタックス

Function AmCreateDeviceLink(lAssetId As Long, lPortId As Long, lPrevLinkId As Long, iTraceDir As Long, lLabelRuleId As Long) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lAssetId：接続される資産の識別子。**
- **lPortId：接続予定のポートの識別子。**
- **lPrevLinkId：接続を可能にするデバイスのケーブルリンクの識別子。**
- **iTraceDir： 接続の方向**
  - 0=**ホストからユーザへの方向**
  - 1=**ユーザからホストへの方向**
- **lLabelRuleId：新規接続に使用されるラベル付け規則の識別子。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmCreateEstimFromReq()

購入依頼から見積を作成して、その見積の識別子を返します。

APIシンタックス

long AmCreateEstimFromReq(long hApiCnxBase, long lReqId, long lSuppId);

内部Basicシンタックス

Function AmCreateEstimFromReq(lReqId As Long, lSuppId As Long) As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **lReqId：見積の作成に使用する購入依頼の識別子。**
- **lSuppId：この関数によって作成される見積のサプライヤ識別子。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

AmCreateEstimsFromAllReqLines()

依頼から見積を作成して、その見積の識別子を返します。

APIシンタックス

long AmCreateEstimsFromAllReqLines(long hApiCnxBase, long lReqId, long bMergeLines, long lDefSuppId);

内部Basicシンタックス

Function AmCreateEstimsFromAllReqLines(lReqId As Long, bMergeLines As Long, lDefSuppId As Long) As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lReqId：見積の作成元となる依頼の識別子。**

- **bMergeLines：同じ依頼明細（bMergeLines=1）を1つの明細に統合するかどうかを指定できます。統合する明細の数量を合計して、1つの明細を作成します。**
- **lDefSupplId：見積のデフォルトサプライヤの識別子。**

戻りコード

- 0：**成功**
- 0**以外：エラーコード**

AmCreateInvFromPO()

発注から請求書を作成して、その請求書の識別子を返します。

APIシンタックス

long AmCreateInvFromPO(long hApiCnxBase, long lPOrdId);

内部Basicシンタックス

Function AmCreateInvFromPO(lPOrdId As Long) As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **lPOrdId：請求書の作成元となる発注の識別子。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

AmCreateLink()

レコードのリンクを変更して、ターゲットテーブルの新規レコード（hApiRecDest）を指すようにします。つまり2レコード間のリンクを作成します。

APIシンタックス

long AmCreateLink(long hApiRecord, char *strLinkName, long hApiRecDest);

内部Basicシンタックス

Function AmCreateLink(hApiRecord As Long, strLinkName As String, hApiRecDest As Long) As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiRecord**：変更するリンクを含むレコードのハンドル
- **strLinkName**：変更するリンクのSQL名

- **hApiRecDest：リンクのターゲットレコードのハンドル**

戻りコード

- 0：成功
- 0以外：エラーコード

AmCreateOrUpdateInvoiceFromReceipt()

受領伝票から請求書を作成または更新します。

APIシンタックス

long AmCreateOrUpdateInvoiceFromReceipt(long hApiCnxBase, long lRecptId);

内部Basicシンタックス

Function AmCreateOrUpdateInvoiceFromReceipt(lRecptId As Long) As Long

用途

バージョン：?

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lRecptId：操作に関連する請求書の識別子。**

戻りコード

作成された請求書の識別子を返します。

注

注意:

この関数を外部ツールから呼び出して更新を行うことはできません。

AmCreatePOFromEstim()

見積から発注を作成して、その発注の識別子を返します。

APIシンタックス

long AmCreatePOFromEstim(long hApiCnxBase, long lEstimId);

内部Basicシンタックス

Function AmCreatePOFromEstim(lEstimId As Long) As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **lEstimId：発注の作成に使用する見積の識別子。**

戻りコード

エラーの場合は、次の2**つの処理方法を用意します。**

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**

- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmCreatePOFromReq()

購入依頼から発注を作成して、その発注の識別子を返します。

APIシンタックス

long AmCreatePOFromReq(long hApiCnxBase, long lReqId, long lSuppId);

内部Basicシンタックス

Function AmCreatePOFromReq(lReqId As Long, lSuppId As Long) As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **lReqId：発注の作成に使用する購入依頼の識別子。**
- **lSuppId：作成する発注のサプライヤの識別子。**

戻りコード

エラーの場合は、次の2**つの処理方法を用意します。**

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**

- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献 301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmCreatePOrderFromRequest()

依頼から発注を作成します。

APIシンタックス

long AmCreatePOrderFromRequest(long hApiCnxBase, long lRequestId, long lSupplierId);

内部Basicシンタックス

Function AmCreatePOrderFromRequest(lRequestId As Long, lSupplierId As Long) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **lRequestId：関連する依頼の識別子。**
- **lSupplierId：発注用のサプライヤの識別子。**

戻りコード

エラーの場合は、次の2**つの処理方法を用意します。**

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

AmCreatePOrdersFromRequest()

依頼に応じるために必要な発注をすべて作成します。

APIシンタックス

long AmCreatePOrdersFromRequest(long hApiCnxBase, long lRequestId);

内部Basicシンタックス

Function AmCreatePOrdersFromRequest(lRequestId As Long) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lRequestId：関連する依頼の識別子。**

戻りコード

- 0：**成功**
- 0**以外：エラーコード**

AmCreatePOsFromAllReqLines()

1件の依頼の各依頼明細から発注を作成します。

APIシンタックス

long AmCreatePOsFromAllReqLines(long hApiCnxBase, long lReqId, long bMergeLines, long lDefSuppId);

内部Basicシンタックス

Function AmCreatePOsFromAllReqLines(lReqId As Long, bMergeLines As Long, lDefSuppId As Long) As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lReqId：発注を作成する依頼の識別子**
- **bMergeLines：同じ依頼明細（bMergeLines=1）を1つの明細に統合するかどうかを指定できます。統合する明細の数量を合計して、1つの明細を作成します。**
- **lDefSuppId：依頼する製品のデフォルトのサプライヤの識別子。このパラメータはオプションで、デフォルトの設定は「0」です。**

戻りコード

- 0：**成功**
- 0**以外：エラーコード**

AmCreateProjectCable()

AmCreateProjectCable APIは、ケーブル（lCableId）をプロジェクト（lProjectId）と作業指示（lWorkOrderId）に追加します。コメント（strComment）は実行されるアクションを説明します（例：新規ケーブルの設置）。

APIシンタックス

long AmCreateProjectCable(long hApiCnxBase, long lProjectId, long lWorkOrderId, long lCableId, char *strComment);

内部Basicシンタックス

Function AmCreateProjectCable(lProjectId As Long, lWorkOrderId As Long, lCableId As Long, strComment As String) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lProjectId：新規ケーブルを取得するプロジェクトの**ID
- **lWorkOrderId：ケーブル用の作業指示の**ID
- **lCableId：ケーブルの**ID
- **strComment：作業指示に添付されるコメント**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**

- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmCreateProjectDevice()

AmCreateProjectDevice API**は、デバイス（**lAssetId**）をプロジェクト（**lProjectId**）と作業指示（**lWorkOrderId**）に追加します。コメント（**strComment**）は実行されるアクションを説明します（例：新規デバイスの設置）。**

APIシンタックス

long AmCreateProjectDevice(long hApiCnxBase, long lProjectId, long lWorkOrderId, long lAssetId, char *strComment);

内部Basicシンタックス

Function AmCreateProjectDevice(lProjectId As Long, lWorkOrderId As Long, lAssetId As Long, strComment As String) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **lProjectId：新規デバイスを取得するプロジェクトの**ID
- **lWorkOrderId：新規デバイスを取得する作業指示の**ID
- **lAssetId：資産としての新規デバイスの**ID
- **strComment：作業指示に添付されるコメント**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

AmCreateProjectTrace()

AmCreateProjectTrace APIはトレース（strTrace）をプロジェクト（lProjectId）と作業指示（lWorkOrderId）に追加します。トレースの用途は、ある用途（lDutyId）を基に設定されます。トレースのタイプ（iTraceType）は、トレースが接続（lTraceType＝1）であるか、切断（lTraceType＝2）であるかを指定します。変更されたユーザリンクのラベル（strModLinkLabel）は、トレースのどの部分が変更されたかを識別します。コメント（strComment）は実行されるアクションを説明します（例：デバイスの接続）。

APIシンタックス

```
long AmCreateProjectTrace(long hApiCnxBase, long lProjectId, long lWorkOrderId,
long iTraceType, long lDutyId, char *strModLinkLabel, char *strTrace, char
*strComment);
```

内部Basicシンタックス

```
Function AmCreateProjectTrace(lProjectId As Long, lWorkOrderId As Long,
iTraceType As Long, lDutyId As Long, strModLinkLabel As String, strTrace As String,
strComment As String) As Long
```

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lProjectId：トレースの情報を取得するプロジェクトのID**
- **lWorkOrderId：トレースの情報を取得する作業指示のID**
- **iTraceType ：トレースのタイプ**
  - 1=**接続**
  - 2=**切断**
- **lDutyId：用途のID。作業指示内に表示されます。**
- **strModLinkLabel：作業指示用に使用されるコメント**
- **strTrace：作業指示用に使用されるトレースの説明の文字列**
- **strComment：作業指示に添付されるコメント**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献 301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmCreateReceiptFromPOrder()

この関数は発注から受領を作成します。

APIシンタックス

long AmCreateReceiptFromPOrder(long hApiCnxBase, long lPOrderId);

内部Basicシンタックス

Function AmCreateReceiptFromPOrder(lPOrderId As Long) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lPOrderId：関連する発注の識別子。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmCreateRecord()

デフォルトの値を取る空のレコードをテーブル内に作成します。このレコードは、データベースに挿入した時に作成されます。

APIシンタックス

long AmCreateRecord(long hApiCnxBase, char *strTable);

内部Basicシンタックス

Function AmCreateRecord(strTable As String) As Long

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strTable：レコード作成先のテーブルの**SQL名

例

次の例ではデータベースに従業員を作成します。

```
Dim lErr As Long
Dim hRecord As Long
hRecord = amCreateRecord("amEmplDept")
lErr = amSetFieldStrValue(hRecord, "Name", "Doe")
lErr = amSetFieldStrValue(hRecord, "FirstName", "John")
lErr = amInsertRecord(hRecord)
```

AmCreateRequestToInvoice()

調達サイクルの全オブジェクト（依頼、発注、受領、請求）を作成します。

APIシンタックス

long AmCreateRequestToInvoice(long hApiCnxBase, double dQty, long lCatRefId, double dUnitPrice, char *strCur, long lRequesterId, long lCostId, long lUserId, long lStockId);

内部Basicシンタックス

Function AmCreateRequestToInvoice(dQty As Double, lCatRefId As Long, dUnitPrice As Double, strCur As String, lRequesterId As Long, lCostId As Long, lUserId As Long, lStockId As Long) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **dQty：発注、受領、請求する数量（パッケージ単位）**
- **lCatRefId：カタログリファレンスの識別子。**
- **dUnitPrice：カタログリファレンスの単価**
- **strCur：カタログリファレンスの価格用の通貨コード**
- **lRequesterId：依頼者の識別子。**
- **lCostId：影響を受けるコストセンタの識別子。**
- **lUserId：発注品のユーザ識別子。**
- **lStockId：受領品の配達先在庫の識別子。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献** 300**]関数（必要に応じて**AmLastErrorMsg() **[献**301**]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

注

次の関数を順番に呼び出すのと同じ操作になります。amCreateRequestToReceipt, amCreateOrUpdateInvoiceFromReceipt

---

AmCreateRequestToPOrder()

調達サイクルのオブジェクト（依頼、発注）を作成します。

APIシンタックス

long AmCreateRequestToPOrder(long hApiCnxBase, double dQty, long lCatRefId, double dUnitPrice, char *strCur, long lRequesterId, long lCostId, long lUserId, long lStockId);

内部Basicシンタックス

Function AmCreateRequestToPOrder(dQty As Double, lCatRefId As Long, dUnitPrice As Double, strCur As String, lRequesterId As Long, lCostId As Long, lUserId As Long, lStockId As Long) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **dQty：発注する数量（パッケージ単位）**
- **lCatRefId：カタログリファレンスの識別子。**
- **dUnitPrice：カタログリファレンスの単価**
- **strCur：カタログリファレンスの価格用の通貨コード**
- **lRequesterId：依頼者の識別子。**
- **lCostId：影響を受けるコストセンタの識別子。**
- **lUserId：発注品のユーザ識別子。**
- **lStockId：受領品の配達先在庫の識別子。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献** 300**]関数（必要に応じて**AmLastErrorMsg() **[献** 301**]関数も）を呼び出し、エラーが発生し**

たかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

AmCreateRequestToReceipt()

調達サイクルのオブジェクト（依頼、発注、受領）を作成します。

APIシンタックス

long AmCreateRequestToReceipt(long hApiCnxBase, double dQty, long lCatRefId, double dUnitPrice, char *strCur, long lRequesterId, long lCostId, long lUserId, long lStockId);

内部Basicシンタックス

Function AmCreateRequestToReceipt(dQty As Double, lCatRefId As Long, dUnitPrice As Double, strCur As String, lRequesterId As Long, lCostId As Long, lUserId As Long, lStockId As Long) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **dQty：発注し受領する数量（パッケージ単位）**
- **lCatRefId：カタログリファレンスの識別子。**
- **dUnitPrice：カタログリファレンスの単価**
- **strCur：カタログリファレンスの価格用の通貨コード**
- **lRequesterId：依頼者の識別子。**
- **lCostId：影響を受けるコストセンタの識別子。**
- **lUserId：発注品のユーザ識別子。**

- **lStockId：受領品の配達先在庫の識別子。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

注

次の関数を順番に呼び出すのと同じ操作になります。amCreateRequestToPOrder, amCreateReceiptFromPOrder

AmCreateReturnFromReceipt()

受領伝票から返品伝票を作成します。

APIシンタックス

long AmCreateReturnFromReceipt(long hApiCnxBase, long lRecptId);

内部Basicシンタックス

Function AmCreateReturnFromReceipt(lRecptId As Long) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lRecptId：受領明細の識別子。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmCreateTraceHist()

AmCreateTraceHist APIは、ソースデバイス／ケーブルからターゲットデバイス／ケーブルへの既存の接続に基づいて、トレース履歴とトレースの処理を作成します。

APIシンタックス

```
long AmCreateTraceHist(long hApiCnxBase, long lSrcLinkId, long lDestLinkId, long iTraceDir, long lCabTraceOutId, char *strComment);
```

内部Basicシンタックス

```
Function AmCreateTraceHist(lSrcLinkId As Long, lDestLinkId As Long, iTraceDir As Long, lCabTraceOutId As Long, strComment As String) As Long
```

用途

バージョン：4.00

	使用
AssetCenter API	✓

使用
リンクまたはフィールドの設定スクリプト
「スクリプト」タイプのアクションの設定 ✓
ウィザードスクリプト
ウィザードの「FINISH.DO」スクリプト ✓

パラメータ

- **lSrcLinkId：ソースケーブルリンクに割り当てられているケーブルリンクのID**
- **lDestLinkId：ターゲットケーブルリンクに割り当てられているケーブルリンクのID**
- **iTraceDir： 接続の方向**
  - **0=ホストからユーザへの方向**
  - **1=ユーザからホストへの方向**
- **lCabTraceOutId：ケーブルトレースの説明のID**
- **strComment：トレースの処理に関連するコメント**

戻りコード

- **0：成功**
- **0以外：エラーコード**

AmCreateTraceLink()

ケーブルデバイスの間にリンクを作成します。

内部Basicシンタックス

Function AmCreateTraceLink(iLinkType As Long, lAstCabId As Long, lPrtBunId As Long, lPrevLinkId As Long, iTraceDir As Long, lDutyId As Long, lLabelRuleId As Long) As Long

用途

バージョン：4.3.0

使用
AssetCenter API
リンクまたはフィールドの設定スクリプト

	使用
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **iLinkType：考慮される要素のタイプを識別します（「1」はケーブルデバイス、「0」はケーブル）。**
- **lAstCabId：ケーブルデバイスに関連付けられた資産の識別子。**
- **lPrtBunId：操作に関連するレコードの識別子。この識別子は、ケーブルの場合はamCableBundleテーブル、ケーブルデバイスの場合はamPortテーブルで得られます。**
- **lPrevLinkId：リンクの開始点として用いられる要素の識別子。**
- **iTraceDir：リンクの方向を指定します。「HOST_TO_USER」（ホストからユーザへ）または「USER_TO_HOST」（ユーザからホストへ）。**
- **lDutyId：リンクの用途の識別子。**
- **lLabelRuleId：リンクのラベル付け規則の識別子（デフォルトではNull）。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

AmCryptPassword()

ログインとパスワードで識別されるユーザのパスワードを暗号化します。

APIシンタックス

long AmCryptPassword(long hApiCnxBase, char *strUser, char *strPasswd, char *pStrCrypted, long lpStrCrypted);

内部Basicシンタックス

Function AmCryptPassword(strUser As String, strPasswd As String) As String

用途

バージョン：3.5

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **strUser：暗号化するパスワードを使用するユーザのログイン**
- **strPasswd：暗号化するパスワード（普通のテキストで）**

戻りコード

エラーの場合は、次の2**つの処理方法を用意します。**

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献** 300**]関数（必要に応じて**AmLastErrorMsg() **[献**301**]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmCurrentDate()

クライアントワークステーションの現在の日付を返します。

APIシンタックス

long AmCurrentDate();

内部Basicシンタックス

Function AmCurrentDate() As Date

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

戻りコード

エラーの場合は、次の2**つの処理方法を用意します。**

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

注

タイムゾーンを使用するようにデータベースが設定されている場合、この関数の動作は、直接 AssetCenter**で呼び出されたか、または外部プログラムにより呼び出されたかにより変化します。**AssetCenter**では、この関数は**Basic Now()**と同様に動作します。外部プログラムからの場合、この関数が返す値は**GMT+0**のタイムゾーンで表記され、夏時間との時間差は考慮されません。**

AmCurrentIsoLang()

AssetCenter**内で使用されている言語の**ISO**言語コードを返します（英語は「**en**」、フランス語は「**fr**」、など）。**

APIシンタックス

long AmCurrentIsoLang(char *pstrLanguage, long lLanguage);

内部Basicシンタックス

Function AmCurrentIsoLang() As String

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmCurrentLanguage()

AssetCenter**の言語バージョンを返します（英語バージョンの場合は「**US**」、フランス語バージョンの場合は「**FR**な」ど）。**

APIシンタックス

long AmCurrentLanguage(char *pstrLanguage, long lLanguage);

内部Basicシンタックス

Function AmCurrentLanguage() As String

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

戻りコード

エラーの場合は、次の2**つの処理方法を用意します。**

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmCurrentServerDate()

サーバの現在の日付を返します。

APIシンタックス

long AmCurrentServerDate(long hApiCnxBase);

内部Basicシンタックス

Function AmCurrentServerDate() As Date

用途

バージョン：3.5

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

AmDateAdd()

開始日に実際の経過時間を追加して、新たな日付を計算します。

APIシンタックス

```
long AmDateAdd(long tmStart, long tsDuration);
```

内部Basicシンタックス

```
Function AmDateAdd(tmStart As Date, tsDuration As Long) As Date
```

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **tmStart：経過時間を追加する日付**
- **tsDuration：tmStartの日付に追加する時間（秒単位）**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

以下の例は、関数amDateAdd() **と関数**amDateAddLogical()**の違いを示しています。各関数は**30**日の期間を日付**1/1/1999**（**1999**年**1**月**1**日）に追加します。**

AmDateAdd**は実際の期間（この例では**30**日）を追加します。**

```
RetVal=AmDateAdd("1999/01/01", 2592000)
```

関数は次の値を返します。

```
1999/01/31
```

AmDateAddLogical**は論理期間を追加します。この例では**30**日（**=1**ヶ月）です。**

```
RetVal=AmDateAddLogical("1999/01/01", 2592000)
```

関数は次の値を返します。

```
1999/02/01
```

AmDateAddLogical()

開始日に論理上の経過時間を追加して、新たな日付を計算します（1ヶ月を30日として計算します）。

APIシンタックス

long AmDateAddLogical(long tmStart, long tsDuration);

内部Basicシンタックス

Function AmDateAddLogical(tmStart As Date, tsDuration As Long) As Date

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **tmStart：経過時間を追加する日付**
- **tsDuration：tmStartの日付に追加する時間（秒単位）**

戻りコード

エラーの場合は、次の2**つの処理方法を用意します。**

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

以下の例は、関数amDateAdd() **と関数**amDateAddLogical()**の違いを示しています。各関数は**30**日の期間を日付**1/1/1999**（**1999**年**1**月**1**日）に追加します。**

AmDateAdd**は実際の期間（この例では**30**日）を追加します。**

```
RetVal=AmDateAdd("1999/01/01", 2592000)
```

関数は次の値を返します。

```
1999/01/31
```

AmDateAddLogicalは論理期間を追加します。この例では30日（=1ヶ月）です。

```
RetVal=AmDateAddLogical("1999/01/01", 2592000)
```

関数は次の値を返します。

```
1999/02/01
```

AmDateDiff()

指定した2つの日付の間の時間を秒単位で計算します。

APIシンタックス

long AmDateDiff(long tmEnd, long tmStart);

内部Basicシンタックス

Function AmDateDiff(tmEnd As Date, tmStart As Date) As Date

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **tmEnd**：計算する期間の終了日
- **tmStart**：計算する期間の開始日

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

例

1998年1月1日から1999年1月1日までの間に経過する時間を計算します。

```
AmDateDiff("1998/01/01 00:00:00", "1999/01/01 00:00:00")
```

AmDbExecAql()

データベースに対してAQLクエリを実行します。

APIシンタックス

long AmDbExecAql(long hApiCnxBase, char *strAqlStatement);

内部Basicシンタックス

Function AmDbExecAql(strAqlStatement As String) As Long

用途

バージョン：4.1.0

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strAqlStatement**：実行するAQLクエリ。

戻りコード

- 0：成功
- 0以外：エラーコード

AmDbGetDate()

AQLクエリの結果を日付形式で返します。クエリが結果を返さない場合は「0」が返され、エラーは発生しません。

APIシンタックス

long AmDbGetDate(long hApiCnxBase, char *strQuery);

内部Basicシンタックス

Function AmDbGetDate(strQuery As String) As Date

用途

バージョン：3.5

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strQuery**：実行するAQLクエリの全文

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生し

たかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

AmDbGetDouble()

AQLクエリの結果を倍精度の数値で返します。クエリが結果を返さない場合は「0」が返され、エラーは発生しません。

APIシンタックス

double AmDbGetDouble(long hApiCnxBase, char *strQuery);

内部Basicシンタックス

Function AmDbGetDouble(strQuery As String) As Double

用途

バージョン：3.5

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strQuery：実行するAQLクエリの全文**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生し**

たかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

AmDbGetList()

AQLクエリの結果をリストとして返します。AQLクエリで選択できる要素数は最高で99個です。

APIシンタックス

long AmDbGetList(long hApiCnxBase, char *strQuery, char *pstrResult, long lResult, char *strColSep, char *strLineSep, char *strIdSep);

内部Basicシンタックス

Function AmDbGetList(strQuery As String, strColSep As String, strLineSep As String, strIdSep As String) As String

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strQuery：**実行するAQLクエリ
- **strColSep：**クエリの結果の列区切りに使用する文字
- **strLineSep：**クエリの結果の行区切りに使用する文字
- **strIdSep：**クエリの結果の識別子区切りに使用する文字。

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

AmDbGetListEx()

AQLクエリの結果をリストとして返します。AmDbGetList関数と異なり、AQLクエリで選択できる要素数に制限がありません。

APIシンタックス

long AmDbGetListEx(long hApiCnxBase, char *strQuery, char *pstrResult, long lResult, char *strColSep, char *strLineSep, char *strIdSep);

内部Basicシンタックス

Function AmDbGetListEx(strQuery As String, strColSep As String, strLineSep As String, strIdSep As String) As String

用途

バージョン：3.5

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strQuery：**実行するAQLクエリ
- **strColSep：**クエリの結果の列区切りに使用する文字
- **strLineSep：**クエリの結果の行区切りに使用する文字
- **strIdSep：**クエリの結果の識別子区切りに使用する文字。

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

注

AmDbGetList関数が返すデータに、列、行や識別子区切り文字に使用する文字が含まれる場合、それらの文字は'\'文字を使用してエスケープする必要があります。

UnEscapeSeparators関数を使用して、AmDbGetListが返す文字列からエスケープ文字を削除することを推奨します。

AmDbGetLong()

AQLクエリの結果を返します。クエリが結果を返さない場合は「0」が返され、エラーは発生しません。

APIシンタックス

```
long AmDbGetLong(long hApiCnxBase, char *strQuery);
```

内部Basicシンタックス

```
Function AmDbGetLong(strQuery As String) As Long
```

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strQuery：実行するAQLクエリの全文**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

製品のサプライヤの識別子を返します。

```
AmDbGetLong("SELECT lSuppId FROM amProdSupp WHERE lProdId="+Str([ProdId])+"
)
```

AmDbGetPk()

AQLクエリのWHERE句に従って、テーブルの主キーを返します。クエリが結果を返さない場合は「0」が返され、エラーは発生しません。

APIシンタックス

long AmDbGetPk(long hApiCnxBase, char *strTableName, char *strWhere);

内部Basicシンタックス

Function AmDbGetPk(strTableName As String, strWhere As String) As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strTableName**：主キーを取得するテーブルのSQL名
- **strWhere**：AQLクエリのWHERE句

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

AmDbGetString()

AQLクエリの結果を書式化した文字列として返します。AQLクエリで選択できる要素数は最高で99個です。

警告:

文字列タイプの1フィールド値を取得するためにはこの関数を使用しないでください。 この関数は、AmDbGetListまたはAmDbGetListExに類似しています。

APIシンタックス

```
long AmDbGetString(long hApiCnxBase, char *strQuery, char *pstrResult, long lResult, char *strColSep, char *strLineSep);
```

内部Basicシンタックス

Function AmDbGetString(strQuery As String, strColSep As String, strLineSep As String) As String

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **strQuery：実行する**AQL**クエリ**
- **strColSep：クエリで取得した文字列の列区切りに使用する文字**
- **strLineSep：クエリで取得した文字列の行区切りに使用する文字**

戻りコード

エラーの場合は、次の2**つの処理方法を用意します。**

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

注

APIシンタックスでは、パラメータIResultは関数の結果値の予想されるサイズを含みます。

例

```
Dim strList As String
strList = amDbGetList("Select Name, FullName from amEmplDept Where Name Like 'C%'", "|", ",", "=")
```

これは次の文字列を返します。

```
Carpenter|/Taltek/I.S. Department/Carpenter\, Jerome\, DEMO-M016/=23459,Chavez|/Taltek/I.S. Department/Chavez\, Philip\, DEMO-M014/=23460,Chouraqui|/Taltek/Sales/Los Angeles Agency/Chouraqui\, Thomas\, DEMO-M017/=23491,Cipriani|/Taltek/Sales/Los Angeles Agency/Cipriani\, Fred\, DEMO-M018/=23492,Clech|/Taltek/Sales/Burbank Agency/Clech\, Richard\, DEMO-M021/=23482,Colombo|/Taltek/Finance/Colombo\, Gerald\, DEMO-M022/=23441
```

コンマの前には、エスケープ文字「\」を使用します。

amDbGetString()関数で実行する同じクエリは、エスケープ文字を追加しないので、リストを埋めるには不適切になります。例えば、

```
amDbGetString("Select FullName from amEmplDept Where Name Like 'C%'", "|", chr(10), "")
```

は以下の結果を表示します。

```
/Taltek/I.S. Department/Carpenter, Jerome, DEMO-M016/
/Taltek/I.S. Department/Chavez, Philip, DEMO-M014/
/Taltek/Sales/Los Angeles Agency/Chouraqui, Thomas, DEMO-M017/
/Taltek/Sales/Los Angeles Agency/Cipriani, Fred, DEMO-M018/
/Taltek/Sales/Burbank Agency/Clech, Richard, DEMO-M021/
/Taltek/Finance/Colombo, Gerald, DEMO-M022/
```

AmDbGetStringEx()

AQL**クエリの結果を文字列として返します。AmDbGetString関数と異なり、**AQL**クエリで選択できる要素数に制限がありません。**

警告:

文字列タイプの1**フィールド値を取得するためにはこの関数を使用しないでください。この関数は、**AmDbGetList**または**AmDbGetListEx**に類似しています。**

APIシンタックス

long AmDbGetStringEx(long hApiCnxBase, char *strQuery, char *pstrResult, long lResult, char *strColSep, char *strLineSep);

内部Basicシンタックス

Function AmDbGetStringEx(strQuery As String, strColSep As String, strLineSep As String) As String

用途

バージョン：3.5

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **strQuery：実行する**AQL**クエリ**
- **strColSep：クエリで取得した文字列の列区切りに使用する文字**
- **strLineSep：クエリで取得した文字列の行区切りに使用する文字**

戻りコード

エラーの場合は、次の2**つの処理方法を用意します。**

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmDeadLine()

カレンダ、開始日、および作業時間の秒数から期限を出算します。

APIシンタックス

long AmDeadLine(long hApiCnxBase, char *strCalendarSqlName, long tmStart, long tsDuration);

内部Basicシンタックス

Function AmDeadLine(strCalendarSqlName As String, tmStart As Date, tsDuration As Long) As Date

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **strCalendarSqlName：期限の計算に使う業務用カレンダの**SQL名
- **tmStart：作業期間の開始日**
- **tsDuration：開始日以降の作業時間の秒数**

戻りコード

エラーの場合は、次の2**つの処理方法を用意します。**

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

「Calendar_Paris**」というSQL名のカレンダに基づいて期限を計算します。期間の開始日は**1998**年**9**月**1**日午前**8**時、秒数は**450,000**です。**

```
AmDeadLine("Calendar_Paris", "1998/09/01 08:00:00", 450000)
```

期限は、1998年9月22日午後6時になります。

AmDecrementLogLevel()

ウィザードの最終ページのログウィンドウの階層内で1つ上のレベルに移動します。

内部Basicシンタックス

Function AmDecrementLogLevel() As Long

用途

バージョン：3.5

使用	
AssetCenter API	
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献 301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmDefAssignee()

指定した従業員グループのデフォルトのチケット責任者のID番号を検索します。

APIシンタックス

long AmDefAssignee(long hApiCnxBase, long lGroupId);

内部Basicシンタックス

Function AmDefAssignee(lGroupId As Long) As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **lGroupId：従業員グループの**ID**番号**

戻りコード

エラーの場合は、次の2**つの処理方法を用意します。**

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

従業員グループのデフォルトの責任者の識別子を返します。

```
AmDefAssignee([lGroupId])
```

次の例のように、識別子の数値を直接入力することもできます。

```
AmDefAssignee(24)
```

AmDefaultCurrency()

AssetCenterで使用しているデフォルトの通貨を返します。

APIシンタックス

long AmDefaultCurrency(long hApiCnxBase, char *return, long lreturn);

内部Basicシンタックス

Function AmDefaultCurrency() As String

用途

バージョン：3.5

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmDefEscalationScheme()

場所とヘルプデスクチケットの優先度に応じて、デフォルトのエスカレーション処理を検索します。

APIシンタックス

long AmDefEscalationScheme(long hApiCnxBase, char *strLocFullName, long lSeverityLvl);

内部Basicシンタックス

Function AmDefEscalationScheme(strLocFullName As String, lSeverityLvl As Long) As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **strLocFullName：場所の完全名**
- **lSeverityLvl：優先度の値**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献** 300**]関数（必要に応じて**AmLastErrorMsg() **[献**301**]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

場所と優先度に応じて、デフォルトのエスカレーション処理の識別子を返します。

```
AmDefEscalationScheme([Asset.Location.FullName], [Severity.lSeverityLvl])
```

次の例のように、パラメータの値を直接入力することもできます。

```
AmDefEscalationScheme ( "/Location/", 24)
```

AmDefGroup()

問題のタイプ、場所、およびメンテナンス契約に応じて、デフォルトのヘルプデスクグループのID番号を返します。

APIシンタックス

long AmDefGroup(long hApiCnxBase, long lProblemClassId, char *strLocFullName, long lAssetMainCntId);

内部Basicシンタックス

Function AmDefGroup(lProblemClassId As Long, strLocFullName As String, lAssetMainCntId As Long) As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lProblemClassId：問題のタイプのID番号**
- **strLocFullName：場所の完全名**
- **lAssetMainCntId：メンテナンス契約のID番号**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

注

デフォルトのヘルプデスクグループを定義する方法は以下の通りです。

1. 関数は、ヘルプデスクチケットの問題のタイプに関連するヘルプデスクグループを検索します。
2. 検索されたグループの中から、関数は資産の場所に最も「近い」場所に関連付けられているヘルプデスクグループを検索します。資産の場所にヘルプデスクグループがない場合は、資産の場所の親所在地で探し、最終的にはルート場所まで検索します。
3. 上記の方法でヘルプデスクグループが見つからないと、DBMSが2重の外結合をサポートする場合は、関数は場所に関連付けられていないグループを検索します。

   2重の外結合をサポートするDBMSの詳細は、マニュアル『ヘルプデスク』の、「参考情報（ヘルプデスク）」の章、「外部二重結合をサポートするDBMS」の節を参照してください。
4. DBMSが2重の外結合をサポートする場合、関数は最初に検索されたグループの中から、資産のメンテナンス契約に関連付けられているヘルプデスクグループを検索します。
5. グループが見つからない場合、関数は階層構造で1段階高い問題のタイプで手順1、2、3、4を繰り返し、グループが見つかるまでルートの問題のタイプまで検索します。

例

問題のタイプ、場所、およびメンテナンス契約の3つのパラメータに応じて、デフォルトのヘルプデスクグループのID番号を返します。

```
AmDefGroup([ProblemClass.lPbClassId],[Asset.Location.FullName],[Asset.lMaintCntrId])
```

次の例のように、パラメータのID番号の数値を直接入力することもできます。

```
AmDefGroup(0, [Asset.Location.FullName], 0)
```

AmDeleteLink()

レコードのリンクを削除します。

APIシンタックス

long AmDeleteLink(long hApiRecord, char *strLinkName, long hApiRecDest);

内部Basicシンタックス

Function AmDeleteLink(hApiRecord As Long, strLinkName As String, hApiRecDest As Long) As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **hApiRecord：削除するリンクを含むレコードのハンドル**
- **strLinkName：削除するリンクの**SQL**名**
- **hApiRecDest： リンクのターゲットレコードのハンドル**

戻りコード

- 0**：成功**
- 0**以外：エラーコード**

AmDeleteRecord()

データベースから1**つのレコードを削除します。**

APIシンタックス

long AmDeleteRecord(long hApiRecord);

内部Basicシンタックス

Function AmDeleteRecord(hApiRecord As Long) As Long

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **hApiRecord：削除するレコードのハンドル**

戻りコード

- 0**：成功**
- 0**以外：エラーコード**

AmDisconnectTrace()

AmDisconnectTrace API**はユーザノード（**lEndId**）とホストノード（**lStartId**）間のトレースを切断します。ノードがトレースの最後にある場合、ノードはケーブルリンクテーブルから削除されます。切断後、トレース履歴とトレースの処理を作成します。**

APIシンタックス

long AmDisconnectTrace(long hApiCnxBase, long lStartId, long lEndId, char *strComment, long lCabTraceOutId);

内部Basicシンタックス

Function AmDisconnectTrace(lStartId As Long, lEndId As Long, strComment As String, lCabTraceOutId As Long) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **lStartId：切断されるホストの接続**ID
- **lEndId：切断されるユーザの接続**ID
- **strComment：新規接続と切断を示すトレース処理の文字列**
- **lCabTraceOutId：ケーブルのトレースの説明の**ID

戻りコード

- 0**：成功**
- 0**以外：エラーコード**

AmDuplicateRecord()

この関数はレコードを複製します。

APIシンタックス

long AmDuplicateRecord(long hApiRecord, long bInsert);

内部Basicシンタックス

Function AmDuplicateRecord(hApiRecord As Long, bInsert As Long) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiRecord：複製するレコードのハンドル**
- **bInsert：このパラメータでは、複製されたレコードを即時に挿入するか（=1）しないか（=0）を指定します。**

戻りコード

- 0：**成功**
- 0**以外：エラーコード**

AmEndOfNthBusinessDay()

カレンダに応じて、指定した日付からn番目の日付（lDayCountで特定）の最後の業務時間を割り出します。

APIシンタックス

long AmEndOfNthBusinessDay(long hApiCnxBase, char *strCalendarSqlName, long tmStart, long lDayCount);

内部Basicシンタックス

Function AmEndOfNthBusinessDay(strCalendarSqlName As String, tmStart As Date, lDayCount As Long) As Date

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strCalendarSqlName：計算に使用するカレンダの名前**
- **tmStart：計算の開始日**
- **lDayCount：計算する時にdStartに追加する業務日数**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError()[献300]関数（必要に応じてAmLastErrorMsg()[献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmEnumValList()

カスタムリストデータのすべての値の文字列を返します。各値は、アルファベット順に並べ替えられ、strLineSepパラメータの文字で区切られます。

リストデータの値の中に区切り文字またはバックスラッシュ（\）が含まれている場合は、バックスラッシュが接頭文字として使われます。

APIシンタックス

```
long AmEnumValList(long hApiCnxBase, char *strEnumName, char *pstrValList,
long lValList, long bNoCase, char *strLineSep);
```

内部Basicシンタックス

```
Function AmEnumValList(strEnumName As String, bNoCase As Long, strLineSep
As String) As String
```

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strEnumName：このパラメータには、値を取得するリストデータのSQL名が入ります。**
- **bNoCase：アルファベット順の並べ替えが大文字小文字を区別するか（=1）しないか（=0）を指定します。**
- **strLineSep：リストデータの各値を区切る文字**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmESDAddComputers()

内部Basicシンタックス

Function AmESDAddComputers(lESDTaskId As Long, selTarget As String, lplIgnoredCount As Long) As Long

用途

バージョン：5.0.0

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

AmESDCreateTask()

内部Basicシンタックス

Function AmESDCreateTask(strDescription As String, lESDDelivMethodId As Long, lESDPackageId As Long, dttimeStart As Date, selTarget As String, bStart As Long, lplIgnoredCount As Long) As Long

用途

バージョン：5.0.0

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

AmEvalScript()

現在のコンテキストでスクリプトの名前からスクリプトを評価します。この関数には2つの用途があります。

- システムスクリプトを評価する（デフォルト値、必須、など）。
- スクリプトライブラリから関数を呼び出す。

内部Basicシンタックス

Function AmEvalScript(strScriptName As String, strObject As String, strPath As String, ...) As Variant

用途

バージョン：4.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	

パラメータ

- **strScriptName：**評価するスクリプト名。1番目の用途では、システムスクリプト名（DefVal、など）を入力します。2番目の用途では、スクリプトライブラリの一部を成す関数の名前（ライブラリの名前はパラメータstrObjectで指定）を入力します。
- **strObject：**スクリプトに関連するオブジェクト。フィールドのSQL名またはライブラリの関数名です。

- **strPath：このオプションパラメータは、スクリプト評価のコンテキストを移動させるためのパス（リンク.リンク.リンク...）を指定します。このオプションは、関数を2番目の用途に使用する場合には機能しません。**
- **...：スクリプトライブラリから関数を呼び出す際、呼び出された関数にパラメータを渡します。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献** 300]**関数（必要に応じて**AmLastErrorMsg()**[献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

注

使用可能なシステムスクリプト名のリストは次の通りです。

- **テーブル用：**IsValid、IsRelevant
- **フィールド用：**DefVal、Mandatory、Historized、ReadOnly、 Irrelevant
- **リンク用：** Historized、Filter、Irrelevant
- **任意管理項目用：** DefVal、Mandatory、Available、Historized

AmExecTransition()

現在のページから有効なトランジションを起動します。

内部Basicシンタックス

Function AmExecTransition(strTransName As String) As Long

用途

バージョン：3.00

使用
AssetCenter API
リンクまたはフィールドの設定スクリプト

	使用
「スクリプト」タイプのアクションの設定	
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strTransName：ウィザードのスクリプトに定義されているトランジションの名前**

戻りコード

- 0：**成功**
- 0**以外：エラーコード**

AmExecuteActionById()

識別子でアクションを識別し、そのアクションを実行します。

APIシンタックス

long AmExecuteActionById(long hApiCnxBase, long lActionId, char *strTableName, long lRecordId);

内部Basicシンタックス

Function AmExecuteActionById(lActionId As Long, strTableName As String, lRecordId As Long) As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lActionId：実行するアクションの識別子。**
- **strTableName：コンテキストを指定するアクションの場合は、アクションを実行するテーブルのSQL名。コンテキストを指定するアクションでこのパラメータを省略すると、この関数は機能しません。コンテキストを指定しないアクションの場合は、このパラメータは解釈されないため、必要ありません。**
- **lRecordId：アクションで使用する可能性のあるレコードの識別子。コンテキストを指定しないアクションの場合は、このパラメータは解釈されないため、必要ありません。**

戻りコード

- **0：成功**
- **0以外：エラーコード**

AmExecuteActionByName()

SQL名でアクションを識別し、そのアクションを実行します。

APIシンタックス

long AmExecuteActionByName(long hApiCnxBase, char *strSqlName, char *strTableName, long lRecordId);

内部Basicシンタックス

Function AmExecuteActionByName(strSqlName As String, strTableName As String, lRecordId As Long) As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓

	使用
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strSqlName：実行するアクションのSQL名**
- **strTableName：コンテキストを指定するアクションの場合は、アクションを実行するテーブルのSQL名。コンテキストを指定するアクションでこのパラメータを省略すると、この関数は機能しません。コンテキストを指定しないアクションの場合は、このパラメータは解釈されないため、必要ありません。**
- **lRecordId：アクションで使用する可能性のあるレコードの識別子。コンテキストを指定しないアクションの場合は、このパラメータは解釈されないため、必要ありません。**

戻りコード

- 0：**成功**
- 0**以外：エラーコード**

AmExportDocument()

レコードに添付されているドキュメントをエクスポートします。

APIシンタックス

long AmExportDocument(long hApiCnxBase, long lDocId, char *strFileName);

内部Basicシンタックス

Function AmExportDocument(lDocId As Long, strFileName As String) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lDocId：エクスポートするドキュメントの識別子。**
- **strFileName：エクスポートするドキュメントの名前。ドキュメントのテーブルの**FileName**フィールド内の名前を使用します。**

戻りコード

- 0**：成功**
- 0**以外：エラーコード**

AmExportReport()

データベースからCrystal Report**をエクスポートします。**

内部Basicシンタックス

Function AmExportReport(lReportId As Long, strFileName As String) As Long

用途

バージョン：4.3.0

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lReportId：エクスポートする**Crystal Reports**レコードの識別子。**

- **strFileName：このファイルには、エクスポート先となるファイルのフルパスが含まれます。**

戻りコード

- 0：成功
- 0以外：エラーコード

AmFindCable()

AmFindCable APIは、あるユーザ（lUserId）とホスト（lHostId）の場所間にある使用可能な次のケーブルを検索します。ケーブルのタイプ（strCabType）と役割（strCableRole）は、特定のタイプと役割でなければなりません。またケーブルのステータスは「使用可能」でなければなりません。ケーブルはIDで昇順に並べ替えられ、前のケーブルのID（lPrevCabId）よりもIDの大きいケーブルのみが選択されます。

APIシンタックス

```
long AmFindCable(long hApiCnxBase, long lPrevCableId, char *strCabType, long lUserId, long lHostId, char *strCableRole);
```

内部Basicシンタックス

```
Function AmFindCable(lPrevCableId As Long, strCabType As String, lUserId As Long, lHostId As Long, strCableRole As String) As Long
```

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lPrevCableId：前のケーブルのID**
- **strCabType：検索するケーブルのタイプ**
- **lUserId：ユーザの場所のID**
- **lHostId：ホストの場所のID**
- **strCableRole：検索するケーブルの役割**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmFindDevice()

AmFindDevice APIは、あるタイプ（strDevType）で一定の場所（lLocationId）にあるデバイスを検索します。デバイスはIDで昇順に並べ替えられ、前のデバイスID（lPrevDeviceId）よりもIDの大きいデバイスが選択されます。

APIシンタックス

```
long AmFindDevice(long hApiCnxBase, long lPrevDeviceId, char *strDeviceType,
long lLocationId);
```

内部Basicシンタックス

```
Function AmFindDevice(lPrevDeviceId As Long, strDeviceType As String, lLocationId
As Long) As Long
```

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **lPrevDeviceId：検索される旧デバイスの**ID**。検索を開始するには、値「**0**」が使用されます。**
- **strDeviceType：検索するデバイスのタイプ**
- **lLocationId：検索する場所の**ID

戻りコード

エラーの場合は、次の2**つの処理方法を用意します。**

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献** 300**]関数（必要に応じて**AmLastErrorMsg() **[献**301**]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmFindRootLink()

トレースのルートリンクを取得します。

APIシンタックス

long AmFindRootLink(long hApiCnxBase, long lLinkId);

内部Basicシンタックス

Function AmFindRootLink(lLinkId As Long) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **lLinkId：演算に関連するリンクの識別子。**

戻りコード

ルート（ケーブル）リンクの識別子を返します。

AmFindTermDevice()

AmFindTermDevice API**は、ある成端場所（**lTermField**）で一定の役割（**strCableRole**）を果たす使用可能なデバイスを検索します。デバイスは連続番号で昇順に並び替えられ、旧連続番号（**strPrevTermSeq**）よりも番号の大きい資産のみが選択されます。ピン型のデバイス（**bPinBased=1**）では、必要なピンの総数（**iPinPortCount**）とデバイス上にあるピンの残り総数を比較します。ポート型のデバイス（**bPinBased=0**）では、デバイス上に最低1つのポートがあり、このポートのホスト側またはユーザ側が使用可能であるかどうかをフラグで示します（**bCheckAvail = 0 - user device, bCheckAvail = 1 - host device**）**

APIシンタックス

long AmFindTermDevice(long hApiCnxBase, long iPrevTermSeq, long lTermFieldId, char *strCableRole, long bPinBased, long iPinPortCount, long bCheckAvail);

内部Basicシンタックス

Function AmFindTermDevice(iPrevTermSeq As Long, lTermFieldId As Long, strCableRole As String, bPinBased As Long, iPinPortCount As Long, bCheckAvail As Long) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **iPrevTermSeq：検索される成端場所の前の手順。「0」は検索を開始します。**
- **lTermFieldId：成端場所の**ID
- **strCableRole：検索するケーブルの役割**
- **bPinBased：デバイスがピン型またはポート型であるかを指定します。**
- **iPinPortCount：ピン型のデバイスでは、このパラメータは仮想ポート作成のために必要なピンの総数です。ポート型のデバイスでは必要な各ポート用に**APIが呼び出されるため、このパラメータは「1」です。
- **bCheckAvail：このパラメータでは、ポートのどの側面が使用可能であるべきかを指定できます。**
  - 0=**ユーザデバイス、使用可能なホスト側を検索します。**
  - 1=**ホストデバイス、使用可能なユーザ側を検索します。**

戻りコード

エラーの場合は、次の2**つの処理方法を用意します。**

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmFindTermField()

AmFindTermField API**はある用途（**lDutyId**）をある場所（**lLocationId**）で果たす成端場所を検索します。**lPrevTermFieldId**の値が「**0**」よりも大きい場合、このAPI**はある一定の用途と場所の追加成端場所の検索を続行します。**

APIシンタックス

long AmFindTermField(long hApiCnxBase, long lDutyId, long lLocationId, long lPrevTermFieldId);

内部Basicシンタックス

Function AmFindTermField(lDutyId As Long, lLocationId As Long, lPrevTermFieldId As Long) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **lDutyId：このパラメータは検索する機能を定義します。**
- **lLocationId：検索する場所の**ID
- **lPrevTermFieldId：成端場所の**ID

戻りコード

エラーの場合は、次の2**つの処理方法を用意します。**

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmFlushTransaction()

エージェントのタスクリストを消去します（データベースのコミット操作の後と同様）。

APIシンタックス

long AmFlushTransaction(long hApiCnxBase);

内部Basicシンタックス

Function AmFlushTransaction() As Long

用途

バージョン：4.3.0

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

戻りコード

- 0：**成功**
- 0**以外：エラーコード**

AmFormatCurrency()

指定した通貨で金額を表示します。通貨の標準記号も表示されます。

APIシンタックス

long AmFormatCurrency(double dAmount, char *strCurrency, char *pstrDisplay, long lDisplay);

内部Basicシンタックス

Function AmFormatCurrency(dAmount As Double, strCurrency As String) As String

用途

バージョン：4.3.0

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **dAmount：表示する金額。**
- **strCurrency操作に使用する通貨。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献** 300**]関数（必要に応じて**AmLastErrorMsg() **[献**301**]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
RetVal=amFormatCurrency(500, "USD")
```

この例は次のように表示します：

```
US$500.00
```

AmFormatLong()

文字列中のトークンをLong**型の変数の値に置き換えます。**

APIシンタックス

long AmFormatLong(long hApiCnxBase, long lNumber, char *strFormat, char *pstrResult, long lResult);

内部Basicシンタックス

Function AmFormatLong(lNumber As Long, strFormat As String) As String

用途

バージョン：4.3.0

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lNumber：strFormatパラメータに含まれる文字列に挿入する**Long**値。**
- **strFormat：処理する文字列。「**%d**」形式のトークンはすべてlNumberパラメータの値に置き換えられます。**

戻りコード

エラーの場合は、次の2**つの処理方法を用意します。**

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmGeneratePlanningData()

スケジュールのグラフィック表示を作成します。

内部Basicシンタックス

Function AmGeneratePlanningData(strTableSqlName As String, strProperties As String, strIds As String) As String

用途

バージョン：4.3.0

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **strTableSqlName：スケジュールの作成に使用するデータを含むテーブルの**SQL**名。**
- **strProperties：作成するスケジュールのプロパティ。**

注意:

プロパティのシンタックスの詳細については、『管理』ガイドの「参考情報：スケジュールのグラフィック表示のページのパラメータのシンタックス」の項目を参照してください。

- **strIds：スケジュールの作成に用いられるデータを含むレコードの識別子のリスト（カンマ区切り）。**

戻りコード

エラーの場合は、次の2**つの処理方法を用意します。**

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmGenSqlName()

普通の文字列からSQL名を作成します。スペース（空白文字）はアンダースコア（_）に置き換えられます。この関数は、特に任意管理項目の名前からデフォルトのSQL名を作成する場合に便利です。

APIシンタックス

long AmGenSqlName(char *return, long lreturn, char *strText);

内部Basicシンタックス

Function AmGenSqlName(strText As String) As String

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strText：**SQL名の作成に使う文字列

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

例

AssetCenterデータベース内の「Label」というオブジェクトのデフォルトのSQL名を定義します。

```
RetVal=AmGenSQLName([Label])
```

AmGetCatRef()

この関数は、あるモデル用の有効なカタログ外リファレンスを検索します（有効期限が厳守されます）。次の規則があります。

- CatProduct.lModelId=lModelId
- CatProduct.lParentId=0

関数は、仮作成されていないリファレンスを優先して返します。リファレンスが見つからず、パラメータbCreateの値が「1」であると、新規にカタログ外リファレンスと、製品（モデルを参照する）が作成されます。

APIシンタックス

long AmGetCatRef(long hApiCnxBase, long lModelId, long bCreate);

内部Basicシンタックス

Function AmGetCatRef(lModelId As Long, bCreate As Long) As Long

用途

バージョン：4.1.0

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lModelId：演算に関連するモデルの**ID

- **bCreate：検索しても結果が返されない場合に、このパラメータはカタログ外のリファレンスが作成されるかどうかを指定します。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

注

注意:

関数にサプライヤを指定する必要はありません。検索は、全サプライヤのカタログ外のリファレンスを対象に実行されます。

AmGetCatRefFromCatProduct()

この関数は、関数amGetCatRefと同じですが、相違点は特定の製品に対して検索が実行される点です。

APIシンタックス

long AmGetCatRefFromCatProduct(long hApiCnxBase, long lCatProductId, long bCreate);

内部Basicシンタックス

Function AmGetCatRefFromCatProduct(lCatProductId As Long, bCreate As Long) As Long

用途

バージョン：4.1.0

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lCatProductId：演算に関連する製品のID**
- **bCreate：検索しても結果が返されない場合に、このパラメータはカタログ外のリファレンスが作成されるかどうかを指定します。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmGetComputeString()

指定したレコードの説明文字列をテンプレートに従って返します。

APIシンタックス

long AmGetComputeString(long hApiCnxBase, char *strTableName, long lRecordId, char *strTemplate, char *pstrComputeString, long lComputeString);

内部Basicシンタックス

Function AmGetComputeString(strTableName As String, lRecordId As Long, strTemplate As String) As String

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strTableName：説明文字列を取得するレコードのテーブルのSQL名**
- **lRecordId：テーブル内のレコードの識別子。**
- **strTemplate：説明文字列に使うテンプレート（文字列形式）**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
RetVal = amGetComputeString("amEmplDept", [lEmplDeptId], "[Name], [FirstName]")
```

AmGetCurrentNTDomain()

現在のログインのNTドメイン名を返します。

APIシンタックス

long AmGetCurrentNTDomain(char *pstrDomain, long lDomain);

内部Basicシンタックス

Function AmGetCurrentNTDomain() As String

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献 301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
RetVal = amGetCurrentNTDomain()
```

AmGetCurrentNTUser()

Windows（NTまたは2000）に接続しているユーザのログインを取得します。

APIシンタックス

long AmGetCurrentNTUser(char *pstrUser, long lUser);

内部Basicシンタックス

Function AmGetCurrentNTUser() As String

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmGetFeat()

名前に基づいて任意管理項目オブジェクトを作成し、そのオブジェクトのハンドルを返します。

APIシンタックス

```
long AmGetFeat(long hApiTable, long lPos);
```

内部Basicシンタックス

```
Function AmGetFeat(hApiTable As Long, lPos As Long) As Long
```

用途

バージョン：3.5

	使用
AssetCenter API	✓

	使用
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiTable：テーブルのハンドル**
- **lPos：テーブル内の任意管理項目の位置**

AmGetFeatCount()

hApiTableパラメータで指定するテーブルの任意管理項目数を返します。

APIシンタックス

long AmGetFeatCount(long hApiTable);

内部Basicシンタックス

Function AmGetFeatCount(hApiTable As Long) As Long

用途

バージョン：3.5

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiTable：テーブルのハンドル**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

AmGetField()

クエリ、レコード、またはテーブルのハンドルからフィールドオブジェクトを作成し、そのオブジェクトのハンドルを返します。

APIシンタックス

long AmGetField(long hApiObject, long lPos);

内部Basicシンタックス

Function AmGetField(hApiObject As Long, lPos As Long) As Long

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiObject**：クエリ、レコード、またはテーブルのハンドル
- **lPos**：オブジェクト内のフィールドの位置（インデックス）

AmGetFieldCount()

現在のオブジェクトに含まれているフィールド数を返します。

APIシンタックス

long AmGetFieldCount(long hApiObject);

内部Basicシンタックス

Function AmGetFieldCount(hApiObject As Long) As Long

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **hApiObject：有効なレコード、クエリ、またはテーブルのハンドル**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmGetFieldDateOnlyValue()

現在のオブジェクトに含まれるフィールドの値を返します。値は「Date」形式で返されます（外部ツールからはLong）。AmGetFieldDateValue関数と異なり、日付部分だけが返され、時刻部分は除外されます。

APIシンタックス

long AmGetFieldDateOnlyValue(long hApiObject, long lFieldPos);

内部Basicシンタックス

Function AmGetFieldDateOnlyValue(hApiObject As Long, lFieldPos As Long) As Date

用途

バージョン：4.3.0

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiObject：クエリまたはレコードのハンドル。**
- **lFieldPos：現在のオブジェクトに含まれるフィールドの数。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmGetFieldDateValue()

現在のオブジェクトに含まれているフィールドの値を返します。値は日付（「Date」）形式で返されます（外部ツールからの場合は「Long」形式です）。

APIシンタックス

long AmGetFieldDateValue(long hApiObject, long lFieldPos);

内部Basicシンタックス

Function AmGetFieldDateValue(hApiObject As Long, lFieldPos As Long) As Date

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiObject：クエリまたはレコードのハンドル**
- **lFieldPos：現在のオブジェクトに含まれているフィールドの番号**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmGetFieldDescription()

ハンドルでフィールドを識別し、そのフィールドの長い説明を、文字列（String形式）で返します。

APIシンタックス

long AmGetFieldDescription(long hApiField, char *pstrBuffer, long lBuffer);

内部Basicシンタックス

Function AmGetFieldDescription(hApiField As Long) As String

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiField：長い説明を取得するフィールドの有効なハンドル**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

AmGetFieldDoubleValue()

現在のオブジェクトに含まれているフィールドの値を返します。値は倍精度（「Double」）形式で返されます。

APIシンタックス

double AmGetFieldDoubleValue(long hApiObject, long lFieldPos);

内部Basicシンタックス

Function AmGetFieldDoubleValue(hApiObject As Long, lFieldPos As Long) As Double

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiObject：クエリまたはレコードのハンドル**
- **lFieldPos：現在のオブジェクトに含まれているフィールドの番号**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献** 300**]関数（必要に応じて**AmLastErrorMsg() **[献**301**]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmGetFieldFormat()

この関数は、フィールドの入力タイプが次のような場合に有効です。

- システムリストデータ
- リストデータ
- 経過時間
- テーブルまたはフィールドのSQL名

次のように、各入力タイプのフォーマットを返します。

UserType	関数が返すフォーマット
システムリストデータ	リストデータ項目のリスト
リストデータ	フィールドに関連付けられているリストデータの名前
経過時間	表示フォーマット
テーブルまたはフィールドのSQL名	テーブルのSQL名を保存するフィールドのSQL名

APIシンタックス

```
long AmGetFieldFormat(long hApiField, char *pstrBuffer, long lBuffer);
```

内部Basicシンタックス

```
Function AmGetFieldFormat(hApiField As Long) As String
```

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiField**：「UserType」を取得するフィールドの有効なハンドル

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

AmGetFieldFormatFromName()

フィールドの名前からフィールドの入力タイプを取得します。

APIシンタックス

long AmGetFieldFormatFromName(long hApiCnxBase, char *strTableName, char *strFieldName, char *pFieldFormat, long lpFieldFormat);

内部Basicシンタックス

Function AmGetFieldFormatFromName(strTableName As String, strFieldName As String) As String

用途

バージョン：3.5

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strTableName**：処理するフィールドを含んでいるテーブルのSQL名
- **strFieldName**：フィールドのSQL名

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmGetFieldFromName()

名前からフィールドオブジェクトを作成し、作成したフィールドオブジェクトのハンドルを返します。

APIシンタックス

long AmGetFieldFromName(long hApiObject, char *strName);

内部Basicシンタックス

Function AmGetFieldFromName(hApiObject As Long, strName As String) As Long

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **hApiObject：クエリ、レコード、またはテーブルのハンドル**
- **strName：フィールド名**

AmGetFieldLabel()

ハンドルでフィールドを識別し、そのフィールドのラベルを文字列（String形式）で返します。

APIシンタックス

long AmGetFieldLabel(long hApiField, char *pstrBuffer, long lBuffer);

内部Basicシンタックス

Function AmGetFieldLabel(hApiField As Long) As String

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **hApiField：取得するラベルの付いたフィールドの有効なハンドル**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献** 300**]関数（必要に応じて**AmLastErrorMsg() **[献**301**]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmGetFieldLabelFromName()

フィールドのSQL名からフィールドのラベルを取得します。

APIシンタックス

long AmGetFieldLabelFromName(long hApiCnxBase, char *strTableName, char *strFieldName, char *pFieldLabel, long lpFieldLabel);

内部Basicシンタックス

Function AmGetFieldLabelFromName(strTableName As String, strFieldName As String) As String

用途

バージョン：3.5

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strTableName：処理するフィールドを含んでいるテーブルのSQL名**
- **strFieldName：フィールドのSQL名**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmGetFieldLongValue()

現在のオブジェクトに含まれているフィールドの値を返します。

APIシンタックス

long AmGetFieldLongValue(long hApiObject, long lFieldPos);

内部Basicシンタックス

Function AmGetFieldLongValue(hApiObject As Long, lFieldPos As Long) As Long

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **hApiObject：クエリまたはレコードのハンドル**
- **lFieldPos：現在のオブジェクトに含まれているフィールドの番号**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献** 300**]関数（必要に応じて**AmLastErrorMsg() **[献**301**]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

注

注意:

この関数を使って日付、時刻、または日付+時刻のデータ型フィールドの値を取得する場合は、この関数が返す倍長整数は1970**年**1**月**1**日午前**0**時**0**分**0**秒の時点からの秒数になります。**

AmGetFieldName()

現在のオブジェクトに含まれているフィールドの名前を返します。

APIシンタックス

long AmGetFieldName(long hApiObject, long lFieldPos, char *pstrBuffer, long lBuffer);

内部Basicシンタックス

Function AmGetFieldName(hApiObject As Long, lFieldPos As Long) As String

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **hApiObject：クエリ、レコード、またはテーブルのハンドル**
- **lFieldPos：現在のオブジェクトに含まれているフィールドの番号。例えば、「**0**」という値は最初のフィールドを示します。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

AmGetFieldRights()

現在のオブジェクト内のフィールドに対するユーザ権限を返します。ユーザ権限は3つの文字のいずれかで返されます。3つの文字はそれぞれ、読取り、挿入、更新の権限を示します。

- 「r」はデータを読み取る権限を示します。
- 「i」はデータを挿入する権限を示します。
- 「u」はデータを更新する権限を示します。

例えば、フィールドのユーザ権限が読取り専用の場合は、「r」を返します。

APIシンタックス

```
long AmGetFieldRights(long hApiObject, long lFieldPos, char *pstrBuffer, long lBuffer);
```

内部Basicシンタックス

```
Function AmGetFieldRights(hApiObject As Long, lFieldPos As Long) As String
```

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiObject：クエリ、レコード、またはテーブルのハンドル**
- **lFieldPos：現在のオブジェクトに含まれているフィールドの番号**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmGetFieldSize()

フィールドのサイズを返します。

APIシンタックス

long AmGetFieldSize(long hApiField);

内部Basicシンタックス

Function AmGetFieldSize(hApiField As Long) As Long

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiField：サイズを取得するフィールドのハンドル**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmGetFieldSqlName()

ハンドルでフィールドを識別し、そのフィールドのSQL名を、文字列（String形式）で返します。

APIシンタックス

long AmGetFieldSqlName(long hApiField, char *pstrBuffer, long lBuffer);

内部Basicシンタックス

Function AmGetFieldSqlName(hApiField As Long) As String

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiField**：SQL名を取得するフィールドの有効なハンドル

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

AmGetFieldStrValue()

現在のオブジェクトに含まれているフィールドの値を返します。この値は、文字列形式で返されます。

注意：この関数をAssetCenterのAPIを介して使用する場合は、さらに**pszBuffer**と**lBuffer**パラメータが必要となります。これらのパラメータはそれぞれ、取得した文字列を保管するバッファとして使用する文字列と、そのバッファのサイズを指定します。**pszBuffer**文字列は、書式化され（文字で埋められ）、**lBuffer**。次に示すコードは不正確で、バッファとして使用する文字列のサイズの指定が正しくありません。

```
Dim strBuffer as String
Dim lRec as Long
Dim lBuffer as Long
lBuffer=20
lRec=AmGetFieldStrValue(1, 0, strBuffer, lBuffer)
```

次に正しいコードを示します。

```
Dim strBuffer as String
Dim lRec as Long
Dim lBuffer as Long
strBuffer=String(21, " ") ' バッファを21文字に設定 (" ")
lBuffer=20
lRec=AmGetFieldStrValue(1, 0, strBuffer, lBuffer)
```

String関数を使ってバッファ文字列を書式化する時に、パッド文字として「0」を使わないようにします。特に、この関数がループ内で同一の文字列をバッファとして使用する場合は、AmGetFieldStrValue関数を呼び出す前に、バッファのサイズを指定してください。

APIシンタックス

long AmGetFieldStrValue(long hApiObject, long lFieldPos, char *pstrBuffer, long lBuffer);

内部Basicシンタックス

Function AmGetFieldStrValue(hApiObject As Long, lFieldPos As Long) As String

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiObject：クエリまたはレコードのハンドル**
- **lFieldPos：現在のオブジェクトに含まれているフィールドの番号**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献 301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmGetFieldType()

フィールドのデータ型を返します。

APIシンタックス

long AmGetFieldType(long hApiField);

内部Basicシンタックス

Function AmGetFieldType(hApiField As Long) As Long

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **hApiField：データ型を取得するフィールドのハンドル**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

注

注意:

次の表に、AmGetFieldType関数がフィールドの各データ型について返す値を示します。

返す値	対応するフィールドのデータ型
0	Undefined
1	Byte
2	Short
3	Long
4	Float
5	Double
6	String
7	Time stamp
8	Bin
9	Blob
10	Date
11	Time
12	Memo

AmGetFieldUserType()

この関数は、ハンドルで識別されるフィールドの"UserType"**（参照：**database.txt**ファイル）を、倍長整数型で返します。フィールド用の有効な戻り値は次の通りです。**

値	説明
0	**デフォルト**
1	**数値**
2	**はい／いいえ**
3	**金額**
4	**日付**
5	**日付+時刻**
7	**システムリストデータ**
8	**リストデータ**
10	**パーセント**
11	**経過時間**
12	**テーブルまたはフィールドの**SQL**名**

リンク用の有効な戻り値は次の通りです。

値	説明
0	普通
1	コメント
2	画像
3	履歴
4	任意管理項目値

バージョン4.0.0までは、関数はリンク用に必ず0を返していました。AssetCenter バージョン4.1.0以降からは、関数は、リンク用に以下の値の1つを返します。

- 0 : 普通
- 1 : コメント
- 2 : 画像
- 3 : 履歴
- 5 : スクリプト

APIシンタックス

```
long AmGetFieldUserType(long hApiField);
```

内部Basicシンタックス

```
Function AmGetFieldUserType(hApiField As Long) As Long
```

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- hApiField：入力タイプを取得するフィールドの有効なハンドル

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

AmGetForeignKey()

ハンドルでリンクを識別し、そのリンクの外部キーのハンドルを取得します。

APIシンタックス

long AmGetForeignKey(long hApiField);

内部Basicシンタックス

Function AmGetForeignKey(hApiField As Long) As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiField：処理するリンクのハンドル**

AmGetIndex()

クエリ、レコード、またはテーブルのハンドルからインデックスオブジェクトを作成し、そのオブジェクトのハンドルを返します。

APIシンタックス

long AmGetIndex(long hApiTable, long lPos);

内部Basicシンタックス

Function AmGetIndex(hApiTable As Long, lPos As Long) As Long

用途

バージョン：3.5

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiTable：テーブルのハンドル**
- **lPos：テーブル内のインデックスの位置**

AmGetIndexCount()

hApiTableパラメータに指定したテーブルに含まれているインデックス数を返します。

APIシンタックス

long AmGetIndexCount(long hApiTable);

内部Basicシンタックス

Function AmGetIndexCount(hApiTable As Long) As Long

用途

バージョン：3.5

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiTable：テーブルのハンドル**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmGetIndexField()

インデックス内のフィールドの位置（インデックスのlpos番目のフィールド）で識別されるフィールド上のハンドルを返します。

APIシンタックス

long AmGetIndexField(long hApiIndex, long lPos);

内部Basicシンタックス

Function AmGetIndexField(hApiIndex As Long, lPos As Long) As Long

用途

バージョン：3.5

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiIndex：操作に関連するインデックス上で有効なハンドル**
- **lPos：インデックス内でのフィールドの位置**

AmGetIndexFieldCount()

インデックスを構成するフィールドの数を返します。

APIシンタックス

long AmGetIndexFieldCount(long hApiIndex);

内部Basicシンタックス

Function AmGetIndexFieldCount(hApiIndex As Long) As Long

用途

バージョン：3.5

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiIndex：操作に関連するインデックス上で有効なハンドル**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献 301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmGetIndexFlags()

インデックスのパラメータを返します。

APIシンタックス

long AmGetIndexFlags(long hApiIndex);

内部Basicシンタックス

Function AmGetIndexFlags(hApiIndex As Long) As Long

用途

バージョン：3.5

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **hApiIndex：操作に関連するインデックス上で有効なハンドル**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

注

関数に戻される値は、次の値の論理結合（OR）の結果です。

- 1 : インデックスは重複する値を許可します。
- 2 : インデックスはNull値を許可します。
- 4 : インデックスは大文字小文字を区別しません。

関数が値「3」を戻す場合、インデックスは重複する値とNull値を許可すると推論できます（1 OR 2 = 3）。

AmGetIndexName()

インデックスの名前を返します。

APIシンタックス

long AmGetIndexName(long hApiIndex, char *pstrBuffer, long lBuffer);

内部Basicシンタックス

Function AmGetIndexName(hApiIndex As Long) As String

用途

バージョン：3.5

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiIndex：名前を取得する必要があるインデックスの有効なハンドル**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmGetLink()

テーブルのハンドルからリンクオブジェクトを作成し、そのオブジェクトのハンドルを返します。

APIシンタックス

long AmGetLink(long hApiTable, long lPos);

内部Basicシンタックス

Function AmGetLink(hApiTable As Long, lPos As Long) As Long

用途

バージョン：3.02

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiTable：テーブルのハンドル**
- **lPos：オブジェクト内のリンクの位置（インデックス）**

AmGetLinkCardinality()

1つのリンクで関連付け可能なリンク数を返します。

APIシンタックス

long AmGetLinkCardinality(long hApiField);

内部Basicシンタックス

Function AmGetLinkCardinality(hApiField As Long) As Long

用途

バージョン：3.5

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiField：関連付け可能なリンク数を取得するリンクのハンドル**

戻りコード

- 1：1-1（1対1）のリンクが可能です。
- 2：1-n（1対n）のリンクが可能です。

AmGetLinkCount()

現在のテーブル内のリンク数を返します。

APIシンタックス

long AmGetLinkCount(long hApiTable);

内部Basicシンタックス

Function AmGetLinkCount(hApiTable As Long) As Long

用途

バージョン：3.02

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **hApiTable：有効なテーブルのハンドル**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献** 301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmGetLinkDstField()

hApiFieldパラメータで定義したリンクのリンク先フィールド（外部キー）を返します。

APIシンタックス

long AmGetLinkDstField(long hApiField);

内部Basicシンタックス

Function AmGetLinkDstField(hApiField As Long) As Long

用途

バージョン：3.5

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiField：処理するリンクのハンドル**

AmGetLinkFeatureValue()

「リンク」タイプの任意管理項目の値を返します。

APIシンタックス

long AmGetLinkFeatureValue(long hApiObject, long lFieldPos, long lRecordId);

内部Basicシンタックス

Function AmGetLinkFeatureValue(hApiObject As Long, lFieldPos As Long, lRecordId As Long) As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **hApiObject：クエリまたはレコードのハンドル**
- **lFieldPos：現在のオブジェクト内のフィールドの位置**
- **lRecordId：値を取得する任意管理項目のレコード識別子。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim q as String
q = "Select fv_link, lEmplDeptId From amEmplDept Where lEmplDeptId = " & [lEmplDeptId]
Dim hq as Long
hq = amQueryCreate()
Dim lErr as Long
lErr = amQueryGet(hq, q)
Dim lId as Long
```

```
lId = amGetFieldLongValue(hq, 1)
amMsgBox("str: " & amGetFieldStrValue(hq, 0))
amMsgBox("int: " &
amGetFieldLongValue(hq,0))
amMsgBox("lnk: " & amGetLinkFeatureValue(hq,0,lId))
```

AmGetLinkFromName()

名前からリンクオブジェクトを作成し、そのオブジェクトのハンドルを返します。

APIシンタックス

long AmGetLinkFromName(long hApiTable, char *strName);

内部Basicシンタックス

Function AmGetLinkFromName(hApiTable As Long, strName As String) As Long

用途

バージョン：3.02

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **hApiTable：テーブルのハンドル**
- **strName：リンクの**SQL**名**

AmGetLinkType()

リンクのタイプを返します。

APIシンタックス

long AmGetLinkType(long hApiField);

内部Basicシンタックス

Function AmGetLinkType(hApiField As Long) As Long

用途

バージョン：3.02

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **hApiField：タイプを取得するリンクのハンドル**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmGetMainField()

指定したテーブルのメインフィールドに対応するフィールドオブジェクトを作成し、そのオブジェクトのハンドルを返します。

APIシンタックス

long AmGetMainField(long hApiTable);

内部Basicシンタックス

Function AmGetMainField(hApiTable As Long) As Long

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiTable：取得するメインフィールドを含むテーブルのハンドル**

AmGetMemoField()

この関数は、あるテーブルのメモタイプフィールドに対応するフィールドオブジェクトを作成します。関数はこの方法で作成されたフィールドにハンドルを返します。

APIシンタックス

long AmGetMemoField(long hApiTable);

内部Basicシンタックス

Function AmGetMemoField(hApiTable As Long) As Long

用途

バージョン：4.1.0

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiTable：テーブルのメモフィールドの検索する場合の、テーブル上のハンドル**

AmGetNextAssetPin()

AmGetNextAssetPin APIは、デバイス（lAssetId）で使用可能な次のピンを検索します。連続番号がピンを並べ替えます。ポートの方向（bPinPortDir）に応じて、使用可能なピンは昇順（bPinPortDir = 0）または降順（bPinPortDir = 1）で並べ替えられます。

APIシンタックス

long AmGetNextAssetPin(long hApiCnxBase, long lAssetId, long bPinPortDir, long iPrevPinSeq);

内部Basicシンタックス

Function AmGetNextAssetPin(lAssetId As Long, bPinPortDir As Long, iPrevPinSeq As Long) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lAssetId：デバイスの**ID
- **bPinPortDir：検索する方向**
  - 0=**昇順**
  - 1=**降順**
- **iPrevPinSeq**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmGetNextAssetPort()

AmGetNextAssetPort APIは、デバイス（lAssetId）で一定の用途（lDutyId）を果たすまたは全く用途を果たさない、使用可能な次のポートを検索します。ポートのステータスは「使用可能」でなければなりません。ブールのフラグは、ポートのユーザ側（bCheckUser）またはホスト側（bCheckHost）が検証されなければならないかどうかを指定します。ブールのフラグが「真（true）」の場合、APIはユーザ値（bUserAvail）またはホスト値（bHostAvail）を比較します。ポートは連続番号で並べ替えられます。ポートの方向（bPinPortDir）に応じて、使用可能なポートは昇順（bPinPortDir = 0）または降順（bPinPortDir = 1）で並べ替えられます。

APIシンタックス

long AmGetNextAssetPort(long hApiCnxBase, long lAssetId, long lCabCnxTypeId, long lDutyId, long bCheckUser, long bCheckHost, long bUserAvail, long bHostAvail, long bPinPortDir, long iPrevPortSeq);

内部Basicシンタックス

Function AmGetNextAssetPort(lAssetId As Long, lCabCnxTypeId As Long, lDutyId As Long, bCheckUser As Long, bCheckHost As Long, bUserAvail As Long, bHostAvail As Long, bPinPortDir As Long, iPrevPortSeq As Long) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lAssetId：検索するデバイスの**ID
- **lCabCnxTypeId：ポート用ケーブルの接続タイプの**ID
- **lDutyId：ポートの用途の**ID
- **bCheckUser：ユーザ側を検証するフラグ**
- **bCheckHost：ホスト側を検証するフラグ**
- **bUserAvail：確認するユーザ側の使用可能ステータス**
- **bHostAvail：確認するホスト側の使用可能ステータス**
- **bPinPortDir：検索するピンの方向**
  - 0=**昇順**
  - 1=**降順**
- **iPrevPortSeq**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献** 300]**関数（必要に応じて**AmLastErrorMsg()**[献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmGetNextCableBundle()

AmGetNextCableBundle APIは、ケーブル（lCableId）で一定の用途（lDutyId）を果たすまたは全く用途を果たさない、使用可能な次のバンドルを検索します。バンドルのステータスは「使用可能」でなければなりません。ブールのフラグは、バンドルのユーザ側（bCheckUser）またはホスト側（bCheckHost）が検証されなければならないかどうかを指定します。ブールのフラグが「真（true）」の場合、APIはユーザ値（bUserAvail）またはホスト値（bHostAvail）を比較します。

APIシンタックス

long AmGetNextCableBundle(long hApiCnxBase, long lCableId, long lDutyId, long bCheckUser, long bCheckHost, long bUserAvail, long bHostAvail);

内部Basicシンタックス

Function AmGetNextCableBundle(lCableId As Long, lDutyId As Long, bCheckUser As Long, bCheckHost As Long, bUserAvail As Long, bHostAvail As Long) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lCableId：検証するケーブルのID**
- **lDutyId：検索する用途のID**
- **bCheckUser：このパラメータはユーザ側のバンドルの接続を確認するかどうかを指定します。**
- **bCheckHost：このパラメータはホスト側のバンドルの接続を確認するかどうかを指定します。**
- **bUserAvail：検索するユーザ側の接続ステータス**
- **bHostAvail：検索するホスト側の接続ステータス**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

AmGetNextCablePair()

AmGetNextCablePair APIは、ケーブル（lCableId）内の一定のタイプ（lPairTypeId）の使用可能な次のケーブルペアを検索します。ペアはケーブルペアのIDで並べ替えられます。

APIシンタックス

long AmGetNextCablePair(long hApiCnxBase, long lCableId, long lPairTypeId, long iStartPairSeq);

内部Basicシンタックス

Function AmGetNextCablePair(lCableId As Long, lPairTypeId As Long, iStartPairSeq As Long) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lCableId：検索するケーブルのID**

- **lPairTypeId：検索するケーブルペアのタイプ**
- **iStartPairSeq**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmGetNTDomains()

データベースに接続するユーザのドメインを取得します。

APIシンタックス

long AmGetNTDomains(char *pstrDomains, long lDomains);

内部Basicシンタックス

Function AmGetNTDomains() As String

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

AmGetNTMachinesInDomain()

あるドメインのコンピュータを1列に表示して（コンピュータ名はコンマで区切る）取得します。ドメインが空の場合、関数はERR_CANCEL(2)を返しますが、関数の実行は中断されません。

APIシンタックス

long AmGetNTMachinesInDomain(char *strDomain, char *pstrMachines, long lMachines, long bUseDC);

内部Basicシンタックス

Function AmGetNTMachinesInDomain(strDomain As String, bUseDC As Long) As String

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strDomain：検索するドメイン名**
- **bUseDC**: **このパラメータが1に設定されている場合、関数はコンピューター覧についてドメインコントローラに問い合わせます。0に設定されている場合(デ**

フォルト値)、関数はシステム関数ライブラリを使用して、コンピュータの一覧を探します。

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

AmGetNTUsersInDomain()

ドメイン上のユーザのリストを取得します。リストは2つの列で返されます（login,fullname）。列は「|」で区切られ、行は「,」で区切られます。

APIシンタックス

long AmGetNTUsersInDomain(char *strDomain, char *pstrUsers, long lUsers);

内部Basicシンタックス

Function AmGetNTUsersInDomain(strDomain As String) As String

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strDomain：検索するドメイン名**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmGetPOLinePrice()

発注明細の価格を計算します。

APIシンタックス

double AmGetPOLinePrice(long hApiCnxBase, long lPOrdLineId);

内部Basicシンタックス

Function AmGetPOLinePrice(lPOrdLineId As Long) As Double

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lPOrdLineId：発注明細の識別子。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmGetPOLinePriceCur()

発注明細に適用される通貨コードを見つけます。

APIシンタックス

long AmGetPOLinePriceCur(long hApiCnxBase, long lPOrdLineId, char *pstrPrice, long lPrice);

内部Basicシンタックス

Function AmGetPOLinePriceCur(lPOrdLineId As Long) As String

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lPOrdLineId：発注明細の識別子。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmGetPOLineReference()

発注明細に対応するカタログリファレンスの説明を取得します。

APIシンタックス

long AmGetPOLineReference(long hApiCnxBase, long lPOrdLineId, char *pstrRef, long lRef);

内部Basicシンタックス

Function AmGetPOLineReference(lPOrdLineId As Long) As String

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lPOrdLineId：発注明細の識別子。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmGetRecordFromMainId()

テーブルの主キーの値でレコードを識別し、そのレコードのID番号を返します。

APIシンタックス

long AmGetRecordFromMainId(long hApiCnxBase, char *strTable, long lId);

内部Basicシンタックス

Function AmGetRecordFromMainId(strTable As String, lId As Long) As Long

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strTable：処理するレコードを含んでいるテーブルのSQL名**
- **lId：レコードを含んでいるテーブルの主キーの値**

注

この関数はテーブルが存在しない場合を除いて、レコードハンドルを返します。テーブル内のレコードが1つも指定されていないと、この関数に返されるハンドルを使用する関数を新規に実行するたびに、エラーが発生します。

AmGetRecordHandle()

ハンドルでクエリを識別し、そのクエリで取得したレコードのハンドルを返します。このレコードは、データベースへの書込みに使用できます。この関数は、クエリにレコードの主キーが含まれていないと、機能しません。

APIシンタックス

long AmGetRecordHandle(long hApiQuery);

内部Basicシンタックス

Function AmGetRecordHandle(hApiQuery As Long) As Long

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiQuery：クエリオブジェクトの有効なハンドル**

AmGetRecordId()

ハンドルでレコードを識別し、そのレコードのID番号を返します。挿入中のレコードの場合、値は「0」です。

APIシンタックス

long AmGetRecordId(long hApiRecord);

内部Basicシンタックス

Function AmGetRecordId(hApiRecord As Long) As Long

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiRecord：**ID番号を取得するレコードの有効なハンドル

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

AmGetRelDstField()

リンクのターゲットフィールドのハンドルを返します。

APIシンタックス

long AmGetRelDstField(long hApiField);

内部Basicシンタックス

Function AmGetRelDstField(hApiField As Long) As Long

用途

バージョン：3.5

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiField：処理に関連するリンクで有効なハンドル**

AmGetRelSrcField()

リンクのソースフィールドのハンドルを返します。

APIシンタックス

long AmGetRelSrcField(long hApiField);

内部Basicシンタックス

Function AmGetRelSrcField(hApiField As Long) As Long

用途

バージョン：3.5

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiField：処理に関連するリンクの有効なハンドル**

AmGetRelTable()

N対Nリンクの関係テーブルのハンドルを返します。

APIシンタックス

long AmGetRelTable(long hApiField);

内部Basicシンタックス

Function AmGetRelTable(hApiField As Long) As Long

用途

バージョン：3.5

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiField：処理に関連するリンクの有効なハンドル**

戻りコード

エラーが発生した場合に、無効なハンドル（ゼロ）を返します。

AmGetReverseLink()

hApiFieldパラメータに含まれているハンドルで逆リンクを識別し、その逆リンクのハンドルを返します。

APIシンタックス

long AmGetReverseLink(long hApiField);

内部Basicシンタックス

Function AmGetReverseLink(hApiField As Long) As Long

用途

バージョン：3.02

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiField：取得する逆リンクのリンクのハンドル**

AmGetScriptValue()

APIシンタックス

long AmGetScriptValue(long hApiObject, char *strScriptName, char *strObject, char *strPath);

内部Basicシンタックス

Function AmGetScriptValue(hApiObject As Long, strScriptName As String, strObject As String, strPath As String) As Variant

用途

バージョン：?

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmGetSelfFromMainId()

指定したテーブル内のレコードの説明文字列を返します。

APIシンタックス

long AmGetSelfFromMainId(long hApiCnxBase, char *strTableName, long lId, char *pstrRecordDesc, long lRecordDesc);

内部Basicシンタックス

Function AmGetSelfFromMainId(strTableName As String, lId As Long) As String

用途

バージョン：3.5

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **strTableName：処理するレコードを含んでいるテーブルの**SQL**名**
- **lId：処理する**ID**番号**

戻りコード

エラーの場合は、次の2**つの処理方法を用意します。**

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmGetSourceTable()

hApiFieldパラメータに指定されているリンクのソーステーブルのハンドルを返します。

APIシンタックス

long AmGetSourceTable(long hApiField);

内部Basicシンタックス

Function AmGetSourceTable(hApiField As Long) As Long

用途

バージョン：3.02

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiField：取得するソーステーブルのリンクの有効なハンドル**

戻りコード

エラーが発生した場合に、無効なハンドル（ゼロ）を返します。

AmGetTable()

現在接続しているテーブルを位置で識別し、そのテーブルのハンドルを返します。

APIシンタックス

long AmGetTable(long hApiCnxBase, long lPos);

内部Basicシンタックス

Function AmGetTable(lPos As Long) As Long

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **lPos：現在接続しているテーブルの位置。値は「**0**」とAmGetTableCountの間になります。**

戻りコード

エラーが発生した場合に、無効なハンドル（ゼロ）を返します。

AmGetTableCount()

現在接続しているデータベースのテーブル数を返します。

APIシンタックス

long AmGetTableCount(long hApiCnxBase);

内部Basicシンタックス

Function AmGetTableCount() As Long

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmGetTableDescription()

ハンドルでテーブルを識別し、そのテーブルの長い説明を、文字列（String**形式）で返します。**

APIシンタックス

```
long AmGetTableDescription(long hApiTable, char *pstrDesc, long lDesc);
```

内部Basicシンタックス

```
Function AmGetTableDescription(hApiTable As Long) As String
```

用途

バージョン：3.00

	使用
AssetCenter API	✓

	使用
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiTable：取得する長い説明のテーブルの有効なハンドル**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmGetTableFromName()

現在接続しているテーブルをSQL名で識別し、そのテーブルのハンドルを返します。

APIシンタックス

long AmGetTableFromName(long hApiCnxBase, char *strName);

内部Basicシンタックス

Function AmGetTableFromName(strName As String) As Long

用途

バージョン：2.52

	使用
AssetCenter API	✓

	使用
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strName：ハンドルを取得するテーブルのSQL名**

戻りコード

エラーが発生した場合に、無効なハンドル（ゼロ）を返します。

AmGetTableLabel()

ハンドルでテーブルを識別し、そのテーブルのラベルを文字列（String形式）で返します。

APIシンタックス

long AmGetTableLabel(long hApiTable, char *pstrLabel, long lLabel);

内部Basicシンタックス

Function AmGetTableLabel(hApiTable As Long) As String

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiTable：取得するラベルのテーブルの有効なハンドル**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmGetTableName()

テーブルのSQL名を文字列として返します。

APIシンタックス

long AmGetTableName(long hApiTable, char *pstrBuffer, long lBuffer);

内部Basicシンタックス

Function AmGetTableName(hApiTable As Long) As String

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiTable：取得するSQL名のテーブルのハンドル**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmGetTableRights()

ハンドルでテーブルを識別し、そのテーブルのユーザ権限を文字列（String形式）で返します。返される文字列には、追加権限と削除権限を示す2つの文字のいずれか、または両方が含まれます。

- **「c」は、ユーザにそのテーブルでの追加権限があることを示します。**
- **「d」は、ユーザにそのテーブルでの削除権限があることを示します。**

つまり、

- **「c」は、ユーザがそのテーブルで追加権限しか使えないことを示します。**
- **「cd」は、ユーザがそのテーブルで追加権限と削除権限の両方を使えることを示します。**

APIシンタックス

```
long AmGetTableRights(long hApiTable, char *pstrBuffer, long lBuffer);
```

内部Basicシンタックス

```
Function AmGetTableRights(hApiTable As Long) As String
```

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiTable：ユーザ権限を取得するテーブルの有効なハンドル**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmGetTableSqlName()

ハンドルでテーブルを識別し、そのテーブルのSQL名を文字列（String形式）で返します。

APIシンタックス

```
long AmGetTableSqlName(long hApiTable, char *pstrBuffer, long lBuffer);
```

内部Basicシンタックス

```
Function AmGetTableSqlName(hApiTable As Long) As String
```

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiTable：取得するSQL名のテーブルの有効なハンドル**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmGetTargetTable()

リンク先テーブルのSQL名を返します。

APIシンタックス

```
long AmGetTargetTable(long hApiField);
```

内部Basicシンタックス

```
Function AmGetTargetTable(hApiField As Long) As Long
```

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiField：処理するリンクのハンドル**

戻りコード

エラーが発生した場合に、無効なハンドル（ゼロ）を返します。

AmGetTrace()

AmGetTrace API**は、2ノード（**lUserId、lHostId**）間のトレースをケーブルリンクのテーブル内で検索します。トレースの方向（**iTraceDir**）は、トレースがユーザからホスト方向（**iTraceDir = 1**）またはホストからユーザ方向（**iTraceDir = 0**）であるかを識別します。トレースタイプ（**iTraceType**）は、トレースが接続（**iTraceType = 1**）であるかまたは切断（**iTraceType = 2**）であるかを指定します。完全トレースのインジケータ（**bFullTrace**）は、トレースが変更されたノードのみを含むか（**bFullTrace=0**）または完全トレースを含むか（**bFullTrace=1**）を指定します。**

APIシンタックス

```
long AmGetTrace(long hApiCnxBase, long lUserId, long lHostId, long iTraceDir,
long iTraceType, long bFullTrace, char *pstrTrace, long lTrace);
```

内部Basicシンタックス

```
Function AmGetTrace(lUserId As Long, lHostId As Long, iTraceDir As Long,
iTraceType As Long, bFullTrace As Long) As String
```

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lUserId：開始する接続リンクのID**
- **lHostId：終了する接続リンクのID**
- **iTraceDir： 接続の方向**
  - **0=ホストからユーザへの方向**
  - **1=ユーザからホストへの方向**
- **iTraceType ：接続のタイプ**
  - **1=接続**
  - **2=切断**
- **bFullTrace：このパラメータは不完全なトレースを無視し、トレース文字列全体を返すように指定します。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmGetTraceFromHist()

AmGetTraceFromHist APIは、新規の接続と既存の接続を区別するために、トレースの処理を用いてトレース履歴から文字列を計算します。

APIシンタックス

long AmGetTraceFromHist(long hApiCnxBase, long lProjTraceOutId, long iTraceDir, char *strDelimiter, char *pstrTraceint, long lTraceint, long bUpdateFlag);

内部Basicシンタックス

Function AmGetTraceFromHist(lProjTraceOutId As Long, iTraceDir As Long, strDelimiter As String, bUpdateFlag As Long) As String

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **lProjTraceOutId：プロジェクトのトレース**ID
- **iTraceDir： 接続の方向**
  - 0=**ホストからユーザへの方向**
  - 1=**ユーザからホストへの方向**
- **strDelimiter：既存の接続と切断を示す文字列の区切り文字**
- **bUpdateFlag：このオプションパラメータはフィールド** amCabTraceOut.TraceString**を更新します。**
  - 0=**偽（**false**）**
  - 1=**真（**true**）**

戻りコード

エラーの場合は、次の2**つの処理方法を用意します。**

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献** 300**]関数（必要に応じて**AmLastErrorMsg() **[献**301**]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmGetTypedLinkField()

hApiFieldパラメータに指定されているリンクのリンク先テーブルのSQL名を値として含んでいるフィールドのハンドルを返します。

APIシンタックス

long AmGetTypedLinkField(long hApiField);

内部Basicシンタックス

Function AmGetTypedLinkField(hApiField As Long) As Long

用途

バージョン：3.02

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiField：処理の最初でパラメータに入力されるリンクの有効なハンドル**

AmGetUserEnvSessionItem()

APIシンタックス

long AmGetUserEnvSessionItem(long hApiCnxBase, char *return, long lreturn, char *strSection, char *strEntry);

内部Basicシンタックス

Function AmGetUserEnvSessionItem(strSection As String, strEntry As String) As String

用途

バージョン：4.4.0

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

戻りコード

エラーの場合は、次の2**つの処理方法を用意します。**

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmGetVersion()

AssetCenter**のビルド番号を文字列で返します。**

APIシンタックス

long AmGetVersion(char *pstrBuf, long lBuf);

内部Basicシンタックス

Function AmGetVersion() As String

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmHasAdminPrivilege()

接続しているユーザが管理者権限を持っている場合に、TRUE（0でない値）**を返します。**

APIシンタックス

long AmHasAdminPrivilege(long hApiCnxBase);

内部Basicシンタックス

Function AmHasAdminPrivilege() As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓

	使用
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

AmHasRelTable()

この関数では、リンクに関係テーブルがあるかないかをテストできます。

APIシンタックス

long AmHasRelTable(long hApiField);

内部Basicシンタックス

Function AmHasRelTable(hApiField As Long) As Long

用途

バージョン：3.5

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiField：処理に関連するリンクの有効なハンドル**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmHasRightsForCreation()

接続しているユーザが、指定したテーブルの作成権限を持つかどうかを識別します。

内部Basicシンタックス

Function AmHasRightsForCreation(strTable As String) As Long

用途

バージョン：4.3.0

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strTable：操作に関連するテーブルのSQL名。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
RetVal=amHasRightsForCreation("amEmplDept")
```

AmHasRightsForDeletion()

接続しているユーザが、指定したテーブルの削除権限を持つかどうかを識別します。

内部Basicシンタックス

Function AmHasRightsForDeletion(strTable As String) As Long

用途

バージョン：4.3.0

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strTable：操作に関連するテーブルのSQL名。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
RetVal=amHasRightsForDeletion("amEmplDept")
```

AmHasRightsForFieldUpdate()

接続しているユーザが、指定したフィールドの更新権限を持つかどうかを識別します。

内部Basicシンタックス

Function AmHasRightsForFieldUpdate(strTable As String, strField As String) As Long

用途

バージョン：4.3.0

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓

使用	
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strTable：操作に関連するテーブルのSQL名。**
- **strField：操作に関連するフィールドのSQL名（テーブルはstrTableパラメータで指定されます）。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献 301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
RetVal=amHasRightsForFieldUpdate("amEmplDept", "Location")
```

AmHelpdeskCanCloseFile()

接続しているユーザがヘルプデスクチケットをクローズできるかどうかを識別します。

内部Basicシンタックス

Function AmHelpdeskCanCloseFile() As Long

用途

バージョン：4.3.0

使用
AssetCenter API

	使用
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

注

注意:

接続しているユーザがヘルプデスクチケットをクローズできる場合、関数は値「1」を返します。

AmHelpdeskCanProceed()

接続しているユーザがヘルプデスクチケットを処理できるかどうかを識別します。

内部Basicシンタックス

```
Function AmHelpdeskCanProceed() As Long
```

用途

バージョン：4.3.0

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓

	使用
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

注

注意:

接続しているユーザがヘルプデスクチケットを処理できる場合、関数は値「1」を返します。

AmHelpdeskCanSaveCall()

接続しているユーザがヘルプデスクチケットを保存できるかどうかを識別します。

内部Basicシンタックス

Function AmHelpdeskCanSaveCall() As Long

用途

バージョン：4.3.0

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓

	使用
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

注

注意:

接続しているユーザがヘルプデスクチケットを保存できる場合、関数は値「1」を返します。

AmImportDocument()

ファイルからドキュメントを作成しインポートします。

APIシンタックス

```
long AmImportDocument(long hApiCnxBase, long lDocObjId, char *strTableName,
char *strFileName, char *strCategory, char *strDesignation);
```

内部Basicシンタックス

```
Function AmImportDocument(lDocObjId As Long, strTableName As String,
strFileName As String, strCategory As String, strDesignation As String) As Long
```

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lDocObjId**：テーブル［amDocument］の［lDocObjId］フィールドに格納される値
- **strTableName**：［amDocument］テーブルの［DocObjTable］フィールドに格納される値。実際上、これはドキュメントのリンク先レコードを含むテーブルのSQL名です。
- **strFileName**：インポートするファイルの名前
- **strCategory**：AssetCenterに表示されるドキュメントのカテゴリ
- **strDesignation**：AssetCenterに表示されるドキュメントの名前

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

AmImportReport()

ファイルからCrystal Reportをインポートします。amReportテーブル中の既存のレコードにインポートされます。

内部Basicシンタックス

```
Function AmImportReport(lReportId As Long, strFileName As String) As Long
```

用途

バージョン：4.3.0

使用
AssetCenter API
リンクまたはフィールドの設定スクリプト
「スクリプト」タイプのアクションの設定 
ウィザードスクリプト
ウィザードの「FINISH.DO」スクリプト 

パラメータ

- **lReportId：インポートしたレポートを保存するamReportテーブル中のレコードの識別子。**
- **strFileName：インポートするレポートを含むファイルのフルパス。**

戻りコード

- 0：**成功**
- 0**以外：エラーコード**

AmIncrementLogLevel()

strMsgメッセージを履歴ウィンドウに表示し、ウィザードの最終ページでノードを作成します。

ノードには、次のメッセージがすべて表示されます。

内部Basicシンタックス

Function AmIncrementLogLevel(strMsg As String, iType As Long) As Long

用途

バージョン：3.5

使用
AssetCenter API
リンクまたはフィールドの設定スクリプト
「スクリプト」タイプのアクションの設定
ウィザードスクリプト

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strMsg：表示されるメッセージのテキスト**
- **iType：メッセージに関連付けるタイプを定義します。メッセージのタイプがエラーの場合は「1」、警告の場合は「2」、情報の場合は「4」の値になります。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献 301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmInsertRecord()

以前に作成したレコードをデータベースに挿入します。AmCreateRecord関数を使って作成したレコードのみを挿入できます。クエリを使ってアクセスしたレコードは挿入できません。

APIシンタックス

long AmInsertRecord(long hApiRecord);

内部Basicシンタックス

Function AmInsertRecord(hApiRecord As Long) As Long

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiRecord：データベースに挿入するレコードのハンドル**

戻りコード

- 0：成功
- 0以外：エラーコード

AmInstantiateReqLine()

ある依頼明細のインスタンスを直接作成します。

APIシンタックス

long AmInstantiateReqLine(long hApiCnxBase, long lRequestLineId, long bFinal, long lPOrderLineId, double dQty);

内部Basicシンタックス

Function AmInstantiateReqLine(lRequestLineId As Long, bFinal As Long, lPOrderLineId As Long, dQty As Double) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- lRequestLineId：依頼明細の識別子。
- bFinal：このパラメータでは割当を終了するかどうかを指定します。
- lPOrderLineId：発注明細の識別子。
- dQty：インスタント化する数量

戻りコード

- 0：成功
- 0以外：エラーコード

注

依頼された要素を調達サイクルを通さずに作成します。bFinal = FALSEの場合、「納品待ち」ステータスの要素が作成されます。

AmInstantiateRequest()

ある依頼の全内容を直接インスタント化します。

APIシンタックス

```
long AmInstantiateRequest(long hApiCnxBase, long lRequestId, long lMulFactor);
```

内部Basicシンタックス

```
Function AmInstantiateRequest(lRequestId As Long, lMulFactor As Long) As Long
```

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	

	使用
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lRequestId：依頼の識別子。**
- **lMulFactor：実行するインスタント生成の数**

戻りコード

- 0：**成功**
- 0**以外：エラーコード**

AmIsConnected()

現在の接続が有効かどうかをテストします。

APIシンタックス

long AmIsConnected(long hApiCnxBase);

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**

- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献 301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmIsFieldForeignKey()

フィールドがデータベースの外部キーかどうかを識別します。

APIシンタックス

long AmIsFieldForeignKey(long hApiField);

内部Basicシンタックス

Function AmIsFieldForeignKey(hApiField As Long) As Long

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **hApiField：識別するフィールドのハンドル**

戻りコード

- 1**：フィールドは外部キーです。**
- 0**：フィールドは外部キーではありません。**

AmIsFieldIndexed()

フィールドにインデックスが付いているかどうかを識別します。

APIシンタックス

long AmIsFieldIndexed(long hApiField);

内部Basicシンタックス

Function AmIsFieldIndexed(hApiField As Long) As Long

用途

バージョン：3.5

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **hApiField：識別するフィールドのハンドル**

戻りコード

- 1：**フィールドにインデックスが付いています。**
- 0：**フィールドにインデックスは付いていません。**

AmIsFieldPrimaryKey()

フィールドがデータベースの主キーかどうかを識別します。

APIシンタックス

long AmIsFieldPrimaryKey(long hApiField);

内部Basicシンタックス

Function AmIsFieldPrimaryKey(hApiField As Long) As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiField：識別するフィールドのハンドル**

戻りコード

- 1：**フィールドは主キーです。**
- 0：**フィールドは主キーではありません。**

AmIsHelpdeskAdmin()

接続しているユーザがヘルプデスク管理者かどうかを識別します。

内部Basicシンタックス

Function AmIsHelpdeskAdmin() As Long

用途

バージョン：4.3.0

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

注

注意:

接続しているユーザがヘルプデスク管理者の場合、関数は値「1」を返します。

AmIsHelpdeskMember()

接続しているユーザがヘルプデスクグループに属するかどうかを識別します。

内部Basicシンタックス

```
Function AmIsHelpdeskMember() As Long
```

用途

バージョン：4.3.0

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓

	使用
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

注

注意:

接続しているユーザがヘルプデスクグループに属する場合、関数は値「1」を返します。

AmIsHelpdeskSuper()

接続しているユーザがヘルプデスクグループの責任者かどうかを識別します。

内部Basicシンタックス

Function AmIsHelpdeskSuper() As Long

用途

バージョン：4.3.0

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

注

注意:

接続しているユーザがヘルプデスク責任者の場合、関数は値「1」を返します。

AmIsLink()

ハンドルでオブジェクトを識別し、そのオブジェクトがリンクかフィールドかを識別します。

APIシンタックス

```
long AmIsLink(long hApiField);
```

内部Basicシンタックス

```
Function AmIsLink(hApiField As Long) As Long
```

用途

バージョン：3.00

	使用
AssetCenter API	✓

	使用
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiField：処理するオブジェクトのハンドル**

戻りコード

- 1：**オブジェクトはリンクです。**
- 0：**オブジェクトはフィールドです。**

AmIsModuleAuthorized()

接続しているユーザがアプリケーションの指定したモジュールにアクセスできるかどうかを識別します。

内部Basicシンタックス

Function AmIsModuleAuthorized(strModuleName As String) As Long

用途

バージョン：4.3.0

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strModuleName：操作に関連するモジュールの名前。可能なモジュールのリストを次に示します。**
  - ITAM：**ポートフォリオ管理モジュール**

- Reconc：照合更新モジュール
- Contract：契約管理モジュール
- Leasing：リース管理モジュール
- Procurement：調達管理モジュール
- Finance：ファイナンスモジュール
- Helpdesk：ヘルプデスク管理モジュール
- Cable：ケーブルモジュール
- Barcode：バーコードモジュール
- Admin：管理モジュール
- Visio：Visio統合モジュール
- API：APIライブラリ
- Wizard：ウィザード管理モジュール
- Workflow：ワークフロー管理モジュール
- AutoCAD：AutoCAD統合モジュール
- Knowlix：Knowlix統合モジュール
- DA_Automation：自動化モジュール
- DA_RemoteControl：リモートコントロールモジュール

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

注

注意:

モジュールによっては、アプリケーションから使用できないものや有効にできないものもあります。使用できるかどうかは、Peregrine Systems, Inc.から購入されたライセンスに依存します。

AmIsTypedLink()

ハンドルでオブジェクトを識別し、そのオブジェクトが入力されたリンクかどうかを識別します。

APIシンタックス

long AmIsTypedLink(long hApiField);

内部Basicシンタックス

Function AmIsTypedLink(hApiField As Long) As Long

用途

バージョン：3.02

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiField：処理するオブジェクトのハンドル**

戻りコード

- 1：**オブジェクトは入力されたリンクです。**
- 0：**オブジェクトは入力されたリンクではありません。**

AmLastError()

該当する接続のコンテキスト内で最後に実行した関数によって作成された最後のエラーコードを返します。

APIシンタックス

long AmLastError(long hApiCnxBase);

内部Basicシンタックス

Function AmLastError() As Long

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmLastErrorMsg()

現在の接続で発生した最後のエラーメッセージを返します。

APIシンタックス

long AmLastErrorMsg(long hApiCnxBase, char *pstrBuffer, long lBuffer);

内部Basicシンタックス

Function AmLastErrorMsg() As String

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献 301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmListToString()

AmDbGetList関数で取得した文字列の結果を、AmDbGetString関数と同じ形式で表示される文字列に変換します。

APIシンタックス

```
long AmListToString(char *return, long lreturn, char *strSource, char *strColSep, char *strLineSep, char *strIdSep);
```

内部Basicシンタックス

```
Function AmListToString(strSource As String, strColSep As String, strLineSep As String, strIdSep As String) As String
```

用途

バージョン：3.00

使用
AssetCenter API
リンクまたはフィールドの設定スクリプト
「スクリプト」タイプのアクションの設定
ウィザードスクリプト
ウィザードの「FINISH.DO」スクリプト

パラメータ

- **strSource：変換する文字列**
- **strColSep：変換する文字列の列区切りに使用する文字**
- **strLineSep：変換する文字列の行区切りに使用する文字**
- **strIdSep：変換する文字列の識別子区切りに使用する文字。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmLog()

履歴ウィンドウにstrMessageメッセージを表示します。

内部Basicシンタックス

Function AmLog(strMessage As String, iLogType As Long) As Long

用途

バージョン：3.00

使用
AssetCenter API
リンクまたはフィールドの設定スクリプト
「スクリプト」タイプのアクションの設定

	使用
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strMessage：表示するメッセージのテキスト**
- **iLogType：メッセージに関連付けるタイプを定義します。メッセージのタイプがエラーの場合は「1」、警告の場合は「2」、情報の場合は「4」の値になります。**

戻りコード

- 0：**成功**
- 0**以外：エラーコード**

例

```
AmLog("メッセージの本文")
```

AmLoginId()

接続しているユーザの識別子を返します。

APIシンタックス

long AmLoginId(long hApiCnxBase);

内部Basicシンタックス

Function AmLoginId() As Long

用途

バージョン：2.52

	使用
AssetCenter API	✓

	使用
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

例

接続しているユーザの識別子をデータベース内のフィールドのデフォルト値として定義します。

```
RetVal=AmLoginId()
```

AmLoginName()

接続しているユーザのログイン名を返します。

APIシンタックス

long AmLoginName(long hApiCnxBase, char *return, long lreturn);

内部Basicシンタックス

Function AmLoginName() As String

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

例

接続しているユーザのログイン名をデータベース内のフィールドのデフォルト値として定義します。

```
RetVal=AmLoginName()
```

AmMapSubReqLineAgent()

依頼明細と発注明細間のリンクを可能にします。

APIシンタックス

long AmMapSubReqLineAgent(long hApiCnxBase, long lRequestLineId, long lPorderLineId);

内部Basicシンタックス

Function AmMapSubReqLineAgent(lRequestLineId As Long, lPorderLineId As Long) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lRequestLineId：依頼明細の識別子。**
- **lPorderLineId：発注明細の識別子。**

戻りコード

- 0：**成功**
- 0**以外：エラーコード**

AmMoveCable()

AmMoveCable API**はケーブル（**lCableId**）を現在の場所からターゲット場所（**lToLocId**）へ移動させます。プロジェクト（**lProjectId**）と作業指示（**lWorkOrderId**）に値が入力されていると、ケーブルはコメント（**strComment**）と共にプロジェクトと作業指示へ追加されます。コメントはケーブルに実行されるアクション（例：**Aのケーブルを**Bへ移動**）を説明します。**

APIシンタックス

long AmMoveCable(long hApiCnxBase, long lCableId, long lToLocId, long lProjectId, long lWorkOrderId, char *strComment);

内部Basicシンタックス

Function AmMoveCable(lCableId As Long, lToLocId As Long, lProjectId As Long, lWorkOrderId As Long, strComment As String) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **lCableId：移動させるケーブルの**ID
- **lToLocId：ケーブルの新規の場所の**ID
- **lProjectId：プロジェクトの**ID
- **lWorkOrderId：作業指示の**ID
- **strComment：作業指示に添付されるコメント**

戻りコード

- 0**：成功**
- 0**以外：エラーコード**

AmMoveDevice()

AmMoveDevice API**はデバイス（**lDeviceId**）を現在の場所からターゲット場所（**lToLocationId**）へ移動させます。プロジェクト（**lProjectId**）と作業指示（**lWorkOrderId**）に値が入力されていると、デバイスはコメント（**strComment**）と共にプロジェクトと作業指示へ追加されます。コメントはデバイスに実行されるアクション（例：**A**のデバイスを**B**へ移動）を説明します。**

APIシンタックス

long AmMoveDevice(long hApiCnxBase, long lDeviceId, long lToLocationId, long lProjectId, long lWorkOrderId, char *strComment);

内部Basicシンタックス

Function AmMoveDevice(lDeviceId As Long, lToLocationId As Long, lProjectId As Long, lWorkOrderId As Long, strComment As String) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **lDeviceId：移動させられるデバイスの**ID
- **lToLocationId：デバイスの新規場所の**ID
- **lProjectId：プロジェクトの**ID
- **lWorkOrderId：作業指示の**ID
- **strComment：作業指示に添付されるコメント**

戻りコード

- 0**：成功**
- 0**以外：エラーコード**

AmMsgBox()

メッセージが入ったダイアログボックスを表示します。

内部Basicシンタックス

Function AmMsgBox(strMessage As String, lMode As Long) As Long

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strMessage：ダイアログボックスに表示するメッセージ**
- **lMode ：表示されるダイアログボックスのタイプ（**0：OK**ボタンつきの単純ボックス、**1：OK**とキャンセルボタンつきのボックス、**2：**キャンセルボタンつきのボックス）**

戻りコード

- 0：**成功**
- 0**以外：エラーコード**

例

```
AmMsgBox("メッセージの本文")
```

AmOpenConnection()

AssetCenter**データベース名からセッションを作成します。****strDataSource****は有効な**AssetCenter**データソース名でなければなりません（有効な**AssetCenter**データベース接続は、**AssetCenter**のログインボックスに一覧表示されます）。**

同じデータベースまたは異なるデータベースのどちらでも、複数の接続を開くことがます。

APIシンタックス

long AmOpenConnection(char *strDataSource, char *strUser, char *strPwd);

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	

パラメータ

- **strDataSource ：データソースの名前**
- **strUser：接続するユーザ名**
- **strPwd：指定したユーザのパスワード**

AmOpenScreen()

この関数はAssetCenter**で画面やビューを開きます。**

内部Basicシンタックス

Function AmOpenScreen(strScreenId As String, strContext As String, strFilter As String, iMode As Long, strBindField As String, bStayReadOnly As Long) As Long

用途

バージョン：4.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strScreenId：開くシステム画面のビューやユーザ画面のビューのSQL名（この優先度で開きます）。**
- **strContext：このオプションパラメータには、画面を開く時にリスト内で選択されるレコードの識別子のリストを入力します。**
- **strFilter：画面を開く時にリストに適用されるAQLフィルタ**
- **iMode：画面を開く際のモード（参照、編集、など）。入力可能な値は：0（現在進行中のアクションなし）、1（現在進行中のアクションなし）、2（変更中）、3（作成中）、4（複製中）、5（追加中）、6（選択中）。**
- **strBindField：このパラメータでは、画面を開く際のフィルタやモード（リンクしている画面を開く、など）を選択できます。フィールドのSQL名、または現在のコンテキストを使用する場合はCurrentSrcChoice値を入力します。**
- **bStayReadOnly: このパラメータで、画面を読取り専用モードで開くことができます。ユーザ権限に関わらず、変更は禁止されます。**

戻りコード

- **0：成功**
- **0以外：エラーコード**

AmOverflowTables()

指定したテーブルのオーバーフローテーブルのSQL名を返します。

APIシンタックス

long AmOverflowTables(long hApiCnxBase, char *strBasisTable, char *strOverflowTables, long lOverflowTables);

内部Basicシンタックス

Function AmOverflowTables(strBasisTable As String) As String

用途

バージョン：4.3.0

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strBasisTable：操作に関連するテーブルの**SQL**名。**

戻りコード

エラーの場合は、次の2**つの処理方法を用意します。**

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

注

注意:

返されるリストではカンマが区切りとして用いられます。指定したテーブルにオーバーフローテーブルが存在しない場合、空文字列が返されます。

例

次の例は、ポートフォリオ品目テーブル（amPortfolio**）のオーバーフローテーブルを返します。**

```
RetVal = AmOverflowTables("amPortfolio")
```

この例の結果は次の通りです。

```
amComputer,amSoftInstall,amPhone
```

AmPagePath()

ウィザードの実行パス（参照したページのリスト）を含む文字列を返します。前に戻るジャンプは無視します。

内部Basicシンタックス

Function AmPagePath() As String

用途

バージョン：3.00

使用	
AssetCenter API	
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

AmProgress()

パーセンテージ単位で進行状況を表わすインジケータをウィザードの最終ページに表示します。

内部Basicシンタックス

Function AmProgress(iProgress As Long) As Long

用途

バージョン：3.00

使用	
AssetCenter API	
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **iProgress：進行状況インジケータのサイズを定義するのに使う、処理状況を示すパーセント**（0 - 100）

戻りコード

- 0：**成功**
- 0**以外：エラーコード**

例

```
AmProgress(85)
```

この例の場合は、85%**の完了状況を示すインジケータが表示されます。**

AmPurgeRecord()

レコードを破棄します。

APIシンタックス

long AmPurgeRecord(long hApiRecord);

内部Basicシンタックス

Function AmPurgeRecord(hApiRecord As Long) As Long

用途

バージョン：4.3.0

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiRecord：操作に関連するレコードのハンドル。**

戻りコード

- 0：**成功**
- 0**以外：エラーコード**

注

注意:

リンクされたレコードの処理は、リンクのタイプによって異なります。「所有」タイプのリンクの場合、リンクされたレコードは同一の方法で処理されます。「定義」または「普通」タイプのリンクの場合、リンクされたレコードの外部キーが0にリセットされ、アーカイブされるレコードの識別子と説明文字列がアーカイブフィールドに入力されます。

重要項目:

この関数はアーカイブテーブルまたは標準テーブルからのレコードに対して使用できます。

AmQueryCreate()

現在の接続でクエリオブジェクトを作成します。作成したオブジェクトを使って、AQL**ステートメントをデータベースサーバに送信できます。**

APIシンタックス

long AmQueryCreate(long hApiCnxBase);

内部Basicシンタックス

Function AmQueryCreate() As Long

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

AmQueryExec()

AQL**クエリを実行します。クエリの最初の結果を返します。次の結果は AmQueryNext関数を使って取得できます。**

送信したクエリがMemo**型のフィールドを返す場合は、結果の文字数は**255**文字まで(半角の場合)に制限されます。**

APIシンタックス

long AmQueryExec(long hApiQuery, char *strQueryCommand);

内部Basicシンタックス

Function AmQueryExec(hApiQuery As Long, strQueryCommand As String) As Long

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiQuery**：AQLステートメントの送信先となるオブジェクトの有効なハンドル
- **strQueryCommand**：AQLクエリの本文（文字列）

戻りコード

- 0：**成功**
- 0**以外**：**エラーコード**

AmQueryGet()

AQLクエリをカーソルなしで実行します（結果は1つのみ）。クエリの最初の結果のみを返します。

APIシンタックス

long AmQueryGet(long hApiQuery, char *strQueryCommand);

内部Basicシンタックス

Function AmQueryGet(hApiQuery As Long, strQueryCommand As String) As Long

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓

	使用
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiQuery**：AQLステートメントの送信先となるオブジェクトの有効なハンドル
- **strQueryCommand**：AQLクエリの本文（文字列）

戻りコード

- 0：**成功**
- 0**以外**：**エラーコード**

AmQueryNext()

事前にAmQueryExec関数を使って実行したクエリの結果を返します。

APIシンタックス

```
long AmQueryNext(long hApiQuery);
```

内部Basicシンタックス

```
Function AmQueryNext(hApiQuery As Long) As Long
```

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiQuery**：AQLステートメントの送信先となるオブジェクトの有効なハンドル

戻りコード

- 0：成功
- 0以外：エラーコード

AmQuerySetAddMainField()

クエリのモードを変更し、テーブルのメインフィールドが、返されるフィールドのリストに自動的に追加されるようにします。この種のクエリはNull識別子のレコードを返しません。

APIシンタックス

```
long AmQuerySetAddMainField(long hApiQuery, long bAddMainField);
```

内部Basicシンタックス

```
Function AmQuerySetAddMainField(hApiQuery As Long, bAddMainField As Long)
As Long
```

用途

バージョン：3.5

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiQuery：クエリオブジェクトで有効なハンドル**
- **bAddMainField：以下の2つの値があります。**
  - True：**テーブルのメインフィールドが追加される。**
  - False：**テーブルのメインフィールドは追加されない。**

戻りコード

- 0：**成功**
- 0**以外：エラーコード**

AmQuerySetFullMemo()

デフォルトでは、関数AmQueryExecの実行時に、クエリはMemoタイプのフィールドを254文字で切り捨てます。この関数はクエリのモードを変更し、Memoフィールドを完全に回復します。

APIシンタックス

long AmQuerySetFullMemo(long hApiQuery, long bFullMemo);

内部Basicシンタックス

Function AmQuerySetFullMemo(hApiQuery As Long, bFullMemo As Long) As Long

用途

バージョン：3.5

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiQuery：クエリオブジェクトで有効なハンドル**
- **bFullMemo：以下の2つの値があります。**
  - True：**クエリはMemoフィールド全体を返す。**
  - False：**クエリはMemoフィールドを254文字で切り捨てる。**

戻りコード

- 0：**成功**
- 0**以外：エラーコード**

AmQueryStartTable()

ハンドルでクエリを識別し、そのクエリによって処理されるテーブルのハンドルを返します。

APIシンタックス

long AmQueryStartTable(long hApiQuery);

内部Basicシンタックス

Function AmQueryStartTable(hApiQuery As Long) As Long

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiQuery：クエリオブジェクトの有効なハンドル**

戻りコード

エラーが発生した場合に、無効なハンドル（ゼロ）を返します。

AmQueryStop()

ハンドルでクエリを識別し、そのクエリの実行を中断します。このクエリは、AmQueryExec関数を使って予め起動しておく必要があります。

APIシンタックス

long AmQueryStop(long hApiQuery);

内部Basicシンタックス

Function AmQueryStop(hApiQuery As Long) As Long

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiQuery：クエリオブジェクトの有効なハンドル**

戻りコード

- 0：成功
- 0以外：エラーコード

AmReceiveAllPOLines()

1件の発注明細に含まれるすべての製品を受領します（一括受領されます）。

注意:

受領明細は注文品を受領する手続きを行った時点で担当者によって作成されます。従って、その前に受領明細のレコードにアクセスすることはできません。

APIシンタックス

```
long AmReceiveAllPOLines(long hApiCnxBase, long lPOrdId, long lDelivId);
```

内部Basicシンタックス

```
Function AmReceiveAllPOLines(lPOrdId As Long, lDelivId As Long) As Long
```

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- lPOrdId：受領する製品が含まれている発注明細の識別子。
- lDelivId：発注明細中のすべての製品を受領するのに使用する受領伝票の識別子。

戻りコード

- 0：成功
- 0以外：エラーコード

AmReceivePOLine()

1件の発注明細の発注数のうち一部の数量だけを受領し（部分受領）、その受領明細の識別子を返します。

注意:

受領明細は注文品を受領する手続きを行った時点で担当者によって作成されます。従って、その前に受領明細のレコードにアクセスすることはできません。

APIシンタックス

long AmReceivePOLine(long hApiCnxBase, long lPOrdLineId, long lDelivId, double dQty);

内部Basicシンタックス

Function AmReceivePOLine(lPOrdLineId As Long, lDelivId As Long, dQty As Double) As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- lPOrdLineId：受領する製品が含まれている発注明細の識別子。

- **lDelivId：**1件の発注明細の一部の数量の注文品を受領するのに使用する受領伝票の識別子。
- **dQty：**受領伝票に記入する受領数

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmRefreshAllCaches()

AssetCenter**で使用しているキャッシュを更新します。**

APIシンタックス

long AmRefreshAllCaches(long hApiCnxBase);

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	
ウィザードスクリプト	
ウィザードの「FINISH.DO**」スクリプト**	

戻りコード

- 0**：成功**
- 0**以外：エラーコード**

AmRefreshLabel()

AmRefreshLabel APIは、あるテーブル（strTableName）内のレコード（lMainId）のラベル文字列を更新します。

APIシンタックス

long AmRefreshLabel(long hApiCnxBase, long lMainId, char *strTableName, char *pstrLabel, long lLabel);

内部Basicシンタックス

Function AmRefreshLabel(lMainId As Long, strTableName As String) As String

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lMainId：更新されるID**
- **strTableName：lMainIdに関連するテーブルの名前**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmRefreshProperty()

strVarNameプロパティをパラメータで識別し、そのプロパティの値を再評価します。プロパティがスクリプトを使用する場合は、もう一度そのスクリプトを実行します。

スクリプトを使わないプロパティの場合は、階層構造を更新します。

内部Basicシンタックス

Function AmRefreshProperty(strVarName As String) As Long

用途

バージョン：3.00

使用	
AssetCenter API	
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **strVarName：再評価するウィザードのプロパティ名**

戻りコード

- 0：**成功**
- 0**以外：エラーコード**

AmRefreshTraceHist()

AmRefreshTraceHist API**はプロジェクトのトレースの完全な履歴を更新します。**API**にはまた、「個々の」トレース履歴の項目を更新するオプションパラメータがあります。このパラメータがない場合は、トレースの全履歴が更新されます。**

APIシンタックス

long AmRefreshTraceHist(long hApiCnxBase, long lCabTraceOutId, long lTraceHistId);

内部Basicシンタックス

Function AmRefreshTraceHist(lCabTraceOutId As Long, lTraceHistId As Long) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **lCabTraceOutId：ケーブルトレースの説明の**ID
- **lTraceHistId：このオプションパラメータはトレース履歴の「個々の」項目を更新します。**

戻りコード

- 0**：成功**
- 0**以外：エラーコード**

AmReleaseHandle()

オブジェクトのハンドルとサブハンドルを解放します。

APIシンタックス

long AmReleaseHandle(long hApiObject);

内部Basicシンタックス

Function AmReleaseHandle(hApiObject As Long) As Long

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **hApiObject：処理するオブジェクトのハンドル**

戻りコード

- 0：**成功**
- 0**以外：エラーコード**

AmRemoveCable()

AmRemoveCable API**は、ケーブル（**lCableId**）を現在の場所から取り除きます。ケーブルのステータスは「使用不可能」になります。プロジェクト（**lProjectId**）と作業指示（**lWorkOrderId**）に値が入力されていると、ケーブルがコメント（**strComment**）と共にプロジェクトと作業指示に追加されます。このコメントはケーブルに実行されるアクションを説明します（例：「ケーブルを現在の場所から取り除く」）。**

APIシンタックス

long AmRemoveCable(long hApiCnxBase, long lCableId, long lProjectId, long lWorkOrderId, char *strComment);

内部Basicシンタックス

Function AmRemoveCable(lCableId As Long, lProjectId As Long, lWorkOrderId As Long, strComment As String) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **lCableId：取り除くケーブルの**ID
- **lProjectId：プロジェクトの**ID
- **lWorkOrderId：作業指示の**ID
- **strComment：作業指示に添付されるコメント**

戻りコード

- 0**：成功**
- 0**以外：エラーコード**

AmRemoveDevice()

AmRemoveDevice API**は、デバイス（**lDeviceId**）を現在の場所から取り除きます。デバイスのステータスは「使用不可能」になります。プロジェクト（**lProjectId**）と作業指示（**lWorkOrderId**）に値が入力されていると、デバイスがコメント（**strComment**）と共にプロジェクトと作業指示に追加されます。このコメントはデバイスに実行されるアクションを説明します（例：「デバイスを現在の場所から取り除く」）。**

APIシンタックス

long AmRemoveDevice(long hApiCnxBase, long lDeviceId, long lProjectId, long lWorkOrderId, char *strComment);

内部Basicシンタックス

Function AmRemoveDevice(lDeviceId As Long, lProjectId As Long, lWorkOrderId As Long, strComment As String) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **lDeviceId：取り除くデバイスの**ID
- **lProjectId：プロジェクトの**ID
- **lWorkOrderId：作業指示の**ID
- **strComment：作業指示に添付されるコメント**

戻りコード

- 0**：成功**
- 0**以外：エラーコード**

AmResetPassword()

APIシンタックス

long AmResetPassword(long hApiCnxBase, char *strOldPassword, char *strNewPassword);

内部Basicシンタックス

Function AmResetPassword(strOldPassword As String, strNewPassword As String) As Long

用途

バージョン：4.4.0

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

戻りコード

- **0：成功**
- **0以外：エラーコード**

AmResetUserEnvSession()

APIシンタックス

long AmResetUserEnvSession(long hApiCnxBase, char *strSection);

内部Basicシンタックス

Function AmResetUserEnvSession(strSection As String) As Long

用途

バージョン：4.4.0

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓

	使用
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

戻りコード

- 0：成功
- 0以外：エラーコード

AmResetUserPassword()

APIシンタックス

long AmResetUserPassword(long hApiCnxBase, char *strUser, char *strPasswd, char *strNewPasswd);

内部Basicシンタックス

Function AmResetUserPassword(strUser As String, strPasswd As String, strNewPasswd As String) As Long

用途

バージョン：4.4.0

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生し

たかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

AmRestoreRecord()

アーカイブされたレコードを復元します。

APIシンタックス

long AmRestoreRecord(long hApiRecord);

内部Basicシンタックス

Function AmRestoreRecord(hApiRecord As Long) As Long

用途

バージョン：4.3.0

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiRecord：操作に関連するレコードのハンドル。**

戻りコード

- 0：**成功**
- 0**以外：エラーコード**

注

注意:

リンクされたレコードの処理は、リンクのタイプによって異なります。「所有」タイプのリンクの場合、リンクされたレコードは同一の方法で処理されます。「定義」または「普通」タイプのリンクの場合、リンクされたレコードの外部キーが0にリセットされ、アーカイブされるレコードの識別子と説明文字列がアーカイブフィールドに入力されます。

重要項目:

この関数はアーカイブテーブルからのレコードに対して使用できます。

AmReturnAsset()

この関数では資産を返却できます。

APIシンタックス

long AmReturnAsset(long hApiCnxBase, long lAstId, long lReturnId, long bCanMerge);

内部Basicシンタックス

Function AmReturnAsset(lAstId As Long, lReturnId As Long, bCanMerge As Long) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lAstId：返却する資産の識別子。**
- **lReturnId：返品伝票の識別子。**
- **bCanMerge：このパラメータでは、返品伝票にある既存の明細に返品を統合するかどうかを指定できます。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmReturnContract()

この関数では契約を返却できます。

APIシンタックス

long AmReturnContract(long hApiCnxBase, long lCntrId, long lReturnId, long bCanMerge);

内部Basicシンタックス

Function AmReturnContract(lCntrId As Long, lReturnId As Long, bCanMerge As Long) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lCntrId：返却する契約の識別子。**
- **lReturnId：返品伝票の識別子。**
- **bCanMerge：このパラメータでは、返品伝票にある既存の明細に返品を統合するかどうかを指定できます。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmReturnPortfolioItem()

この関数ではポートフォリオ品目を返却できます。

APIシンタックス

long AmReturnPortfolioItem(long hApiCnxBase, long lPfId, double dQty, long lFromRecptLineId, long lReturnId, long bCanMerge);

内部Basicシンタックス

Function AmReturnPortfolioItem(lPfId As Long, dQty As Double, lFromRecptLineId As Long, lReturnId As Long, bCanMerge As Long) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lPfId：返却するポートフォリオ品目の識別子。**
- **dQty：返却する数量（モデルの単位）**
- **lFromRecptLineId：元の受領明細の識別子。**
- **lReturnId：返却明細の識別子。**
- **bCanMerge：このパラメータでは、返品伝票にある既存の明細に返品を統合するかどうかを指定できます。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmReturnTraining()

この関数では研修を返却できます。

APIシンタックス

long AmReturnTraining(long hApiCnxBase, long lTrainingId, long lReturnId, long bCanMerge);

内部Basicシンタックス

Function AmReturnTraining(lTrainingId As Long, lReturnId As Long, bCanMerge As Long) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lTrainingId：返却する研修の識別子。**
- **lReturnId：返品伝票の識別子。**
- **bCanMerge：このパラメータでは、返品伝票にある既存の明細に返品を統合するかどうかを指定できます。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献** 301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmReturnWorkOrder()

この関数では作業指示を返却できます。

APIシンタックス

long AmReturnWorkOrder(long hApiCnxBase, long lWOId, long lReturnId, long bCanMerge);

内部Basicシンタックス

Function AmReturnWorkOrder(lWOId As Long, lReturnId As Long, bCanMerge As Long) As Long

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lWOId：返却する作業指示の識別子。**
- **lReturnId：返品伝票の識別子。**
- **bCanMerge：このパラメータでは、返品伝票にある既存の明細に返品を統合するかどうかを指定できます。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmRevCryptPassword()

可逆的なパスワードを暗号化します。この関数で暗号化したパスワードを解読する関数は提供されていません。

APIシンタックス

```
long AmRevCryptPassword(long hApiCnxBase, char *return, long lreturn, char *strPassword);
```

内部Basicシンタックス

```
Function AmRevCryptPassword(strPassword As String) As String
```

用途

バージョン：3.5

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strPassword**：暗号化するパスワード。

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError()[献300]関数（必要に応じてAmLastErrorMsg()[献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

AmRgbColor()

strTextパラメータに対応する色のRGB値を返します。

APIシンタックス

long AmRgbColor(char *strText);

内部Basicシンタックス

Function AmRgbColor(strText As String) As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strText：色の名前：**
  - White
  - ltGray
  - Gray
  - Dkgray
  - Black
  - Red
  - Green
  - Blue
  - Yellow
  - Cyan
  - Magenta
  - Dkyellow
  - Dkgreen

- Dkcyan
- Dkblue
- Dkmagenta
- Dkred

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献 301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmRollback()

処理開始（AmStartTransaction関数により実行します）を宣言する前に加えられた変更をすべてキャンセルします。

APIシンタックス

long AmRollback(long hApiCnxBase);

内部Basicシンタックス

Function AmRollback() As Long

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	

	使用
ウィザードの「FINISH.DO」スクリプト	✓

戻りコード

- 0：成功
- 0以外：エラーコード

AmSetFieldDateOnlyValue()

レコードのフィールドを変更します。データベースは更新しません。変更が実行されるのは、レコードが更新または挿入されるか、トランザクションがコミットされたときです。

APIシンタックス

```
long AmSetFieldDateOnlyValue(long hApiRecord, char *strFieldName, long dtptmValue);
```

内部Basicシンタックス

```
Function AmSetFieldDateOnlyValue(hApiRecord As Long, strFieldName As String, dtptmValue As Date) As Long
```

用途

バージョン：4.3.0

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiRecord**：変更するフィールドのハンドル。
- **strFieldName**：変更するフィールドのSQL名。

- **dtptmValue：フィールドの新しい「Date」形式だけの値。AmSetFieldDateValue関数と異なり、日付部分だけが処理され、時刻部分は除去されます。**

戻りコード

- **0：成功**
- **0以外：エラーコード**

AmSetFieldDateValue()

レコード内の1つのフィールドを変更します。データベースは更新しません。変更は、レコードの更新または挿入時、またはトランザクションのコミット時に実行されます。

APIシンタックス

long AmSetFieldDateValue(long hApiRecord, char *strFieldName, long tmValue);

内部Basicシンタックス

Function AmSetFieldDateValue(hApiRecord As Long, strFieldName As String, tmValue As Date) As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiRecord：変更するフィールドを含んでいるレコードのハンドル**
- **strFieldName：変更するフィールドの**SQL名

- **tmValue：フィールドの新しい日付（Date）形式の値**

戻りコード

- **0：成功**
- **0以外：エラーコード**

AmSetFieldDoubleValue()

レコード内の1つのフィールドを変更します。データベースは更新しません。

APIシンタックス

long AmSetFieldDoubleValue(long hApiRecord, char *strFieldName, double dValue);

内部Basicシンタックス

Function AmSetFieldDoubleValue(hApiRecord As Long, strFieldName As String, dValue As Double) As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiRecord：変更するフィールドを含んでいるレコードのハンドル**
- **strFieldName：変更するフィールドのSQL名**
- **dValue：フィールドの新しい倍精度（Double）形式の値**

戻りコード

- 0：成功
- 0以外：エラーコード

AmSetFieldLongValue()

レコード内の1つのフィールドを変更します。データベースは更新しません。日付、時刻、または日付+時刻の値を変更するには、1970年1月1日0時0分0秒から経過した秒数で新しい値を指定します。

APIシンタックス

long AmSetFieldLongValue(long hApiRecord, char *strFieldName, long lValue);

内部Basicシンタックス

Function AmSetFieldLongValue(hApiRecord As Long, strFieldName As String, lValue As Long) As Long

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiRecord：変更するフィールドのハンドル**
- **strFieldName：変更するフィールドの**SQL名
- **lValue：フィールドの新しい値**

戻りコード

- 0：成功

- 0以外：エラーコード

AmSetFieldStrValue()

レコード内の1つのフィールドを変更します。データベースは更新しません。

APIシンタックス

long AmSetFieldStrValue(long hApiRecord, char *strFieldName, char *strValue);

内部Basicシンタックス

Function AmSetFieldStrValue(hApiRecord As Long, strFieldName As String, strValue As String) As Long

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiRecord：変更するフィールドを含んでいるレコードのハンドル**
- **strFieldName：変更するフィールドの**SQL名
- **strValue：フィールドの新しい文字列（**String**）形式の値**

戻りコード

- 0：成功
- 0以外：エラーコード

AmSetLinkFeatureValue()

指定したレコードのリンクタイプの任意管理項目の値を設定します。

APIシンタックス

long AmSetLinkFeatureValue(long hApiRecord, char *strFeatSqlName, char *strDstSelfValue, long lDstId);

内部Basicシンタックス

Function AmSetLinkFeatureValue(hApiRecord As Long, strFeatSqlName As String, strDstSelfValue As String, lDstId As Long) As Long

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiRecord：リンクタイプの任意管理項目に関連付けられているレコードの識別子。**
- **strFeatSqlName：値を設定するリンクタイプの任意管理項目のSQL名。任意管理項目のSQL名には、必ず頭に「fv_」が付きます。**
- **strDstSelfValue：レコードに表示される任意管理項目の値。lDstIdの識別子を持つレコードの「Self」値です。無効または存在しない値を指定した場合は、データベースの整合性が壊れる可能性があります。**
- **lDstId：リンクタイプの任意管理項目のリンク先レコードの識別子。**

戻りコード

- 0：**成功**
- 0**以外：エラーコード**

AmSetProperty()

名前でプロパティを識別し、そのプロパティの値を設定します。さらに、このプロパティの階層構造を更新します。

内部Basicシンタックス

Function AmSetProperty(strVarName As String, vValue As Variant) As Long

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	

パラメータ

- **strVarName：値を設定するプロパティの名前**
- **vValue：プロパティの新しい値**

戻りコード

- 0：**成功**
- 0**以外：エラーコード**

AmSetUserEnvSessionItem()

APIシンタックス

long AmSetUserEnvSessionItem(long hApiCnxBase, char *strSection, char *strEntry, char *strValue);

内部Basicシンタックス

Function AmSetUserEnvSessionItem(strSection As String, strEntry As String, strValue As String) As Long

用途

バージョン：4.4.0

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

戻りコード

- 0：**成功**
- 0**以外：エラーコード**

AmShowCableCrossConnect()

ケーブルのクロスコネクションの画面を表示します。

内部Basicシンタックス

Function AmShowCableCrossConnect(lCableId As Long) As Long

用途

バージョン：4.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **ICableId：演算に関連するケーブルの識別子。**

戻りコード

- 0：成功
- 0以外：エラーコード

AmShowDeviceCrossConnect()

ケーブルデバイスのクロスコネクションの画面を表示します。

内部Basicシンタックス

Function AmShowDeviceCrossConnect(lDeviceId As Long) As Long

用途

バージョン：4.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lDeviceId：演算に関連するケーブルデバイスの識別子。**

戻りコード

- 0：成功
- 0以外：エラーコード

AmSqlTextConst()

文字列を、クエリ内で使用可能になるように変換します。以下の演算た文字列上に実行されます。

- シングルクォーテーションマーク（'）は二重（ダブル）になります。
- シングルクォーテーションマークが文字列の最初と最後に追加されます。

APIシンタックス

long AmSqlTextConst(char *return, long lreturn, char *str);

内部Basicシンタックス

Function AmSqlTextConst(str As String) As String

用途

バージョン：4.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **str：処理する文字列**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

例

```
Dim strReq as String
strReq="SELECT lEmplDeptId FROM amEmplDept WHERE Name=" & amSqlTextConst(strName)
```

変数strName**にはシングルクォーテーションマークが含まれていますが、このクエリは有効です。**

AmStandIn()

日付tmDateにID**がlEmployeeIdの従業員の交替を勤める従業員の識別子を返します。**

APIシンタックス

long AmStandIn(long hApiCnxBase, long lEmployeeId, long tmDate);

内部Basicシンタックス

Function AmStandIn(lEmployeeId As Long, tmDate As Date) As Long

用途

バージョン：4.3.0

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **lEmployeeId：交替従業員を知りたい従業員の識別子。**
- **tmDate：検索の対象となる日付。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献300]関数（必要に応じて**AmLastErrorMsg() **[献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

注

注意:

指定した日付tmDateにIDがlEmployeeIdの従業員がいる場合、関数はその識別子を返します。

従業員がおらず、交替従業員が指定されていない場合、関数は0を返します。

例

```
If [User.Parent.Supervisor] = 0 Then
RetVal = amStandIn([User], amDate())
if RetVal = 0 Then RetVal = [User]
Else
RetVal = amStandIn([User.Parent.Supervisor], amDate())
if RetVal = 0 Then RetVal = [User.Parent.Supervisor]
End If
```

AmStandInGroup()

日付tmDateに識別子がlEmployeeIdの従業員の交替を勤める従業員グループの識別子を返します。

APIシンタックス

long AmStandInGroup(long hApiCnxBase, long lEmployeeId, long tmDate);

内部Basicシンタックス

Function AmStandInGroup(lEmployeeId As Long, tmDate As Date) As Long

用途

バージョン：4.3.0

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **lEmployeeId：交替グループを知りたい従業員の識別子。**
- **tmDate：検索の対象となる日付。**

戻りコード

エラーの場合は、次の2**つの処理方法を用意します。**

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

注

注意:

指定した日付tmDateに識別子がlEmployeeIdの従業員がいる場合、関数は0**を返します。**

従業員がおらず、交替グループが指定されていない場合、関数は0**を返します。**

AmStartTransaction()

接続に関連付けられているデータベースで新しい処理を開始します。次のCommitまたはRollbackステートメントによって、データベースに加えられたすべての変更を有効にするかキャンセルするかを指定します。

APIシンタックス

long AmStartTransaction(long hApiCnxBase);

内部Basicシンタックス

Function AmStartTransaction() As Long

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

戻りコード

- 0：**成功**
- 0**以外：エラーコード**

AmStartup()

AssetCenter**ライブラリへの要求を初期化します。この関数は、他のどの関数よりも先に適用する必要があります。**

APIシンタックス

void AmStartup();

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	

AmTableDesc()

テーブルのSQL名から、<テーブルの> （<テーブルのSQL名>）形式の説明文字列を作成します。

APIシンタックス

long AmTableDesc(long hApiCnxBase, char *return, long lreturn, char *strSqlName);

内部Basicシンタックス

Function AmTableDesc(strSqlName As String) As String

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strSqlName：説明文字列を取得するテーブルのSQL名。このパラメータに無効なSQL名が含まれていると、疑問符（？）を返します。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献 301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

資産テーブル（SQL**名：**amAsset**）の説明文字列を作成します。**

```
AmTableDesc("amAsset")
```

次の説明文字列を返します。

```
Assets (amAsset)
```

AmTaxRate()

税金のタイプ、税区分、および日付に応じて、税率を計算します。

APIシンタックス

double AmTaxRate(long hApiCnxBase, char *strTaxRateName, long lTaxLocId, long tmDate, double dValue);

内部Basicシンタックス

Function AmTaxRate(strTaxRateName As String, lTaxLocId As Long, tmDate As Date, dValue As Double) As Double

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓

使用	
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strTaxRateName：税率の計算に使用する税金の種類のSQL名**
- **lTaxLocId：税金のタイプに対応する税区分のID番号**
- **tmDate：税率を計算する日付**
- **dValue：このパラメータは現在使用されておらず、互換性の理由により保持されています。お好きな値に設定してください。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

AmUpdateDetail()

この関数はデータ入力ウィザードに使用します。ウィザードのコンテキスト（ウィザードを使って更新またはデータを入力するレコードのテーブル）をこの関数で明確に指定します。値に応じて、テーブルのフィールドやリンクを更新、またはデータを入力します。この関数は非モーダルのウィザードでは使用不可能です。

内部Basicシンタックス

Function AmUpdateDetail(strFieldName As String, varValue As Variant) As Long

用途

バージョン：3.00

使用
AssetCenter API

使用	
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strFieldName：更新する任意管理項目のSQL名**
- **varValue：フィールドに入力する新しい値**

戻りコード

- 0：**成功**
- 0**以外：エラーコード**

AmUpdateLossLines()

lLossValIdの識別子で参照される損失額規則を使用している契約のすべての損失額を更新します。

内部Basicシンタックス

Function AmUpdateLossLines(lLossValId As Long) As Long

用途

バージョン：4.3.0

使用	
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lLossValId：損失額規則の識別子。**

戻りコード

- 0：成功
- 0以外：エラーコード

AmUpdateRecord()

レコードを更新します。

APIシンタックス

long AmUpdateRecord(long hApiRecord);

内部Basicシンタックス

Function AmUpdateRecord(hApiRecord As Long) As Long

用途

バージョン：2.52

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **hApiRecord：更新するフィールドを含んでいるレコードのハンドル**

戻りコード

- 0：成功
- 0以外：エラーコード

AmUpdateUser()

この情報は、データベース中の従業員に関する情報（ドメイン、NTユーザ名、説明）を更新します。

APIシンタックス

long AmUpdateUser(long hApiCnxBase, long lId, char *strNTUserName, char *strNTDomain, char *strNTUserDesc);

内部Basicシンタックス

Function AmUpdateUser(lId As Long, strNTUserName As String, strNTDomain As String, strNTUserDesc As String) As Long

用途

バージョン：4.3.0

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lId：データベース中の関連するユーザの識別子。**
- **strNTUserName：従業員のNTユーザ名。**
- **strNTDomain：従業員のNTドメイン名。**
- **strNTUserDesc：NTユーザに関連付けられた説明。**

戻りコード

- **0：成功**
- **0以外：エラーコード**

AmValueOf()

この関数はウィザードで使います。strVarNameパラメータで特定したプロパティを返します。

内部Basicシンタックス

Function AmValueOf(strVarName As String) As Variant

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **strVarName：値を取得するプロパティの名前**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

次の例では、「Page1.Label**」プロパティの値が戻ります。**

```
AmValueOf("Page1.Label")
```

この関数を使うと、処理したプロパティの従属文字列が壊れる可能性がありますので、注意してください。

AmWizChain()

ウィザードAに含まれているウィザードBを実行します。ウィザードBが終了すると、ウィザードAに戻ります。

内部Basicシンタックス

Function AmWizChain(strWizSqlName As String) As Long

用途

バージョン：3.00

使用	
AssetCenter API	
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strWizSqlName：実行するウィザードの**SQL名

戻りコード

- 0：成功
- 0以外：エラーコード

AmWorkTimeSpanBetween()

2つの日付（開始日と終了日）で指定する作業期間の時間を秒単位で返します。計算には業務用カレンダの情報が使われます。

APIシンタックス

long AmWorkTimeSpanBetween(long hApiCnxBase, char *strCalendarSqlName, long tmEnd, long tmStart);

内部Basicシンタックス

Function AmWorkTimeSpanBetween(strCalendarSqlName As String, tmEnd As Date, tmStart As Date) As Date

用途

バージョン：3.00

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **strCalendarSqlName：作業期間の時間の計算に使用する業務用カレンダの**SQL**名。このパラメータを省略すると、計算には営業時間が反映されません。**
- **tmEnd：計算する作業期間の終了日**
- **tmStart：計算する作業期間の開始日**

戻りコード

エラーの場合は、次の2**つの処理方法を用意します。**

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

1998年９月1日午前８時から1998年９月24日午後７時までの作業時間を計算します。計算に使用する「Calendar_Paris」というSQL名のカレンダには次の営業時間が設定されています。

- 月曜日 - 木曜日：午前８時 - 正午、午後2時 - 午後６時
- 金曜日：午前８時 - 正午、午後2時 - 午後５時。

```
AmWorkTimeSpanBetween("Calendar_Paris", "1998/09/24 19:00:00", "1998/09/01 08:00:00")
```

この例では、507,600という値が戻ります。この値は2つの日付の間の作業時間を秒で表したものです。

AppendOperand()

関数に渡されるパラメータに応じて文字列を連結します。文字列は次のように連結されます。

```
strExpr strOperator strOperand
```

内部Basicシンタックス

Function AppendOperand(strExpr As String, strOperator As String, strOperand As String) As String

用途

バージョン：3.5

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strExpr**：連結される文字列式
- **strOperator**：連結する演算子
- **strOperand**：連結するオペランド

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

注

注意:

strExprまたはstrOperandパラメータのいずれかを省略すると、strOperatorは使用されません。

ApplyNewVals()

［ListBox**］コントロール内の同じセルに同じ値を割り当てます。**

内部Basicシンタックス

Function ApplyNewVals(strValues As String, strNewVals As String, strRows As String, strRowFormat As String) As String

用途

バージョン：3.5

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strValues：処理する［ListBox］コントロールの値が含まれている文字列**
- **strNewVals：セルに割り当てる新しい値**
- **strRows：処理する行のID。各IDをコンマ（,）で区切ります。**
- **strRowFormat：サブリストの書式化命令。各命令をパイプ文字（|）で区切ります。個々の命令には、strNewValsパラメータを含んでいる列の番号を指定します。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError()［献300］関数（必要に応じてAmLastErrorMsg()［献301］関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

Asc()

文字列の最初の文字のASCIIコード（数値）を返します。

内部Basicシンタックス

Function Asc(strAsc As String) As Long

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strAsc：処理する文字列**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim iCount as Integer
Dim strString as String
For iCount=Asc("A") To Asc("Z")
strString = strString & Str(iCount)
Next iCount
RetVal=strString
```

Atn()

数値のアークタンジェントを返します。単位はラジアンです。

内部Basicシンタックス

Function Atn(dValue As Double) As Double

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **dValue：アークタンジェントを取得する数値**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim dPi as Double
Dim strString as String
dPi=4*Atn(1)
strString = Str(dPi)
RetVal=strString
```

BasicToLocalDate()

BASIC**形式の日付を文字列の日付（**Windows**のコントロールパネルに表示される形式）に変換します。**

内部Basicシンタックス

Function BasicToLocalDate(strDateBasic As String) As String

用途

バージョン：3.5

使用	
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strDateBasic：変換するBASIC形式の日付**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

BasicToLocalTime()

BASIC形式の時刻を文字列の時刻（Windowsのコントロールパネルに表示される形式）に変換します。

内部Basicシンタックス

Function BasicToLocalTime(strTimeBasic As String) As String

用途

バージョン：3.5

使用
AssetCenter API

	使用
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strTimeBasic：変換するBASIC形式の時刻**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

BasicToLocalTimeStamp()

BASIC形式の日付+時刻を文字列の日付+時刻（Windowsのコントロールパネルに表示される形式）に変換します。

内部Basicシンタックス

Function BasicToLocalTimeStamp(strTSBasic As String) As String

用途

バージョン：3.5

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strTSBasic：変換するBASIC形式の日付+時刻**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

Beep()

コンピュータで警告音を鳴らします。

内部Basicシンタックス

Function Beep()

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

CDbl()

式を倍精度型（Double型）に変換します。

内部Basicシンタックス

Function CDbl(dValue As Double) As Double

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **dValue**：変換する式

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

例

```
Dim dNumber As Double
Dim iInteger as Integer
iInteger = 25
dNumber=CDbl(iInteger)
RetVal=dNumber
```

ChDir()

現在のディレクトリを変更します。

内部Basicシンタックス

Function ChDir(strDirectory As String)

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strDirectory：新しいディレクトリ**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

ChDrive()

現在のドライブを変更します。

内部Basicシンタックス

Function ChDrive(strDrive As String)

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **strDrive：新しいドライブ名**

戻りコード

エラーの場合は、次の2**つの処理方法を用意します。**

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

Chr()

iChrパラメータから渡されたASCII**コードに対応する文字列を返します。**

内部Basicシンタックス

Function Chr(IChr As Long) As String

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **IChr：文字の**ASCII**コード**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim iCount as Integer
Dim iIteration as Integer
Dim strMessage as String
Dim strLF as String
strLF=Chr(10)
For iIteration=1 To 2
For iCount=Asc("A") To Asc("Z")
strMessage=strMessage+Chr(iCount)
Next iCount
strMessage=strMessage+strLF
```

```
Next iIteration
RetVal=strMessage
```

CInt()

指定した式を整数型に変換します。

内部Basicシンタックス

Function CInt(iValue As Long) As Long

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **iValue：変換する式**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献** 300**]関数（必要に応じて**AmLastErrorMsg() **[献**301**]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim iNumber As Integer
Dim dDouble as Double
dDouble = 25.24589
iNumber=CInt(dDouble)
RetVal=iNumber
```

CLng()

指定した式を倍長整数型に変換します。

内部Basicシンタックス

Function CLng(lValue As Long) As Long

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **lValue：変換する式**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim lNumber As Long
Dim iInteger as Integer
iInteger = 25
lNumber=CLng(iInteger)
RetVal=lNumber
```

Cos()

数値のコサインを返します。単位はラジアンです。

内部Basicシンタックス

Function Cos(dValue As Double) As Double

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **dValue：コサインを取得する数値**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献** 300**]関数（必要に応じて**AmLastErrorMsg() **[献**301**]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

注

注意:

度からラジアンへの変換公式は次になります:

ラジアンでの角度 = (**度での角度**) * Pi / 180

例

```
Dim dCalc as Double
dCalc=Cos(2.79)
RetVal=dCalc
```

CountOccurences()

1つの文字列内に特定の文字列が何個含まれているかをカウントします。

内部Basicシンタックス

Function CountOccurences(strSearched As String, strPattern As String, strEscChar As String) As Long

用途

バージョン：3.5

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strSearched：処理する文字列**
- **strPattern：strSearchedパラメータ内で検索する文字列**

- **strEscChar：エスケープ文字。関数がstrSearched文字列内でこの文字を検出すると、検索を中止します。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim MyStr
MyStr=CountOccurences("you|me|you,me|you", "you", ",") :' "2"を返します。
MyStr=CountOccurences("you|me|you,me|you", "you", "|") :' "1"を返します。
```

CountValues()

1つの文字列に含まれる要素数をカウントします。区切り文字やエスケープ文字を区別します。

内部Basicシンタックス

Function CountValues(strSearched As String, strSeparator As String, strEscChar As String) As Long

用途

バージョン：3.5

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strSearched：処理する文字列**
- **strSeparator：要素を区切る区切り文字**
- **strEscChar：エスケープ文字。この文字が区切り文字の前に付くと、その区切り文字は無視されます。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim MyStr
MyStr=CountValues("you|me|you\|me|you", "|", "\") : ' 4を返します。
MyStr=CountValues("you|me|you\|me|you", "|", "") : ' 5を返します。
```

CSng()

指定した式を浮動小数点数型（Float型）に変換します。

内部Basicシンタックス

Function CSng(fValue As Single) As Single

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **fValue：変換する式**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim dNumber As Double
Dim iInteger as Integer
iInteger = 25
dNumber=CSng(iInteger)
RetVal=dNumber
```

CStr()

指定した式を文字列型に変換します。

内部Basicシンタックス

Function CStr(strValue As String) As String

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strValue：変換する式**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

例

```
Dim dNumber As Double
Dim strMessage as String
dNumber = 2,452873
strMessage=CStr(dNumber)
RetVal=strMessage
```

CurDir()

現在のパスを返します。

内部Basicシンタックス

Function CurDir() As String

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

CVar()

指定した式を可変型（Variant型）に変換します。

内部Basicシンタックス

Function CVar(vValue As Variant) As Variant

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **vValue：変換する式**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

Date()

現在のシステムの日付を返します。

内部Basicシンタックス

Function Date() As Date

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

DateAdd()

開始日に実際の経過時間を追加して、新たな日付を計算します。

内部Basicシンタックス

Function DateAdd(tmStart As Date, tsDuration As Long) As Date

用途

バージョン：2.51

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **tmStart：経過時間の追加先の日付**
- **tsDuration：tmStartの日付に追加する時間（秒単位）**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献 301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

DateAddLogical()

開始日に論理上の経過時間を追加して、新たな日付を計算します（1ヶ月を30日として計算します）。

内部Basicシンタックス

Function DateAddLogical(tmStart As Date, tsDuration As Long) As Date

用途

バージョン：2.51

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **tmStart：経過時間の追加先の日付**
- **tsDuration：tmStartの日付に追加する時間（秒単位）**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

DateDiff()

指定した2つの日付の間の時間を秒単位で計算します。

内部Basicシンタックス

Function DateDiff(tmEnd As Date, tmStart As Date) As Date

用途

バージョン：2.51

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **tmEnd：計算する期間の終了日**
- **tmStart：計算する期間の開始日**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

次の例では、1998**年**1**月**1**日から**1999**年**1**月**1**日までの間に経過する時間を計算します。**

```
AmDateDiff("1998/01/01 00:00:00", "1999/01/01 00:00:00")
```

DateSerial()

iYear、iMonth、およびiDayパラメータで指定した形式で日付型の値を返します。

内部Basicシンタックス

Function DateSerial(iYear As Long, iMonth As Long, iDay As Long) As Date

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **iYear：西暦。**0 - 99**の間の場合は、**1900**年 - **1999**年までの年を表します。その他の年に関しては、４桁（**1800**など）で指定する必要があります。**
- **iMonth：月**
- **iDay：日**

戻りコード

エラーの場合は、次の2**つの処理方法を用意します。**

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

各パラメータに、それぞれ日、月、または年を表す数式を利用できます。

```
DateSerial(1999-10, 3-2, 15-8)
```

例えば、上記の例は次の値を返します。

```
1989/1/7
```

パラメータの値が予想される範囲（日付ならば1 - 31**、月ならば**1 - 12**など）以外の値の場合、関数は空の日付を返します。**

DateValue()

「日付＋時刻」の値の日付の部分を返します。

内部Basicシンタックス

Function DateValue(tmDate As Date) As Date

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **tmDate：「日付＋時刻」形式の日付**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献** 300**]関数（必要に応じて**AmLastErrorMsg() **[献**301**]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

次の例は、

```
DateValue ("1999/09/24 15:00:00")
```

次の値を返します。

```
1999/09/24
```

Day()

tmDateパラメータの日付を返します。

内部Basicシンタックス

Function Day(tmDate As Date) As Long

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **tmDate：処理する「時刻+日付」形式のパラメータ**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim strDay as String
strDay=Day(Date())
RetVal=strDay
```

EnumToComboBox()

不特定のリストデータの項目を整理し、ウィザードのリストコントロールと互換性のある形式にします。これにより、不特定のリストデータの値をウィザードのドロップダウンリストに表示できます。

内部Basicシンタックス

Function EnumToComboBox(strFormat As String) As String

用途

バージョン：4.3.0

使用	
AssetCenter API	
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	
ウィザードスクリプト	
ウィザードの「FINISH.DO**」スクリプト**	

パラメータ

- **strFormat：システムリストデータのエントリのリスト。AmDbGetList()関数の実行結果をこのパラメータに指定するのがよい方法です。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献 301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

次の例は、不特定のリストデータamWOPriority**の値をウィザードのリストコントロールと互換性のある形式に整理します。**

```
Dim strValues As String
strValues = AmDbGetList("SELECT Value FROM amItemListVal WHERE ItemizedList.Iden
tifier = 'amWOPriority'", "", ",", "")
RetVal = EnumToComboBox(strValues)
```

EscapeSeparators()

区切り文字の前にエスケープ文字を付けます。

内部Basicシンタックス

Function EscapeSeparators(strSource As String, strSeparators As String, strEscChar As String) As String

用途

バージョン：3.5

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strSource：処理する文字列**
- **strSeparators：エスケープ文字を付ける区切り文字。複数の区切り文字を宣言する場合は、strEscCharパラメータで指定するエスケープ文字で区切る必要があります。**
- **strEscChar：エスケープ文字。strSeparatorsパラメータのすべての区切り文字の前にこのエスケープ文字が付けられます。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**

- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献 301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim MyStr
MyStr=EscapeSeparators("you|me|you,me|you", "|\,", "\") :' "you\|me\|you\,me\|you"を返します。
```

ExeDir()

この関数は実行可能ファイルのフルパスを返します。

内部Basicシンタックス

Function ExeDir() As String

用途

バージョン：3.60

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献 301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim strPath as string
strPath=ExeDir()
```

Exp()

数値のべき乗を返します。

内部Basicシンタックス

Function Exp(dValue As Double) As Double

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **dValue：べき乗を取得する数値。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim iSeed as Integer
iSeed = Int((10*Rnd)-5)
RetVal = Exp(iSeed)
```

ExtractValue()

1つの文字列内で、区切り文字が前に付いていない値を取り出します。取り出した値は、その文字列から削除されます。この関数は、エスケープ文字も区別します。値を抽出する文字列に区切り文字が含まれていない場合は、その文字列全体が抽出され、元の文字列は完全に削除されます。

内部Basicシンタックス

Function ExtractValue(pstrData As String, strSeparator As String, strEscChar As String) As String

用途

バージョン：3.5

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **pstrData：処理する文字列**
- **strSeparator：処理する文字列の区切り文字**
- **strEscChar：エスケープ文字。この文字が区切り文字の前に付くと、その区切り文字は無視されます。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

例

```
Dim MyStr
MyStr=ExtractValue("you,me", ",", "\") :' "you"を返し、"me"を文字列に残します。
MyStr=ExtractValue(",you,me", ",", "\") :' ""を返し、"you,me"を文字列に残します。
MyStr=ExtractValue("you", ",", "\") :' "you"を返し、""を文字列に残します。
MyStr=ExtractValue("you\,me", ",", "\") :' "you\,me"を返し、""を文字列に残します。
MyStr=ExtractValue("you\,me", ",", "") :' "you\"を返し、"me"を文字列に残します。
RetVal=""
```

FileCopy()

ファイルまたはフォルダをコピーします。

内部Basicシンタックス

Function FileCopy(strSource As String, strDest As String) As Long

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strSource：コピーするファイルまたはディレクトリのフルパス**

- **strDest：コピー先のファイルまたはディレクトリのフルパス**

戻りコード

- 0：成功
- 0以外：エラーコード

FileDateTime()

ファイルの時刻と日付を倍長整数型で返します。

内部Basicシンタックス

Function FileDateTime(strFileName As String) As Date

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strFileName：処理するファイルのフルパス**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献 301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

FileExists()

ファイルが存在するかどうかテストします。

- 0：ファイルが見つからない
- 1ファイルが見つかった

内部Basicシンタックス

Function FileExists(strFileName As String) As Long

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strFileName**：存在するかテストするファイルの完全パス

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

例

```
If FileExists("c:\tmp\myfile.log") Then
strFileName = "c:\archive\" + FormatDate(Date, "dddd d mmm yyyy") + ".log"
```

```
FileCopy("c:\tmp\myfile.log", strFileName)
End if
```

FileLen()

ファイルのサイズを返します。

内部Basicシンタックス

Function FileLen(strFileName As String) As Long

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **strFileName：処理するファイルのフルパス**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献 301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

Fix()

数値の整数部分を返します（負数の場合は、その値の正の方向の一番近い整数を返します）。

内部Basicシンタックス

Function Fix(dValue As Double) As Long

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **dValue：整数部分を取得する数値**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim dSeed as Double
dSeed = (10*Rnd)-5
RetVal = Fix(dSeed)
```

FormatDate()

strFormatパラメータの式に従って日付を書式化します。

内部Basicシンタックス

Function FormatDate(tmFormat As Date, strFormat As String) As String

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **tmFormat：書式化する日付**
- **strFormat：書式化命令が含まれている式**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

次の例は、日付を書式化するコードの例です。

```
Dim MyDate
MyDate="2000/03/14"
```

RetVal=FormatDate(MyDate, "dddd d mmmm yyyy") :' "Tuesday 14 March 2000"を返します。

FormatResString()

文字列内の変数$1、$2、$3、$4、**および** $5**を、strParamOne、strParamTwo、strParamThree、strParamFour、およびstrParamFiveパラメータに渡された文字列にそれぞれ置き換えます。**

内部Basicシンタックス

Function FormatResString(strResString As String, strParamOne As String, strParamTwo As String, strParamThree As String, strParamFour As String, strParamFive As String) As String

用途

バージョン：3.5

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **strResString：処理する文字列**
- **strParamOne：変数**$1**を置き換える文字列**
- **strParamTwo：変数**$2**を置き換える文字列**
- **strParamThree：変数**$3**を置き換える文字列**
- **strParamFour：変数**$4**を置き換える文字列**
- **strParamFive：変数**$5**を置き換える文字列**

戻りコード

エラーの場合は、次の2**つの処理方法を用意します。**

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

例

次の例は、

```
FormatResString("I$1he$2you$3", "you", "we", "they")
```

"IyouheweyouthEy"を返します。

FV()

定額の定期的な支払、および固定利率に基づいて計算した、実際の年間支払金額を返します。

内部Basicシンタックス

Function FV(dblRate As Double, iNper As Long, dblPmt As Double, dblPV As Double, iType As Long) As Double

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **dblRate**：支払日ごとの利率。例えば、年利が6%のローンの毎月の支払日の利率は、次のようになります。

0.06/12=0.005 または 0.5%

- **iNper：総支払回数**
- **dblPmt：1回の支払金額。支払金額には一般的に元金と利子が含まれます。**
- **dblPV：実際に支払わなければならない金額（総額）**
- **iType：支払期限を示します。次のいずれかの値になります。**
  - **0：支払が後払い（期間内の最後）の場合**
  - **1：支払が先払い（期間内の初め）の場合**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

注

注意:

- **RateとNperパラメータの計算には同じ単位の支払額を使用する必要があります。**
- **支払う金額（Pmtパラメータの金額）はマイナスの数値、受け取る総額はプラスの数値で表わされます。**

GetEnvVar()

環境変数の値を返します。環境変数が存在しない場合は、空の値が返されます。

内部Basicシンタックス

Function GetEnvVar(strVar As String, bExpand As Long) As String

用途

バージョン：3.2.0

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strVar：環境変数の名前**
- **bExpand：このブール型パラメータは、環境変数が1つまたは複数の別の環境変数を参照する場合にのみ有用です。この際、パラメータの値が1であると（デフォルト値）、参照される各変数は値に変換されます。それ以外の場合は変数はそのままで変換されません。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
RetVal = getEnvVar("PROMPT")
```

GetListItem()

区切り文字で区切られている文字列のINb番目の文字列を返します。

内部Basicシンタックス

Function GetListItem(strFrom As String, strSep As String, lNb As Long, strEscChar As String) As String

用途

バージョン：3.5

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strFrom：処理する文字列**
- **strSep：処理する文字列の区切り文字**
- **INb：取得する文字列の位置**
- **strEscChar：エスケープ文字。この文字が区切り文字の前に付くと、その区切り文字は無視されます。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

次に2つの例を示します。

```
GetListItem("this_is_a_test", "_", 2, "%")
```

"is"**を返します。**

```
GetListItem("this%_is_a_test", "_", 2, "%")
```

"a"**を返します。**

Hex()

10進法のパラメータの16進法の値を返します。

内部Basicシンタックス

Function Hex(dValue As Double) As String

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **dValue：**16進法の値を取得する10進法の数値

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献 301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

Hour()

tmTimeパラメータの時間の部分を返します。

内部Basicシンタックス

Function Hour(tmTime As Date) As Long

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **tmTime：処理する「時刻+日付」形式のパラメータ**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim strHour as String
strHour=Hour(Date())
RetVal=strHour
```

InStr()

文字列内で、検索する文字列が最初に見つかった文字の位置を返します。

内部Basicシンタックス

Function InStr(lPosition As Long, strSource As String, strPattern As String, bCaseSensitive As Long) As Long

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **lPosition**: **検索の開始位置。このパラメータは必須であり、**65,535**以下の有効な自然数である必要があります。**
- **strSource：処理する文字列**
- **strPattern：検索する文字列**
- **bCaseSensitive：このパラメータを使って、大文字と小文字を区別する（**=1**）、または区別しない（**=0**）かを指定します。**

戻りコード

エラーの場合は、次の2**つの処理方法を用意します。**

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

注

注意:

最初の一致する文字列の位置は必ず1**です。検索対象の文字列が見つからない場合、関数は**0**を戻します。**

例

```
Dim strSource as String
Dim strToSearch as String
Dim iPosition
strSource = "Good Bye"
strToSearch = "Bye"
iPosition = Instr(2, strSource, strToSearch)
RetVal=iPosition
```

Int()

数値の整数部分を返します（負数の場合は、その値の負の方向の一番近い整数を返します）。

内部Basicシンタックス

Function Int(dValue As Double) As Long

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **dValue：整数部分を取得する数値**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**

- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献 301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim iSeed as Integer
iSeed = Int((10*Rnd)-5)
RetVal = Abs(iSeed)
```

IPMT()

年間支払金額のうち、指定した支払日の利子の金額を返します。

内部Basicシンタックス

Function IPMT(dblRate As Double, iPer As Long, iNper As Long, dblPV As Double, dblFV As Double, iType As Long) As Double

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **dblRate：支払日ごとの利率。例えば、年利が**6%**のローンの毎月の支払日の利率は、次のようになります。**

```
0.06/12=0.005 または 0.5%
```

- **iPer：計算する期間。**1**から****Nper****の数値の間で指定します。**
- **iNper：総支払回数**

- **dblPV**：実際に支払わなければならない金額（総額）
- **dblFV**：支払が終了した時の差引残高または将来の支払用に確保する金額。一般に、特にローンを返済する場合は、このパラメータを「0」に設定します。実際、すべての支払を済ませるとローンの値はゼロになります。
- **iType**：支払期限を示します。次のいずれかの値になります。
- **0**：支払が後払い（期間内の最後）の場合
- **1**：支払が先払い（期間内の初め）の場合

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

注

注意:

- RateとNperパラメータの計算には同じ単位の支払額を使用する必要があります。
- 支払う金額（Pmtパラメータの金額）はマイナスの数値、受け取る総額はプラスの数値で表わされます。

IsNumeric()

文字列に数値が含まれるかどうかを識別します。

内部Basicシンタックス

Function IsNumeric(strString As String) As Long

用途

バージョン：3.5

使用	
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strString：解析する文字列**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

Kill()

ファイルを削除します。

内部Basicシンタックス

Function Kill(strKilledFile As String) As Long

用途

バージョン：3.00

使用	
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strKilledFile：処理するファイルのフルパス**

戻りコード

- 0：成功
- 0以外：エラーコード

LCase()

文字列パラメータに含まれるすべての文字を小文字に変換して返します。

内部Basicシンタックス

Function LCase(strString As String) As String

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strString：小文字に変換する文字列**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

例

```
' This example uses the LTrim and RTrim functions to strip leading ' and trailing spaces,
respectively, from a string variable.
' It uses the Trim function alone to strip both types of spaces.
' LCase and UCase are also shown in this example as well as the use
' of nested function calls

Dim strString as String
Dim strTrimString as String
strString = "  <-Trim->  " :' Initialize string.
strTrimString = LTrim(strString) :' strTrimString = "<-Trim->  ".
strTrimString = LCase(RTrim(strString)) :' strTrimString = "  <-trim->".
strTrimString = LTrim(RTrim(strString)) :' strTrimString = "<-Trim->".
' Using the Trim function alone achieves the same result.
strTrimString = UCase(Trim(strString)) :' strTrimString = "<-TRIM->".
RetVal= "|" & strTrimString & "|"
```

Left()

文字列の左端から、iNumber**で指定した数の文字を返します。**

内部Basicシンタックス

Function Left(strString As String, lNumber As Long) As String

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓

	使用
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strString：処理する文字列**
- **lNumber: 戻される文字数。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim lWord, strMsg, rWord, iPos :' Declare variables.
strMsg = "Left() Test."
iPos = InStr(1, strMsg, " ") : ' スペースを探す
lWord = Left(strMsg, iPos - 1) :' 左側の単語を取得
rWord = Right(strMsg, Len(strMsg) - iPos) :' 右側の単語を取得
strMsg=rWord+lWord :' 2つの単語を交換
RetVal=strMsg
```

LeftPart()

strSepパラメータに指定されている区切り文字の左側の文字列を1つ取得します。
左から右に向かって区切り文字を探します。
bCaseSensitiveパラメータを使って、大文字と小文字を区別することもできます。

内部Basicシンタックス

Function LeftPart(strFrom As String, strSep As String, bCaseSensitive As Long) As String

用途

バージョン：3.5

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **strFrom：処理する文字列**
- **strSep：処理する文字列の区切り文字**
- **bCaseSensitive：このパラメータを使って、大文字と小文字を区別する（**=1**）、または区別しない（**=0**）かを指定します。**

戻りコード

エラーの場合は、次の2**つの処理方法を用意します。**

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献** 301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

文字列"This_is_a_test"**に、LeftPart、LeftPartFromRight、RightPart、および RightPartFromLeft関数を使った例を示します。**

```
LeftPart("This_is_a_test","_",0)
```

"This"**を返します。**

```
LeftPartFromRight("This_is_a_test","_",0)
```

"This_is_a"を返します。

```
RightPart("This_is_a_test","_",0)
```

"test"を返します。

```
RightPartFromLeft("This_is_a_test","_",0)
```

"is_a_test"を返します。

LeftPartFromRight()

strSepパラメータに指定されている区切り文字の左側にある文字列を1つ取得します。
右から左に向かって区切り文字を探します。
bCaseSensitiveパラメータを使って、大文字と小文字を区別することもできます。

内部Basicシンタックス

Function LeftPartFromRight(strFrom As String, strSep As String, bCaseSensitive As Long) As String

用途

バージョン：3.5

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strFrom：処理する文字列**
- **strSep：処理する文字列の区切り文字**
- **bCaseSensitive：このパラメータを使って、大文字と小文字を区別する（=1）、または区別しない（=0）かを指定します。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献** 300**]関数（必要に応じて**AmLastErrorMsg() **[献**301**]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

文字列"This_is_a_test"**に、LeftPart、LeftPartFromRight、RightPart、および RightPartFromLeft関数を使った例を示します。**

```
LeftPart("This_is_a_test","_",0)
```

"This"**を返します。**

```
LeftPartFromRight("This_is_a_test","_",0)
```

"This_is_a"**を返します。**

```
RightPart("This_is_a_test","_",0)
```

"test"**を返します。**

```
RightPartFromLeft("This_is_a_test","_",0)
```

"is_a_test"**を返します。**

Len()

文字列または可変型データに含まれる文字数を返します。

内部Basicシンタックス

Function Len(vValue As Variant) As Long

用途

バージョン：3.00

使用
AssetCenter API

	使用
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **vValue：処理する可変型データ**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim strTest as String
Dim iLength as Integer
strTest = "Peregrine Systems"
iLength = Len(strTest) : ' iLengthの値は17
RetVal=iLength
```

LocalToBasicDate()

文字列形式の日付（Windows**のコントロールパネルに表示される形式）を**BASIC**形式の日付に変換します。**

内部Basicシンタックス

Function LocalToBasicDate(strDateLocal As String) As String

用途

バージョン：3.5

使用	
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strDateLocal：変換する文字列形式の日付**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献 301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

LocalToBasicTime()

文字列形式の時刻（Windowsのコントロールパネルに表示される形式）をBASIC**形式の時刻に変換します。**

内部Basicシンタックス

Function LocalToBasicTime(strTimeLocal As String) As String

用途

バージョン：3.5

使用
AssetCenter API

	使用
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strTimeLocal：変換する文字列形式の時刻**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

LocalToBasicTimeStamp()

文字列形式の日付+時刻（Windowsのコントロールパネルに表示される形式）をBASIC形式の日付+時刻に変換します。

内部Basicシンタックス

Function LocalToBasicTimeStamp(strTSLocal As String) As String

用途

バージョン：3.5

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strTSLocal：変換する文字列形式の日付+時刻**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

LocalToUTCDate()

「日付＋時刻」フォーマットの日付をUTCフォーマット（タイムゾーンに関係しない） へ変換します。

内部Basicシンタックス

Function LocalToUTCDate(tmLocal As Date) As Date

用途

バージョン：3.5

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **tmLocal：「日付＋時刻」形式の日付**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

Log()

数値の自然対数を返します。

内部Basicシンタックス

Function Log(dValue As Double) As Double

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **dValue：対数を取得する数値**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献300]関数（必要に応じて**AmLastErrorMsg() **[献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim dSeed as Double
dSeed = Int((10*Rnd)-5)
RetVal = Log(dSeed)
```

LTrim()

文字列の先頭のスペースをすべて取り除きます。

内部Basicシンタックス

Function LTrim(strString As String) As String

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **strString：処理する文字列**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
' This example uses the LTrim and RTrim functions to strip leading ' and trailing spaces,
respectively, from a string variable.
' It uses the Trim function alone to strip both types of spaces.
' LCase and UCase are also shown in this example as well as the use
' of nested function calls

Dim strString as String
Dim strTrimString as String
strString = "  <-Trim->  " :' Initialize string.
strTrimString = LTrim(strString) :' strTrimString = "<-Trim->  ".
strTrimString = LCase(RTrim(strString)) :' strTrimString = "  <-trim->".
strTrimString = LTrim(RTrim(strString)) :' strTrimString = "<-Trim->".
' Using the Trim function alone achieves the same result.
strTrimString = UCase(Trim(strString)) :' strTrimString = "<-TRIM->".
RetVal= "|" & strTrimString & "|"
```

MakeInvertBool()

ブール値の逆の値を返します（0は1、他の値はすべて0になります）。

内部Basicシンタックス

Function MakeInvertBool(lValue As Long) As Long

用途

バージョン：3.5

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lValue：処理する数値**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim MyValue
MyValue=MakeInvertBool(0) : ' 1を返します。
MyValue=MakeInvertBool(1) : ' 0を返します。
MyValue=MakeInvertBool(254) : ' 0を返します。
```

Mid()

文字列内の一部分の文字列を返します。

内部Basicシンタックス

Function Mid(strString As String, lStart As Long, lLen As Long) As String

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strString：処理する文字列**
- **lStart**: strString**内から展開する文字列の開始位置。**
- **lLen**: **展開する文字列の長さ。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim strTest as String
strTest="One Two Three" : ' テスト文字列を定義
strTest=Mid(strTest,5,3) :' strTest="Two"
RetVal=strTest
```

Minute()

tmTimeパラメータに含まれている時刻の分の部分を返します。

内部Basicシンタックス

Function Minute(tmTime As Date) As Long

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **tmTime：処理する「時刻+日付」形式のパラメータ**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献 301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim strMinute
strMinute=Minute(Date())
RetVal=strMinute : '現在の時刻の分の部分を返します。例えば、時刻が15:45:30の場合は、"45"を返します。
```

MkDir()

新しいディレクトリを作成します。

内部Basicシンタックス

Function MkDir(strMkDirectory As String) As Long

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strMkDirectory：作成するディレクトリのフルパス**

戻りコード

- 0：**成功**
- 0**以外：エラーコード**

例

```
Dim lErr as Long
'c:\tmpディレクトリの作成
lErr = MkDir("c:\tmp")
```

Month()

tmDateパラメータに含まれている日付の月の部分を返します。

内部Basicシンタックス

Function Month(tmDate As Date) As Long

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **tmDate：処理する「時刻+日付」形式のパラメータ**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献 301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim lMonth as Long
lMonth=Month(Date())
RetVal=lMonth :' 現在の月を返します。
```

Name()

ファイルの名前を変更します。

内部Basicシンタックス

Function Name(strSource As String, strDest As String)

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strSource：名前を変更するファイルのフルパス**
- **strDest：新しいファイル名**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim lErr as Long
' "C:\tmp\src.txt"を"D:\tmp\dst.txt"に名前変更
lErr = Name("C:\tmp\src.txt", "D:\tmp\dst.txt")
```

Now()

現在の日付と時刻を返します。

内部Basicシンタックス

Function Now() As Date

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

NPER()

定額の定期的な支払、および一定利率に基づく、年間の支払回数を返します。

内部Basicシンタックス

Function NPER(dblRate As Double, dblPmt As Double, dblPV As Double, dblFV As Double, iType As Long) As Double

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **dblRate：支払日ごとの利率。例えば、年利が6%のローンの毎月の支払日の利率は、次のようになります。**

0.06/12=0.005 **または** 0.5%

- **dblPmt：1回の支払金額。支払金額には一般的に元金と利子が含まれます。**
- **dblPV：実際に支払わなければならない金額（総額）**
- **dblFV：支払が終了したときの差引残高または将来の支払用に確保する金額。一般に、特にローンを返済する場合は、このパラメータを「0」に設定します。実際、すべての支払を済ませるとローンの値はゼロになります。**
- **iType：支払期限を示します。次のいずれかの値になります。**
  - **0：支払が後払い（期間内の最後）の場合**
  - **1：支払が先払い（期間内の初め）の場合**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

注

注意:

支払う金額（Pmtパラメータの金額）はマイナスの数値、受け取る総額はプラスの数値で表わされます。

Oct()

10進法のパラメータの８進法の値を返します。

内部Basicシンタックス

Function Oct(dValue As Double) As String

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **dValue：８進法の値を取得する数値**

戻りコード

エラーの場合は、次の2**つの処理方法を用意します。**

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim dSeed as Double
dSeed = Int((10*Rnd)-5)
RetVal = Oct(dSeed)
```

ParseDate()

文字列フォーマットの日付を、Basic**日付オブジェクトへ変換します。**

内部Basicシンタックス

Function ParseDate(strDate As String, strFormat As String, strStep As String) As Date

用途

バージョン：3.6.0

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **strDate：文字列フォーマットの日付**
- **strFormat：文字列に含まれている日付のフォーマット。以下の値が使用可能です。**
  - DD/MM/YY
  - DD/MM/YYYY
  - MM/DD/YY
  - MM/DD/YYYY
  - YYYY/MM/DD
  - Date：**クライアントコンピュータの日付パラメータに基づいた日付**
  - DateInter ：**国際標準形式の日付**
- **strStep：このオプションパラメータには、文字列内で使用される日付の区切り文字が含まれます。許可される区切り文字は「\」と「-」です。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim dDate as date
dDate=ParseDate("2001/05/01", "YYYY/MM/DD")
```

ParseDMYDate()

次の形式の日付から、日付オブジェクト（BASIC形式）を返します。

```
dd/mm/yyyy
```

内部Basicシンタックス

Function ParseDMYDate(strDate As String) As Date

用途

バージョン：3.5

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strDate：文字列として保存されている日付**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献 301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim dDate as Date
dDate = ParseDMYDate("31/02/2003")
```

ParseMDYDate()

次の形式の日付から、日付オブジェクト（BASIC形式）を返します。

```
mm/dd/yyyy
```

内部Basicシンタックス

Function ParseMDYDate(strDate As String) As Date

用途

バージョン：3.5

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strDate：文字列として保存されている日付**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim dDate as Date
dDate = ParseMDYDate("02/31/2003")
```

ParseYMDDate()

この関数は、(Basic**で識別される**)Date**オブジェクトを**yyyy/mm/dd**形式で返します。**

内部Basicシンタックス

Function ParseYMDDate(strDate As String) As Date

用途

バージョン：3.5

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **strDate：文字列として保存されている日付**

戻りコード

エラーの場合は、次の2**つの処理方法を用意します。**

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献** 301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim dDate as Date
dDate = ParseYMDDate("2003/02/31")
```

PMT()

定額の定期的な支払、および一定利率に基づいて計算した、年間支払金額を返します。

内部Basicシンタックス

Function PMT(dblRate As Double, iNper As Long, dblPV As Double, dblFV As Double, iType As Long) As Double

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **dblRate：支払日ごとの利率。例えば、年利が**6%**のローンの毎月の支払日の利率は、次のようになります。**

```
0.06/12=0.005 または 0.5%
```

- **iNper：総支払回数**
- **dblPV：実際に支払わなければならない金額（総額）**
- **dblFV：支払が終了した時の差引残高または将来の支払用に確保する金額。一般に、特にローンを返済する場合は、このパラメータを「**0**」に設定します。実際、すべての支払を済ませるとローンの値はゼロになります。**
- **iType：支払期限を示します。次のいずれかの値になります。**
  - **0：支払が後払い（期間内の最後）の場合**
  - **1：支払が先払い（期間内の初め）の場合**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

注

注意:

- RateおよびNperパラメータの計算には同じ単位の支払金額を使用する必要があります。
- 支払う金額（Pmtパラメータの金額）はマイナスの数値、受け取る総額はプラスの数値で表わされます。

PPMT()

定額の定期的な支払い、および一定利率に基づき、指定した支払日に返済する元金の金額を返します。

内部Basicシンタックス

```
Function PPMT(dblRate As Double, iPer As Long, iNper As Long, dblPV As Double,
dblFV As Double, iType As Long) As Double
```

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **dblRate：支払日ごとの利率。例えば、年利が6%のローンの毎月の支払日の利率は、次のようになります。**

0.06/12=0.005 **または** 0.5%

- **iPer：計算する期間。1からNperの数値の間で指定します。**
- **iNper：総支払回数**
- **dblPV：実際に支払わなければならない金額（総額）**
- **dblFV：支払が終了した時の差引残高または将来の支払用に確保する金額。一般に、特にローンを返済する場合は、このパラメータを「0」に設定します。実際、すべての支払を済ませるとローンの値はゼロになります。**
- **iType：支払期限を示します。次のいずれかの値になります。**
  - **0：支払が後払い（期間内の最後）の場合**
  - **1：支払が先払い（期間内の初め）の場合**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

注

注意:

- **Rate およびNperパラメータの計算には同じ単位の支払金額を使用する必要があります。**
- **支払う金額（Pmtパラメータの金額）はマイナスの数値、受け取る総額はプラスの数値で表わされます。**

PV()

定額の定期的な支払い、および固定利率に基づいて計算した、実際の年間支払総額を返します。

内部Basicシンタックス

Function PV(dblRate As Double, iNper As Long, dblPmt As Double, dblFV As Double, iType As Long) As Double

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **dblRate：支払日ごとの利率。例えば、年利が6%のローンの毎月の支払日の利率は、次のようになります。**

0.06/12=0.005 **または** 0.5%

- **iNper：総支払回数**
- **dblPmt：1回の支払金額。支払金額には一般的に元金と利子が含まれます。**
- **dblFV：支払が終了した時の差引残高または将来の支払用に確保する金額。一般に、特にローンを返済する場合は、このパラメータを「0」に設定します。実際、すべての支払を済ませるとローンの値はゼロになります。**
- **iType：支払期限を示します。次のいずれかの値になります。**
  - **0：支払が後払い（期間内の最後）の場合**
  - **1：支払が先払い（期間内の初め）の場合**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

注

注意:

- Rate およびNperパラメータの計算には同じ単位の支払金額を使用する必要があります。
- 支払う金額（Pmtパラメータの金額）はマイナスの数値、受け取る総額はプラスの数値で表わされます。

Randomize()

乱数発生関数を初期化します。

内部Basicシンタックス

Function Randomize(lValue As Long)

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lValue：特定の新しい初期値を指定して乱数発生関数であるRnd関数を初期化するときに使います。 このパラメータを省略すると、システムクロックからの値が初期値として使われます。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献** 301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

注

次も参照してください:

- Rnd() [**献** 461]

例

```
Dim MyNumber
Randomize
MyNumber= Int((10*Rnd)+1) : ' 1 - 10の乱数値を返します。
RetVal=MyNumber
```

RATE()

年間支払金額のうちの1回分の支払金額の利率を返します。

内部Basicシンタックス

Function RATE(iNper As Long, dblPmt As Double, dblFV As Double, dblPV As Double, iType As Long, dblGuess As Double) As Double

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **iNper：総支払回数**
- **dblPmt：**1**回の支払金額。支払いには一般的に元金と利子が含まれます。**
- **dblFV：支払が終了した時の差引残高または将来の支払用に確保する金額。一般に、特にローンを返済する場合は、このパラメータを「**0**」に設定します。実際、すべての支払を済ませるとローンの値はゼロになります。**
- **dblPV：実際に支払わなければならない金額（総額）**
- **iType：支払期限を示します。次のいずれかの値になります。**
  - **0：支払が後払い（期間内の最後）の場合**
  - **1：支払が先払い（期間内の初め）の場合**
- **dblGuess：**1**回の支払の利率の推定値**

戻りコード

エラーの場合は、次の2**つの処理方法を用意します。**

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

注

注意:

- 支払う金額（Pmtパラメータの値）はマイナスの数値、受け取る総額はプラスの数値になります。
- この関数は、Guessパラメータに指定した値から開始して、繰り返し計算を実行します。20回繰り返しても結果が出ない場合は、この関数は無効となります。

RemoveRows()

strRowNamesパラメータに指定されている行をリストから削除します。

この関数は、［ListBox］コントロールタイプの値を処理する時に役立ちます。このタイプのコントロールの値は、次の文字で区切られています。

- **パイプ文字（|）は、列を区切ります。**
- **コンマ（,）は、行を区切ります。**
- **各行の終わりには、等号（=）とその後に固有のIDが付いています。**

内部Basicシンタックス

Function RemoveRows(strList As String, strRowNames As String) As String

用途

バージョン：3.5

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strList：処理する［ListBox］コントロールの値が含まれている文字列**
- **strRowNames：削除する行のID。IDが複数ある場合は、カンマで区切ります。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

注

次も参照してください:

- SubList() [**献** 477]
- SetSubList() [**献** 467]
- ApplyNewVals() [**献** 369]

例

```
Dim MyStr
MyStr=RemoveRows("a1|a2=a0,b1|b2=b0", "a0,c0") :' "b1|b2=b0"を返します。
RetVal=MyStr
```

Replace()

strDataパラメータの文字列に含まれるstrOldPatternパラメータの文字列をすべてstrNewPatternパラメータの文字列で置き換えます。bCaseSensitiveパラメータを使って、検索するstrOldPatternパラメータの文字列の大文字／小文字を区別できます。

内部Basicシンタックス

Function Replace(strData As String, strOldPattern As String, strNewPattern As String, bCaseSensitive As Long) As String

用途

バージョン：3.5

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strData：置換される文字列を含んでいる文字列**
- **strOldPattern：strDataパラメータに含まれている検索の対象となる文字列。**
- **strNewPattern：検索した文字列を置換する文字列**
- **bCaseSensitive：このパラメータを使って、大文字と小文字を区別する（=1）、または区別しない（=0）かを指定します。デフォルトではこのパラメータは1に設定されています。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献** 301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim MyStr
MyStr=Replace("youmeyoumeyou", "you", "me",0) :' "mememememe"を返します。
MyStr=Replace("youmeyoumeyou", "You", "me",1) :' "youmeyoumeyou"を返します。
MyStr=Replace("youmeYoumeyou", "You", "me",1) :' "youmememeyou"を返します。
```

Right()

文字列の右端からiNumber**で指定した数の文字を返します。**

内部Basicシンタックス

Function Right(strString As String, lNumber As Long) As String

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **strString：処理する文字列**
- **lNumber**: **戻される文字数。**

戻りコード

エラーの場合は、次の2**つの処理方法を用意します。**

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim lWord, strMsg, rWord, iPos :' Declare variables.
strMsg = "Left() Test."
iPos = lnStr(1, strMsg, " ") : ' スペースを探す
lWord = Left(strMsg, iPos - 1) : ' 左側の単語を取得
rWord = Right(strMsg, Len(strMsg) - iPos) : ' 右側の単語を取得
strMsg=rWord+lWord : ' 2つの単語を交換
RetVal=strMsg
```

RightPart()

strSepパラメータに指定されている区切り文字の右側の文字列を1つ取得します。
右から左に向かって区切り文字を探します。
bCaseSensitiveパラメータを使って、大文字と小文字を区別することもできます。

内部Basicシンタックス

Function RightPart(strFrom As String, strSep As String, bCaseSensitive As Long) As String

用途

バージョン：3.5

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **strFrom：処理する文字列**
- **strSep：処理する文字列の区切り文字**
- **bCaseSensitive：このパラメータを使って、大文字と小文字を区別する（**=1**）、または区別しない（**=0**）かを指定します。**

戻りコード

エラーの場合は、次の2**つの処理方法を用意します。**

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献** 300**]関数（必要に応じて**AmLastErrorMsg() **[献**301**]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

文字列"This_is_a_test"**に、LeftPart、LeftPartFromRight、RightPart、および RightPartFromLeft関数を使った例を示します。**

```
LeftPart("This_is_a_test","_",0)
```

"This"**を返します。**

```
LeftPartFromRight("This_is_a_test","_",0)
```

"This_is_a"**を返します。**

```
RightPart("This_is_a_test","_",0)
```

"test"**を返します。**

```
RightPartFromLeft("This_is_a_test","_",0)
```

"is_a_test"**を返します。**

RightPartFromLeft()

strSepパラメータに指定されている区切り文字の右側の文字列を1つ取得します。
左から右に向かって区切り文字を探します。
bCaseSensitiveパラメータを使って、大文字と小文字を区別することもできます。

内部Basicシンタックス

Function RightPartFromLeft(strFrom As String, strSep As String, bCaseSensitive As Long) As String

用途

バージョン：3.5

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strFrom：処理する文字列**
- **strSep：処理する文字列の区切り文字**
- **bCaseSensitive：このパラメータを使って、大文字と小文字を区別する（=1）、または区別しない（=0）かを指定します。デフォルトではこのパラメータは1に設定されています。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

文字列"This_is_a_test"に、LeftPart、LeftPartFromRight、RightPart、およびRightPartFromLeft関数を使った例を示します。

```
LeftPart("This_is_a_test","_",0)
```

"This"を返します。

```
LeftPartFromRight("This_is_a_test","_",0)
```

"This_is_a"を返します。

```
RightPart("This_is_a_test","_",0)
```

"test"を返します。

```
RightPartFromLeft("This_is_a_test","_",0)
```

"is_a_test"を返します。

RmAllInDir()

この関数は、フォルダ内の全アイテム(ファイルとフォルダ)を削除します。フォルダ自体は削除しません。

内部Basicシンタックス

Function RmAllInDir(strRmDirectory As String, bStopIfError As Long) As Long

用途

バージョン：3.4.0

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **strRmDirectory**: **このパラメータには、操作に関係するフォルダのフルパスが入ります。**
- **bStopIfError**:**このパラメータが**1**に設定されている場合、ファイルやフォルダの削除が行えない際に削除操作を中断します。**0**に設定されている場合、操作は継続され次のファイルやフォルダに移ります。**

戻りコード

- 0**：成功**
- 0**以外：エラーコード**

例

```
RetVal = RmAllInDir("c:\files\test", 1)
```

RmDir()

既存のディレクトリを1つ削除します。

内部Basicシンタックス

Function RmDir(strRmDirectory As String) As Long

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strRmDirectory：削除するディレクトリのフルパス**

戻りコード

- 0：**成功**
- 0**以外：エラーコード**

注

注意:

削除するディレクトリは空である必要があります。空でない場合、関数は実行されません

例

```
RetVal = RmDir("c: mp")
```

Rnd()

乱数を含んでいる値を返します。

内部Basicシンタックス

Function Rnd(dValue As Double) As Double

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **dValue：関数の実行モードを定義する場合に使うパラメータ**
  - **0より小さい値：関数を実行するたびに同じ数値を発生します。**
  - **0より大きい値：次に発生する乱数**
  - **0：直前に発生した乱数**
  - **省略：次に発生する乱数**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献** 300**]関数（必要に応じて**AmLastErrorMsg() **[献**301**]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

注

注意:

この関数を呼び出す前に、Randomize関数をパラメータなしで使用し、乱数ジェネレータを初期化する必要があります。

次も参照してください:

- Randomize() [**献** 449]

例

```
Dim MyNumber
Randomize
MyNumber= Int((10*Rnd)+1) : ' 1 - 10のいずれかの値を乱数として返します。
RetVal=MyNumber
```

RoundValue()

この関数は、iDigitsパラメータで指定した小数点以下の桁数で、数値の丸め値を計算します。

内部Basicシンタックス

Function RoundValue(dValue As Double, iDigits As Long) As Double

用途

バージョン：3.4.0

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **dValue**: **このパラメータには、丸め演算の対象となる数値が入ります。**

- **iDigits**: このパラメータには、丸め演算を行う小数点位置の数値が入ります。

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

例

次の例:

```
RetVal = RoundValue(1.2568, 2)
```

は、次の値を戻します:

```
1.26
```

次の例:

```
RetVal = RoundValue(1.2568, 0)
```

は、次の値を戻します:

```
1
```

RTrim()

文字列の末尾に含まれるスペースをすべて取り除きます。

内部Basicシンタックス

Function RTrim(strString As String) As String

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strString：処理する文字列**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
' This example uses the LTrim and RTrim functions to strip leading ' and trailing spaces,
respectively, from a string variable.
' It uses the Trim function alone to strip both types of spaces.
' LCase and UCase are also shown in this example as well as the use
' of nested function calls

Dim strString as String
Dim strTrimString as String
strString = "  <-Trim->  " :' Initialize string.
strTrimString = LTrim(strString) :' strTrimString = "<-Trim->  ".
strTrimString = LCase(RTrim(strString)) :' strTrimString = "  <-trim->".
strTrimString = LTrim(RTrim(strString)) :' strTrimString = "<-Trim->".
' Using the Trim function alone achieves the same result.
strTrimString = UCase(Trim(strString)) :' strTrimString = "<-TRIM->".
RetVal= "|" & strTrimString & "|"
```

Second()

tmTimeパラメータの時刻の秒の部分の数値を返します。

内部Basicシンタックス

Function Second(tmTime As Date) As Long

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **tmTime：処理する「時刻+日付」形式のパラメータ**

戻りコード

エラーの場合は、次の2**つの処理方法を用意します。**

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim strSecond
strSecond=Second(Date())
RetVal=strSecond : '現在の時刻の秒の部分を返します。例えば、時刻が15:45:30の場合は、"30"を返します。
```

SetMaxInst()

Basicスクリプトが実行できる命令の最大数を設定します。デフォルトでは命令の数は10000個に制限されています。

APIシンタックス

long SetMaxInst(long lMaxInst);

内部Basicシンタックス

Function SetMaxInst(lMaxInst As Long) As Long

用途

バージョン：4.3.0

	使用
AssetCenter API	✓
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **lMaxInst：スクリプトが実行できる命令の最大数。**

戻りコード

- 0：成功
- 0以外：エラーコード

注

注意:

lMaxInstパラメータを「0」に設定すると、スクリプトが実行できる命令の数は無制限になります。

SetSubList()

［ListBox］コントロールのサブリストの値を定義します。

内部Basicシンタックス

Function SetSubList(strValues As String, strRows As String, strRowFormat As String) As String

用途

バージョン：3.5

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strValues：処理する［ListBox］コントロールの値が含まれている文字列**
- **strRows：strValuesパラメータの文字列に追加する値またはその文字列に含まれている文字を置換する値の一覧。各値をパイプ文字（|）で区切ります。処理する行は、等号（=）記号の右側にあるIDで識別されます。不明な行は処理されません。**
- **strRowFormat：サブリストの書式化命令。各命令をパイプ文字（|）で区切ります。このパラメータには、次の文字を使います。**
  - **1は、サブリストの最初の列の情報を示します。**
  - **i -jは、列のグループを定義します。**
  - **ハイフン（-）は、すべての列を処理することを示します。**
  - **不明な列の値は返しません。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**

- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim MyStr
MyStr=SetSubList("a1|a2|a3=a0,b1|b2|b3=b0,c1|c2|c3=c0", "A2|A1=a0, B2|B1=b0", "2|1
") : ' "A1|A2|a3=a0,B1|B2|b3=b0,c1|c2|c3=c0"を返します。
MyStr=SetSubList("a1|a2|a3=a0,b1|b2|b3=b0,c1|c2|c3=c0", "Z2=*,B2=b0", "2") : ' "a1|Z2
|a3=a0,b1|B2|b3=b0,c1|Z2|c3=c0"を返します。
MyStr=SetSubList("a1|a2|a3=a0,b1|b2|b3=b0,c1|c2|c3=c0", "B5|B6|B7=b0,C5|C6,C7=c0
", "5-7") : ' "a1|a2|a3=a0,b1|b2|b3||B5|B6|B7=b0,c1|c2|c3||C5|C6|C7=c0"を返します。
MyStr=SetSubList("a1|a2|a3=a0,b1|b2|b3=b0,c1|c2|c3=c0", "B1|B2|B3|B4=b0", "-") : ' "a1
|a2|a3=a0,B1|B2|B3|B4=b0,c1|c2|c3=c0"を返します。
MyStr=SetSubList("A|B|C,D|E|F", "X=*", "2") : ' "A|X|C,D|X|F"を返します。
RetVal=""
```

Sgn()

数値の記号を表わす値を返します。

内部Basicシンタックス

Function Sgn(dValue As Double) As Double

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **dValue：記号を取得する数値**

戻りコード

次のいずれかの値を返します。

- **1：0より大きい数値を示します。**
- **0：0を示します。**
- **-1：0より小さい数値を示します。**

例

```
Dim dNumber as Double
dNumber=-256
RetVal=Sgn(dNumber)
```

Shell()

実行可能プログラムを起動します。

内部Basicシンタックス

Function Shell(strExec As String, bShowWindow As Long, bBackground As Long) As Long

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓

	使用
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strExec：起動する実行可能ファイルのフルパス**
- **bShowWindow: パラメータが1に設定されている場合(デフォルト値)、プログラム起動時にコマンドボックスが表示されます。0に設定されている場合、コマンドボックスは表示されません。**
- **bBackground: このパラメータが1に設定されている場合(デフォルト値)、関数はプログラムの実行完了を待ってからユーザに制御を返します(同期実行)。0に設定されている場合、プログラムは非同期に実行されます。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献 301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim MyId
MyId=Shell("C:\WinNT\explorer.exe")
RetVal=""
```

Sin()

数値のサインを返します。単位はラジアンです。

内部Basicシンタックス

Function Sin(dValue As Double) As Double

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **dValue：サインを取得する数値**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

注

注意:

度からラジアンへの変換公式は次になります:

ラジアンでの角度 = (**度での角度**) * Pi / 180

例

```
Dim dCalc as Double
dCalc=Sin(2.79)
RetVal=dCalc
```

Space()

iSpaceパラメータに指定されている数のスペース（空白文字）を挿入した文字列を作成します。

内部Basicシンタックス

Function Space(iCount As Long) As String

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **iCount**: **文字列に挿入する空白文字の個数。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献** 301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

注

注意:

この関数は、文字列を書式化したり、固定長の文字列から日付を削除する時に使います。

例

```
Dim MyString
' 10個のスペースを返します。
MyString = Space(10)
' 2つの文字列の間に10個のスペースを挿入します。
MyString = "Space" & Space(10) &  "inserted"
RetVal=MyString
```

Sqr()

数値の平方根を返します。

内部Basicシンタックス

Function Sqr(dValue As Double) As Double

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **dValue：平方根を取得する数値**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献 301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim dCalc as Double
dCalc=Sqr(81)
RetVal=dCalc
```

Str()

数値を文字列に変換します。

内部Basicシンタックス

Function Str(strValue As String) As String

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **strValue：文字列に変換する数値**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim dNumber as Double
dNumber=Cos(2.79)
RetVal=Str(dCalc)
```

StrComp()

2つの文字列を比較します。

内部Basicシンタックス

Function StrComp(strString1 As String, strString2 As String, iOptionCompare As Long) As Long

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **strString1：最初の文字列**
- **strString2：**2**つめの文字列**

- **iOptionCompare：比較のタイプ。バイナリの比較には「0」、テキストの比較には「1」を設定します。**

戻りコード

- -1：**strString1はstrString2よりも大きい。**
- 0：**strString1はstrString2と等しい。**
- 1：**strString1はstrString2よりも小さい。**

String()

strString文字をiCount回繰り返した文字列を返します。

内部Basicシンタックス

Function String(iCount As Long, strString As String) As String

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **iCount：strString文字を繰り返す回数**
- **strString：繰り返す文字**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**

- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim iCount as Integer
Dim strTest as String
strTest="T"
iCount=5
RetVal=String(iCount,strTest)
```

SubList()

［ListBox］**コントロールの値の文字列に含まれている値一覧のサブリストを返します。**

内部Basicシンタックス

Function SubList(strValues As String, strRows As String, strRowFormat As String) As String

用途

バージョン：3.5

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **strValues：処理する**［ListBox］**コントロールの値が含まれている文字列**
- **strRows：サブリストに含める行の**ID**。各**ID**をカンマ（,）で区切って指定します。特定のワイルドカード文字を使用できます。**

- （*）は、サブリスト内のすべての行を含むことを示します。
- 不明なIDを指定した場合は、サブリストの空の値が返されます。

- **strRowFormat：サブリストの書式化命令。各命令をパイプ文字（|）で区切ります。このパラメータには、次の文字を使います。**
  - 1は、取得するサブリストのリストの最初の列の情報を示します。
  - 0は、取得するサブリストのリストの行のIDを示します。
  - アスタリスク（*）は、行のIDを除くすべての列の情報を示します。
  - 不明な列の値は返しません。

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

例

```
Dim MyStr
MyStr=SubList("a1|a2|a3=a0,b1|b2|b3=b0,c1|c2|c3=c0", "a0,b0,a0", "3|2|3") : ' "a3|a2|a3,b3|b2|b3,a3|a2|a3"を返します。
MyStr=SubList("a1|a2|a3=a0,b1|b2|b3=b0,c1|c2|c3=c0", "*", "*|0") :'Returns "a1|a2|a3|a0,b1|b2|b3|b0,c1|c2|c3|c0"
MyStr=SubList("a1|a2|a3=a0,b1|b2|b3=b0,c1|c2|c3=c0", "*", "*=0") : ' "a1|a2|a3=a0,b1|b2|b3=b0,c1|c2|c3=c0"を返します。
MyStr=SubList("a1|a2|a3=a0,b1|b2|b3=b0,c1|c2|c3=c0", "*", "999=0") : ' "=a0,=b0,=c0"を返します。
MyStr=SubList("a1|a2|a3=a0,b1|b2|b3=b0,c1|c2|c3=c0", "z0", "*=0") : ' ""を返します。
MyStr=SubList("a1|a2|a3=a0,b1|b2|b3=b0,c1|c2|c3=c0", "*", "=1") : ' "=a1,=b1,=c1"を返します。
MyStr=SubList("A|B|C,D|E|F", "*", "2=0") : ' "B,E"を返します。
RetVal=""
```

SysEnumToComboBox()

システムリストデータの項目を整理し、ウィザードのリストコントロールと互換性のある形式にします。これにより、システムリストデータの値をウィザードのドロップダウンリストに表示できます。

内部Basicシンタックス

Function SysEnumToComboBox(strFormat As String) As String

用途

バージョン：4.3.0

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	
「スクリプト」タイプのアクションの設定	
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strFormat：システムリストデータのエントリのリスト。AmGetFieldFormat()関数の実行結果をこのパラメータに指定するのがよい方法です。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献** 300**]関数（必要に応じて**AmLastErrorMsg() **[献**301**]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

次の例は、amWorkOrder**テーブル中のシステムリストデータ**seStatus**の値をウィザードのリストコントロールと互換性のある形式に整理します。**

```
Dim strFormat As String
strFormat = AmGetFieldFormat(AmGetFieldFromName(AmGetTableFromName("amW
orkOrder"), "seStatus"))
RetVal = SysEnumToComboBox(strFormat)
```

Tan()

数値のタンジェントを返します。単位はラジアンです。

内部Basicシンタックス

Function Tan(dValue As Double) As Double

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **dValue：タンジェントを取得する数値**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

注

注意:

度からラジアンへの変換公式は次になります:

ラジアンでの角度 = (**度での角度**) * Pi / 180

例

```
Dim dCalc as Double
dCalc=Tan(2.79)
RetVal=dCalc
```

Time()

現在の時刻を返します。

内部Basicシンタックス

Function Time() As Date

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

戻りコード

エラーの場合は、次の2**つの処理方法を用意します。**

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**

- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献 301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
RetVal = Time()
```

Timer()

午前0時0分から経過した秒数を返します。

内部Basicシンタックス

Function Timer() As Double

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献 301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
RetVal = Timer()
```

TimeSerial()

iHour、iMinute、iMinuteパラメータの形式に従って時刻を返します。

内部Basicシンタックス

Function TimeSerial(iHour As Long, iMinute As Long, iSecond As Long) As Date

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **iHour：時間**
- **iMinute：分**
- **iSecond：秒**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

各パラメータにはそれぞれ、日、月、または年を表す数式を使用できます。

```
TimeSerial(12-8, -10, 0)
```

例えば、上記の例は次の値を返します。

```
3:50:00
```

パラメータの値が予想される範囲（分ならば0 - 59、時間ならば0 - 24など）以外の値の場合は、次のレベルのパラメータに繰り上げられます。つまり、iMinuteに75を入力すると、1時間と15分に解釈されます。

次の例は、

```
TimeSerial (16, 50, 45)
```

次の値を返します。

```
16:50:45
```

TimeValue()

「日付＋時刻」の値の時刻の部分を返します。

内部Basicシンタックス

Function TimeValue(tmTime As Date) As Date

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **tmTime：「日付＋時刻」形式の日付**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

次の例は、

```
TimeValue ("1999/09/24 15:00:00")
```

次の値を返します。

```
15:00:00
```

ToSmart()

ソース文字列の各語の始めを大文字にします。

内部Basicシンタックス

Function ToSmart(strString As String) As String

用途

バージョン：3.5

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **strString：処理するソース文字列**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

例

次の例:

```
RetVal = ToSmart ("hello world")
```

は次の値を戻します:

```
Hello World
```

Trim()

先頭と末尾のスペースを削除した文字列を返します。

内部Basicシンタックス

Function Trim(strString As String) As String

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strString：処理する文字列**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
' This example uses the LTrim and RTrim functions to strip leading ' and trailing spaces,
respectively, from a string variable.
' It uses the Trim function alone to strip both types of spaces.
' LCase and UCase are also shown in this example as well as the use
' of nested function calls

Dim strString as String
Dim strTrimString as String
strString = "  <-Trim->  " :' Initialize string.
strTrimString = LTrim(strString) :' strTrimString = "<-Trim->  ".
strTrimString = LCase(RTrim(strString)) :' strTrimString = "  <-trim->".
strTrimString = LTrim(RTrim(strString)) :' strTrimString = "<-Trim->".
' Using the Trim function alone achieves the same result.
strTrimString = UCase(Trim(strString)) :' strTrimString = "<-TRIM->".
RetVal= "|" & strTrimString & "|"
```

UCase()

文字列に含まれるすべての小文字を大文字に変換して返します。

内部Basicシンタックス

Function UCase(strString As String) As String

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strString：大文字に変換する文字列**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
' This example uses the LTrim and RTrim functions to strip leading ' and trailing spaces,
respectively, from a string variable.
' It uses the Trim function alone to strip both types of spaces.
' LCase and UCase are also shown in this example as well as the use
' of nested function calls

Dim strString as String
Dim strTrimString as String
strString = "  <-Trim->  " :' Initialize string.
strTrimString = LTrim(strString) :' strTrimString = "<-Trim->  ".
strTrimString = LCase(RTrim(strString)) :' strTrimString = "  <-trim->".
strTrimString = LTrim(RTrim(strString)) :' strTrimString = "<-Trim->".
' Using the Trim function alone achieves the same result.
strTrimString = UCase(Trim(strString)) :' strTrimString = "<-TRIM->".
RetVal= "|" & strTrimString & "|"
```

UnEscapeSeparators()

1つの文字列からすべてのエスケープ文字を削除します。

内部Basicシンタックス

Function UnEscapeSeparators(strSource As String, strEscChar As String) As String

用途

バージョン：3.5

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **strSource：処理する文字列**
- **strEscChar：削除するエスケープ文字**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim MyStr
MyStr=UnEscapeSeparators("you\|me\|you\|", "\") :' "you|me|you|"を返します。
RetVal=""
```

Union()

区切り文字で区切られている2つの文字列を合体します。重複する文字列は削除されます。

内部Basicシンタックス

Function Union(strListOne As String, strListTwo As String, strSeparator As String, strEscChar As String) As String

用途

バージョン：3.5

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strListOne：最初の文字列**
- **strListTwo：2番目の文字列**
- **strSeparator：各文字列の区切り文字**
- **strEscChar：エスケープ文字。この文字が区切り文字の前に付くと、その区切り文字は無視されます。**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- **AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献 300]関数（必要に応じて**AmLastErrorMsg() **[献 301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim MyStr
MyStr=Union("a1|a2,b1|b2", "a1|a3,b1|b2", ",", "\") :' "a1|a2,b1|b2,a1|a3"を返します。
MyStr=Union("a1|a2,b1|b2", "a1|a3\,b1|b2", ",", "\") :' "a1|a2,b1|b2,a1|a3\,b1|b2"を返します。
RetVal=""
```

UTCToLocalDate()

UTCフォーマットの日付（タイムゾーンに関係しない）「日付＋時刻」型の日付に変換します。

内部Basicシンタックス

Function UTCToLocalDate(tmUTC As Date) As Date

用途

バージョン：3.5

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO**」スクリプト**	✓

パラメータ

- **tmUTC：**UTCフォーマットの日付

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() [**献** 300]**関数（必要に応じて**AmLastErrorMsg() [**献**301]**関数も）を呼び出し、エラーが発生し**

たかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

例

```
RetVal = UTCToLocaldate([DateTime])
```

Val()

数値を表す文字列を倍精度型に変換します。

内部Basicシンタックス

Function Val(strString As String) As Double

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **strString：変換する文字列**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。
- 外部プログラムからこの関数を呼び出した場合は、AmLastError() [献 300]関数（必要に応じてAmLastErrorMsg() [献301]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。

例

```
Dim strYear
Dim dYear as Double
strYear=Year(Date())
dYear=Val(strYear)
RetVal=dYear :'Returns the current year
```

WeekDay()

tmDateパラメータの日付の曜日の部分を返します。

内部Basicシンタックス

Function WeekDay(tmDate As Date) As Long

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **tmDate：処理する「時刻+日付」形式のパラメータ**

戻りコード

「1」は日曜日、「2」は月曜日、…、「7」は土曜日のように、週の曜日に対応した数値を返します。

例

```
Dim strWeekDay
strWeekDay=WeekDay(Date())
RetVal=strWeekDay : ' 曜日を返します。
```

Year()

tmDateパラメータの日付の年の部分を返します。

内部Basicシンタックス

Function Year(tmDate As Date) As Long

用途

バージョン：3.00

	使用
AssetCenter API	
リンクまたはフィールドの設定スクリプト	✓
「スクリプト」タイプのアクションの設定	✓
ウィザードスクリプト	✓
ウィザードの「FINISH.DO」スクリプト	✓

パラメータ

- **tmDate：処理する「時刻+日付」形式のパラメータ**

戻りコード

エラーの場合は、次の2つの処理方法を用意します。

- AssetCenter**内で、この関数を含むスクリプトを中断し、ユーザの画面にエラーメッセージを表示します。**
- **外部プログラムからこの関数を呼び出した場合は、**AmLastError() **[献** 300**]関数（必要に応じて**AmLastErrorMsg() **[献** 301**]関数も）を呼び出し、エラーが発生したかどうかを確認し、エラーが発生した場合はエラーメッセージを返すようにします。**

例

```
Dim strYear
strYear=Year(Date())
RetVal=strYear :'現在の年を戻します。
```

IV インデックス

使用可能な関数 - 全関数のリスト

- AmConvertDateBasicToUnix
- AmConvertDateIntlToUnix
- AmConvertDateStringToUnix
- AmConvertDateUnixToBasic
- AmConvertDateUnixToIntl
- AmConvertDateUnixToString
- AmConvertDoubleToString
- AmConvertMonetaryToString
- AmConvertStringToDouble
- AmConvertStringToMonetary
- AmCounter
- AmCreateAssetPort
- AmCreateAssetsAwaitingDelivery
- AmCreateCable
- AmCreateCableBundle
- AmCreateCableLink
- AmCreateDelivFromPO
- AmCreateDevice
- AmCreateDeviceLink
- AmCreateEstimFromReq
- AmCreateEstimsFromAllReqLines
- AmCreateInvFromPO
- AmCreateLink
- AmCreateOrUpdateInvoiceFromReceipt
- AmCreatePOFromEstim
- AmCreatePOFromReq
- AmCreatePOrderFromRequest
- AmCreatePOrdersFromRequest
- AmCreatePOsFromAllReqLines
- AmCreateProjectCable
- AmCreateProjectDevice
- AmCreateProjectTrace
- AmCreateReceiptFromPOrder
- AmCreateRecord
- AmCreateRequestToInvoice
- AmCreateRequestToPOrder
- AmCreateRequestToReceipt
- AmCreateReturnFromReceipt
- AmCreateTraceHist
- AmCreateTraceLink
- AmCryptPassword
- AmCurrentDate
- AmCurrentIsoLang
- AmCurrentLanguage
- AmCurrentServerDate
- AmDateAdd
- AmDateAddLogical
- AmDateDiff
- AmDbExecAql
- AmDbGetDate
- AmDbGetDouble
- AmDbGetList
- AmDbGetListEx
- AmDbGetLong
- AmDbGetPk
- AmDbGetString
- AmDbGetStringEx
- AmDeadLine
- AmDecrementLogLevel
- AmDefAssignee
- AmDefaultCurrency
- AmDefEscalationScheme
- AmDefGroup
- AmDeleteLink
- AmDeleteRecord
- AmDisconnectTrace
- AmDuplicateRecord
- AmEndOfNthBusinessDay
- AmEnumValList
- AmESDAddComputers
- AmESDCreateTask
- AmEvalScript
- AmExecTransition
- AmExecuteActionById
- AmExecuteActionByName
- AmExportDocument

- AmExportReport
- AmFindCable
- AmFindDevice
- AmFindRootLink
- AmFindTermDevice
- AmFindTermField
- AmFlushTransaction
- AmFormatCurrency
- AmFormatLong
- AmGeneratePlanningData
- AmGenSqlName
- AmGetCatRef
- AmGetCatRefFromCatProduct
- AmGetComputeString
- AmGetCurrentNTDomain
- AmGetCurrentNTUser
- AmGetFeat
- AmGetFeatCount
- AmGetField
- AmGetFieldCount
- AmGetFieldDateOnlyValue
- AmGetFieldDateValue
- AmGetFieldDescription
- AmGetFieldDoubleValue
- AmGetFieldFormat
- AmGetFieldFormatFromName
- AmGetFieldFromName
- AmGetFieldLabel
- AmGetFieldLabelFromName
- AmGetFieldLongValue
- AmGetFieldName
- AmGetFieldRights
- AmGetFieldSize
- AmGetFieldSqlName
- AmGetFieldStrValue
- AmGetFieldType
- AmGetFieldUserType
- AmGetForeignKey
- AmGetIndex
- AmGetIndexCount
- AmGetIndexField
- AmGetIndexFieldCount
- AmGetIndexFlags
- AmGetIndexName
- AmGetLink
- AmGetLinkCardinality
- AmGetLinkCount
- AmGetLinkDstField
- AmGetLinkFeatureValue
- AmGetLinkFromName
- AmGetLinkType
- AmGetMainField
- AmGetMemoField
- AmGetNextAssetPin
- AmGetNextAssetPort
- AmGetNextCableBundle
- AmGetNextCablePair
- AmGetNTDomains
- AmGetNTMachinesInDomain
- AmGetNTUsersInDomain
- AmGetPOLinePrice
- AmGetPOLinePriceCur
- AmGetPOLineReference
- AmGetRecordFromMainId
- AmGetRecordHandle
- AmGetRecordId
- AmGetRelDstField
- AmGetRelSrcField
- AmGetRelTable
- AmGetReverseLink
- AmGetScriptValue
- AmGetSelfFromMainId
- AmGetSourceTable
- AmGetTable
- AmGetTableCount
- AmGetTableDescription

- AmGetTableFromName
- AmGetTableLabel
- AmGetTableName
- AmGetTableRights
- AmGetTableSqlName
- AmGetTargetTable
- AmGetTrace
- AmGetTraceFromHist
- AmGetTypedLinkField
- AmGetUserEnvSessionItem
- AmGetVersion
- AmHasAdminPrivilege
- AmHasRelTable
- AmHasRightsForCreation
- AmHasRightsForDeletion
- AmHasRightsForFieldUpdate
- AmHelpdeskCanCloseFile
- AmHelpdeskCanProceed
- AmHelpdeskCanSaveCall
- AmImportDocument
- AmImportReport
- AmIncrementLogLevel
- AmInsertRecord
- AmInstantiateReqLine
- AmInstantiateRequest
- AmIsConnected
- AmIsFieldForeignKey
- AmIsFieldIndexed
- AmIsFieldPrimaryKey
- AmIsHelpdeskAdmin
- AmIsHelpdeskMember
- AmIsHelpdeskSuper
- AmIsLink
- AmIsModuleAuthorized
- AmIsTypedLink
- AmLastError
- AmLastErrorMsg
- AmListToString
- AmLog
- AmLoginId
- AmLoginName
- AmMapSubReqLineAgent
- AmMoveCable
- AmMoveDevice
- AmMsgBox
- AmOpenConnection
- AmOpenScreen
- AmOverflowTables
- AmPagePath
- AmProgress
- AmPurgeRecord
- AmQueryCreate
- AmQueryExec
- AmQueryGet
- AmQueryNext
- AmQuerySetAddMainField
- AmQuerySetFullMemo
- AmQueryStartTable
- AmQueryStop
- AmReceiveAllPOLines
- AmReceivePOLine
- AmRefreshAllCaches
- AmRefreshLabel
- AmRefreshProperty
- AmRefreshTraceHist
- AmReleaseHandle
- AmRemoveCable
- AmRemoveDevice
- AmResetPassword
- AmResetUserEnvSession
- AmResetUserPassword
- AmRestoreRecord
- AmReturnAsset
- AmReturnContract
- AmReturnPortfolioItem
- AmReturnTraining

- AmReturnWorkOrder
- AmRevCryptPassword
- AmRgbColor
- AmRollback
- AmSetFieldDateOnlyValue
- AmSetFieldDateValue
- AmSetFieldDoubleValue
- AmSetFieldLongValue
- AmSetFieldStrValue
- AmSetLinkFeatureValue
- AmSetProperty
- AmSetUserEnvSessionItem
- AmShowCableCrossConnect
- AmShowDeviceCrossConnect
- AmSqlTextConst
- AmStandIn
- AmStandInGroup
- AmStartTransaction
- AmStartup
- AmTableDesc
- AmTaxRate
- AmUpdateDetail
- AmUpdateLossLines
- AmUpdateRecord
- AmUpdateUser
- AmValueOf
- AmWizChain
- AmWorkTimeSpanBetween
- AppendOperand
- ApplyNewVals
- Asc
- Atn
- BasicToLocalDate
- BasicToLocalTime
- BasicToLocalTimeStamp
- Beep
- CDbl
- ChDir
- ChDrive
- Chr
- CInt
- CLng
- Cos
- CountOccurences
- CountValues
- CSng
- CStr
- CurDir
- CVar
- Date
- DateAdd
- DateAddLogical
- DateDiff
- DateSerial
- DateValue
- Day
- EnumToComboBox
- EscapeSeparators
- ExeDir
- Exp
- ExtractValue
- FileCopy
- FileDateTime
- FileExists
- FileLen
- Fix
- FormatDate
- FormatResString
- FV
- GetEnvVar
- GetListItem
- Hex
- Hour
- InStr
- Int
- IPMT

- IsNumeric
- Kill
- LCase
- Left
- LeftPart
- LeftPartFromRight
- Len
- LocalToBasicDate
- LocalToBasicTime
- LocalToBasicTimeStamp
- LocalToUTCDate
- Log
- LTrim
- MakeInvertBool
- Mid
- Minute
- MkDir
- Month
- Name
- Now
- NPER
- Oct
- ParseDate
- ParseDMYDate
- ParseMDYDate
- ParseYMDDate
- PMT
- PPMT
- PV
- Randomize
- RATE
- RemoveRows
- Replace
- Right
- RightPart
- RightPartFromLeft
- RmAllInDir
- RmDir
- Rnd
- RoundValue
- RTrim
- Second
- SetMaxInst
- SetSubList
- Sgn
- Shell
- Sin
- Space
- Sqr
- Str
- StrComp
- String
- SubList
- SysEnumToComboBox
- Tan
- Time
- Timer
- TimeSerial
- TimeValue
- ToSmart
- Trim
- UCase
- UnEscapeSeparators
- Union
- UTCToLocalDate
- Val
- WeekDay
- Year

使用可能な関数 - データの変換 - 計算の実行

- AmBusinessSecondsInDay
- AmCalcConsolidatedFeature
- AmConvertDateBasicToUnix
- AmConvertDateIntlToUnix
- AmConvertDateStringToUnix
- AmConvertDateUnixToBasic
- AmConvertDateUnixToIntl
- AmConvertDateUnixToString
- AmConvertDoubleToString
- AmConvertMonetaryToString
- AmConvertStringToDouble
- AmConvertStringToMonetary
- AmCounter
- AmCryptPassword
- AmDateAdd
- AmDateAddLogical
- AmDateDiff
- AmDbGetDate
- AmDbGetDouble
- AmDbGetList
- AmDbGetListEx
- AmDbGetLong
- AmDbGetPk
- AmDbGetString
- AmDbGetStringEx
- AmDeadLine
- AmEndOfNthBusinessDay
- AmEnumValList
- AmFormatLong
- AmGenSqlName
- AmGetComputeString
- AmListToString
- AmRevCryptPassword
- AmSqlTextConst
- AmTableDesc
- AmWorkTimeSpanBetween
- EnumToComboBox
- SysEnumToComboBox

使用可能な関数 - 情報の取得

- AmBuildNumber
- AmConnectionName
- AmCurrentDate
- AmCurrentIsoLang
- AmCurrentLanguage
- AmCurrentServerDate
- AmGetCurrentNTDomain
- AmGetCurrentNTUser
- AmGetFeat
- AmGetFeatCount
- AmGetField
- AmGetFieldCount
- AmGetFieldDateOnlyValue
- AmGetFieldDateValue
- AmGetFieldDescription
- AmGetFieldDoubleValue
- AmGetFieldFormat
- AmGetFieldFormatFromName
- AmGetFieldFromName
- AmGetFieldLabel
- AmGetFieldLabelFromName
- AmGetFieldLongValue
- AmGetFieldName
- AmGetFieldRights
- AmGetFieldSize
- AmGetFieldSqlName
- AmGetFieldStrValue
- AmGetFieldType
- AmGetFieldUserType
- AmGetForeignKey
- AmGetIndex
- AmGetIndexCount
- AmGetIndexField
- AmGetIndexFieldCount
- AmGetIndexFlags
- AmGetIndexName
- AmGetLink
- AmGetLinkCardinality
- AmGetLinkCount
- AmGetLinkDstField
- AmGetLinkFeatureValue
- AmGetLinkFromName
- AmGetLinkType
- AmGetMainField

- AmGetMemoField
- AmGetNTDomains
- AmGetNTMachinesInDomain
- AmGetNTUsersInDomain
- AmGetRecordFromMainId
- AmGetRecordHandle
- AmGetRecordId
- AmGetRelDstField
- AmGetRelSrcField
- AmGetRelTable
- AmGetReverseLink
- AmGetSelfFromMainId
- AmGetSourceTable
- AmGetTable
- AmGetTableCount
- AmGetTableDescription
- AmGetTableFromName
- AmGetTableLabel
- AmGetTableName
- AmGetTableRights
- AmGetTableSqlName
- AmGetTargetTable
- AmGetTypedLinkField
- AmGetVersion
- AmHasAdminPrivilege
- AmHasRelTable
- AmHasRightsForCreation
- AmHasRightsForDeletion
- AmHasRightsForFieldUpdate
- AmIsConnected
- AmIsFieldForeignKey
- AmIsFieldIndexed
- AmIsFieldPrimaryKey
- AmIsLink
- AmIsModuleAuthorized
- AmIsTypedLink
- AmLastError
- AmLastErrorMsg
- AmLoginId
- AmLoginName
- AmOverflowTables
- AmPagePath
- AmProgress
- AmQueryNext
- AmQueryStartTable
- AmRgbColor
- AmValueOf

使用可能な関数 - AssetCenterでの内部動作のトリガ

- AmCleanup
- AmClearLastError
- AmCloseAllChildren
- AmCloseConnection
- AmDbExecAql
- AmDecrementLogLevel
- AmEvalScript
- AmExecTransition
- AmExecuteActionById
- AmExecuteActionByName
- AmExportDocument
- AmExportReport
- AmGeneratePlanningData
- AmIncrementLogLevel
- AmLog
- AmMsgBox
- AmOpenConnection
- AmOpenScreen
- AmQueryExec
- AmQueryGet
- AmQuerySetAddMainField
- AmQuerySetFullMemo
- AmQueryStop
- AmRefreshAllCaches
- AmRefreshProperty
- AmReleaseHandle
- AmStartup
- AmUpdateDetail
- AmWizChain

使用可能な関数 - 「ファイナンス」モジュール

- AmCalcDepr
- AmCbkReplayEvent
- AmConvertCurrency
- AmDefaultCurrency
- AmFormatCurrency
- AmTaxRate

使用可能な関数 - データベースのデータの変更

- AmArchiveRecord
- AmBackupRecord
- AmCommit
- AmCreateLink
- AmCreateRecord
- AmDeleteLink
- AmDeleteRecord
- AmDuplicateRecord
- AmFlushTransaction
- AmImportDocument
- AmImportReport
- AmInsertRecord
- AmPurgeRecord
- AmRestoreRecord
- AmRollback
- AmSetFieldDateOnlyValue
- AmSetFieldDateValue
- AmSetFieldDoubleValue
- AmSetFieldLongValue
- AmSetFieldStrValue
- AmSetLinkFeatureValue
- AmSetProperty
- AmStartTransaction
- AmUpdateRecord
- AmUpdateUser

使用可能な関数 - 「調達」モジュール

- AmAddAllPOLinesToInv
- AmAddCatRefAndCompositionToPOrder
- AmAddCatRefToPOrder
- AmAddEstimLinesToPO
- AmAddEstimLineToPO
- AmAddLicContentToRequest
- AmAddPOLineToInv
- AmAddPOrderLineToReceipt
- AmAddReceiptLineToInvoice
- AmAddReqLinesToEstim
- AmAddReqLinesToPO
- AmAddReqLineToEstim
- AmAddReqLineToPO
- AmAddRequestLineToPOrder
- AmAddTemplateToPOrder
- AmAddTemplateToRequest
- AmCalculateCatRefQty
- AmCalculateReqLineQty
- AmCreateAssetsAwaitingDelivery
- AmCreateDelivFromPO
- AmCreateEstimFromReq
- AmCreateEstimsFromAllReqLines
- AmCreateInvFromPO
- AmCreateOrUpdateInvoiceFromReceipt
- AmCreatePOFromEstim
- AmCreatePOFromReq
- AmCreatePOrderFromRequest
- AmCreatePOrdersFromRequest
- AmCreatePOsFromAllReqLines
- AmCreateReceiptFromPOrder
- AmCreateRequestToInvoice
- AmCreateRequestToPOrder
- AmCreateRequestToReceipt
- AmCreateReturnFromReceipt
- AmGetCatRef
- AmGetCatRefFromCatProduct
- AmGetPOLinePrice
- AmGetPOLinePriceCur
- AmGetPOLineReference
- AmInstantiateReqLine
- AmInstantiateRequest
- AmMapSubReqLineAgent
- AmReceiveAllPOLines
- AmReceivePOLine

- **AmReturnAsset**
- **AmReturnContract**
- **AmReturnPortfolioItem**
- **AmReturnTraining**
- **AmReturnWorkOrder**

使用可能な関数 - 「契約」モジュール

- AmUpdateLossLines

使用可能な関数 - 「ケーブル」モジュール

- AmCheckTraceDone
- AmConnectTrace
- AmCreateAssetPort
- AmCreateCable
- AmCreateCableBundle
- AmCreateCableLink
- AmCreateDevice
- AmCreateDeviceLink
- AmCreateProjectCable
- AmCreateProjectDevice
- AmCreateProjectTrace
- AmCreateTraceHist
- AmCreateTraceLink
- AmDisconnectTrace
- AmFindCable
- AmFindDevice
- AmFindRootLink
- AmFindTermDevice
- AmFindTermField
- AmGetNextAssetPin
- AmGetNextAssetPort
- AmGetNextCableBundle
- AmGetNextCablePair
- AmGetTrace
- AmGetTraceFromHist
- AmMoveCable
- AmMoveDevice
- AmRefreshLabel
- AmRefreshTraceHist
- AmRemoveCable
- AmRemoveDevice
- AmShowCableCrossConnect
- AmShowDeviceCrossConnect

使用可能な関数 - 'ソフトウェア配布'モジュール

- AmESDAddComputers
- AmESDCreateTask

使用可能な関数 - 「ポートフォリオ」モジュール

- AmStandIn
- AmStandInGroup

使用可能な関数 - AssetCenterからの外部動作のトリガ

- AmActionDde
- AmActionExec
- AmActionMail
- AmActionPrint
- AmActionPrintPreview
- AmActionPrintTo