HP OpenView AssetCenter

软件版本: 5.0

程序员参考

Build号: 328

法律声明

担保

HP 产品和服务的所有担保已在随产品和服务提供的担保声明中阐明。
声明中没有内容构成附加担保条款。
对于其中包含的任何技术、编辑错误或遗漏，HP 概不负责。
此处包含的信息如有更改，恕不另行通知。

受限权利

保密计算机软件
必须有从 HP 获得的有效许可证才能拥有、使用或复制。
根据 FAR 12.211 和 12.212，商业计算机软件、计算机软件文档和商业项目的技术数据已根据供应商标准商业许可条款，授权给美国政府。

商标声明

- 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.

目录

I 简介

1 编程基础知识

在 AssetCenter 中可以使用 Basic 语言，本章介绍使用 Basic 语言编程的基础知识。如果您有编程的经验，并且使用过其他语言，那么本章介绍的大多数内容您都会很熟悉。但是，由于在 AssetCenter 中 Basic 的某些经典功能已被刻意剔除或受到限制，因此建议您仍然通读本章内容。

变量简介

变量用于在程序执行过程中存储数据。变量的识别是通过：

- 变量名，用于引用变量中包含的值。
- 变量类型，决定了变量中可以存储的数据类型。

通常，变量可分为两种截然不同的类型：

- 数组，
- 标量变量，包括所有非数组的变量。

声明变量

在使用变量之前必须先显式声明变量。声明所用的语法如下：

```
Dim <变量名> [As <变量类型>]
```

注:

AssetCenter Basic 中变量的显式声明与在 Microsoft Visual Basic 中使用 Option Explicit 关键字进行声明的方法相同。

变量名必须满足以下条件：

- 以大写或小写字母开头，
- 不能超过 40 个字符，
- 可以包含字母 A 到 Z 和 a 到 z，数字 0 到 9 以及下划线 ("_")。

注:

可以使用撇号，但建议尽量避免使用。

- 不可以使用保留关键字。例如，Basic 函数的名称和子句都是保留关键字。

使用可选的 As 子句，可以定义已定义变量的类型。类型规定了变量中存储的信息的类型。可用的数据类型包括：String、Integer、Variant...

如果省略 As 子句，则该变量被视为 Variant 类型。

单一声明

单一声明时，每个声明语句只包含一个变量，如下例所示：

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

复合声明

复合声明时，每个声明语句可以包含任意多个变量，如下例所示：

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

注:

如前所述，未指定变量的类型时，变量默认类型为Variant。因此，在上例的第二行中，变量 A 和 B 的类型是 Variant，C 的类型是 Integer。

数据类型

下表归纳了函数或参数可用的各种类型：

类型	含义
Integer	整数（从 -32,768 到 +32,767）。
Long	整数（从 -2,147,483,647 到 +2,147,483,646）。
Single	4 字节浮点型数值（单精度）。
Double	8 字节浮点型数值（双精度）。
String	接受所有字符的文本。
Date	日期或日期与时间
Variant	可以代表任何类型的通用类型。

注:

这些类型在外部工具中不可用。只有长整型、双精度和字符串类型的变量可用。变量不存在，整型和日期类型对象使用长整型表示。

数值类型

AssetCenter 中的 Basic 语言提供了多种数值类型：Integer、Long、Single 和 Double。数值数据类型使用的内存通常小于变量类型数据使用的内存。

如果您确定变量将系统地存储整数（例如 123）而不存储小数（例如 3.14），最好将其声明为整型或长整型。这些数据类型与其他数据类型相比，执行速度快，所需内存小。这些数据类型特别适合于循环中使用的计数器。

如果变量必须包含小数，则应声明为单精度或双精度。

注:

浮点数（单精度或双精度）可能会出现舍入错误。

字符串类型

如果确定变量只存储字符串，则将其声明为 String：

```
Dim MyString As String
```

然后就可以在该变量中存储字符串，并使用专门的字符串处理函数对其内容进行操作：

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

默认情况下，字符串类型变量的大小可变。分配用于存储字符串的大小根据指定给变量的数据的大小而定。但是，可以使用以下语法声明字符串类型变量：

```
Dim <变量名> As String * <存储字符串的大小>
```

下例声明了一个包含 20 个字符的变量：

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

如果使用此变量存储小于 20 个字符的字符串，则会在字符串的末尾添加空格，直到足够 20 个字符。相反，如果存储的字符串超过 20 个字符，则该字符串将从第 21 个字符开始被截断。

变量类型

变量类型是可代替其他所有类型的通用类型。无需担心其他数据类型和变量类型之间的转换问题。转换是自动执行的，如下例所示：

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

尽管转换是自动的，但仍须遵守以下规则：

- 如果要对变量类型变量执行算术运算，即使其是由字符串表示的，也必须包含一个数字。
- 如果连接操作中包括变量，则需使用 &; 运算符而不是 +。

变量还可以包含两类特殊值：空值和 Null 值。

空值

在首次为变量赋值之前，变量包含的是空值。空值是一个特殊的值，不同于 0、空字符串或 Null 值。要测试变量是否包含空值，可使用 Basic 函数 **IsEmpty()**，如下例所示：

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

在表达式中可以使用包含空值的变量。根据具体情况，该变量将被作为 0 值或空字符串处理。要重新为变量分配空值，可使用关键字 Empty，如下例所示：

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

Null 值

在数据库中常常使用 Null 值来指定缺少或未知的值。Null 值具有一些特殊性质：

- 包含 Null 值的表达式总是返回 Null 值。Null 值可在表达式中传播。如果表达式的某个部分是 Null，则整个表达式也为 Null。
- 一般地，如果某个函数参数被设置为 Null，则该函数将返回 Null 值。

数据数组

使用数组可以利用一个变量名存储和引用一组变量，用一个数字（索引）唯一标识这些变量。所有数组项的数据类型必须相同。不能创建同时包含字符串和双精度值的数组。可使用变量类型解决此限制问题。

声明数组

数组是一组变量。

按照约定，数组采用以下方法表示：

- 数组的下标：第一项的索引。

  注:

  默认情况下，数组的下标为 0。

- 数组的上标：最后一项的索引。

  注:

  数组的上标不能超过长整型的大小（2,147,483,646 项）。

数组的声明方法与变量的声明类似：

```
Dim <数组的名称>(<数组的上标>) [As <数组中包含的变量的数据类型>]
```

示例：

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

也可以通过下面的声明指定数组的下标：

```
Dim <数组的名称>(<数组的下标> To <数组的上标>) [As <数组中包含的变量的数据类型>]
```

示例：

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

限制

在 AssetCenter Basic 中，对数组有以下限制：

- 不支持可变大小的数组。尤其是不可实时更改数组的大小。
- 不支持多维数组。

控制结构

顾名思义，控制结构可控制程序的执行。控制结构有两类：

- 决策结构：根据一定的条件重新定向和引导程序的执行
- 循环结构：根据一定的条件重复执行程序段。

决策结构

决策结构根据测试的结果有条件地执行指令。存在以下决策结构：

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

If...Then

使用此结构可以在一定条件下执行一条或多条指令。此结构的语法适用于一行或多行语句。单行语句只能执行一条指令：

```
If <条件> Then <指令>
```

```
If <条件> Then <指令> End If
```

这里的条件通常是一个比较关系式，但也可以使用可返回数值结果的任何表达式。然后，Basic 将该值解释为 **True** 或 **False**。**False** 对应于数值 0，所有其他值均被视为 **True**。

如果条件的结果为 **True**，则将执行指令或关键字 **Then** 之后的指令。

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

使用此结构可以定义多个条件指令块。这些块中将只执行第一个所得结果为 **True** 的块。

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

将首先测试第一个条件，如果其所得结果为 **False**，则会继续测试第二个条件，依此类推，直至其中一个条件的所得结果为 **True**。将执行关键字 **Then** 之后的指令集。

关键字 **Else** 是可选的。使用 Else 关键字，可以定义在所有条件的计算结果均为 **False** 时要执行的指令集。

注:

可以在决策结构中嵌套任意多个 **ElseIf** 指令。但是，如果系统地将相同的表达式与不同的值进行比较，那么决策结构的语法可能会变得无谓地复杂、难于理解。在这种情况下，我们建议您使用 **Select...Case** 类型的决策结构。

Select...Case

这种结构的作用与前面的决策结构相同，但总体上讲，这种结构的代码更具可读性。Select...Case 函数在结构的开头执行一项测试，并将测试结果与结构中每个 Case 给定的值进行比较。如果出现匹配的情况，则将执行与该 Case 关联的指令集。

```
Select Case <测试> [Case <值列表 1> <指令1>] [Case <值列表 2> <指令2>] ...[Case Else <指令n>] End Select
```

每个值列表都包含由逗号分割的值列表。如果有多个 Case 关键字的值与测试结果匹配，只执行与第一个匹配的 Case 关联的指令。

如果没有找到与 Case 关键字相匹配的项，则执行与 Case Else 关键字关联的指令。

循环结构

利用循环结构可重复执行一系列指令。存在以下循环结构：

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

Do...Loop

使用这种结构可以无限次地执行一系列指令。在满足条件或不满足条件时，退出循环。此条件是一个值或者是一个计算结果为 False（零）或 True（非零）的表达式。

注:

在执行指令中使用 Exit Do 关键字可以强制退出循环。

这种结构有多种变体形式，最常见的形式如下：

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

在这种情况下，总是先计算条件。如果为 True，则将执行指令，程序将返回 Do While 关键字，再次测试条件，依此类推。当条件的计算结果为 False 时，将退出循环。

下例测试计数器的值，每次循环，计数器都会 +1。当计数器达到 20 时，执行循环。

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

下面的示例以前面的例子为基础，但使用 Exit Do 关键字在计数器包含值 10 时强制退出循环。

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

在 Do...Loop 结构的这一类型中，条件的计算发生在指令执行之前。如果希望先执行指令，然后测试条件，则可使用 Do...Loop 结构：

```
Do <指令> Loop While <条件>
```

注:

这种类型的结构保证了指令至少会被执行一次。

前面两个 Do...Loop 结构，只要条件为 True 就会迭代执行指令。如果希望当条件为 False 时进行迭代，则可使用以下结构之一：

```
Do Until <条件> <指令> Loop

Do <指令> Loop Until <条件>
```

使用这种结构类型，上面的例子可以写成：

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

For...Next

使用这种结构可以无限次地执行一系列指令。与 Do...Loop 不同，For...Next 循环使用名为计数器的变量，每次迭代时，变量的值都会递增或递减。

注:

在执行指令中使用 Exit For 关键字可以强制退出循环。

```
For <计数器> = <初始值> To <终值> [Step <递增>] <指令> Next [<计数器>]
```

重要:

参数计数器、初始值、终值**和**递增**均为数值。**

注:

Increment 可以是正值也可以是或负值。如果为正，要执行指令，初始值**必须小于或等于**终值。**如果为负，要执行指令，**初始值**必须大于或等于**终值。**如果未指定**递增，则默认值为** 1。

执行 For...Next 循环时，将执行以下操作：

1. **计数器初始化并存储初始值，**
2. Basic **代码测试计数器的值是否大于终值。如果大于终值，程序将退出循环。**

   注:

   如果递增为负数，Basic **将测试计数器的值是否小于终值。**

3. **执行指令，**
4. **计数器递增** 1 **或递增指定的值，**
5. **重复执行操作** 2 **到** 4。

以下操作对 1000 **以内的所有偶数求和：**

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

下面的示例以前面的例子为基础，但使用 Exit For 关键字在计数器包含值 500 **时强制退出循环。**

```
Dim iCounter As Integer, lSum As Long
For iCounter = 0 To 1000 Step 2
lSum = lSum + iCounter
If iCounter = 500 Then Exit For
Next
```

运算符

运算符是一种用于对变量执行简单运算（加法、乘法等）或对变量进行比较的符号。有以下几种不同类型的运算符：

- **赋值运算符**
- **算术运算符**
- **关系运算符（也称为赋值运算符）**
- **逻辑运算符**

赋值运算符

使用这种运算符可以为变量赋值。AssetCenter Basic 只使用一个赋值运算符，即等号 "="。赋值的语法如下所示：

```
<变量> = <值>
```

算术运算符

使用算术运算符，可以利用算术修改变量的值，或者对两个表达式执行简单的算术运算。

加法运算符 (+)

使用此运算符，可以将两个值相加。语法如下所示：

```
<结果> = <表达式 1> + <表达式 2>
```

注:

此运算符既可用于将两个值相加，也可用于连接字符串。为避免混淆，建议只使用此运算符进行求和操作，而使用 &; 运算符连接字符串。

减法运算符 (-)

使用此运算符，可以将两个值相减或者对一个值取负（一元运算符）。此运算符有两种语法结构：

```
<结果> = <表达式 1> - <表达式 2>
```

或

```
- <表达式>
```

乘法运算符 (*)

使用此运算符，可以将两个值相乘。语法如下：

```
<结果> = <表达式 1> * <表达式 2>
```

除法运算符 (/)

使用此运算符，可以将两个值相除。语法如下所示：

```
<结果> = <表达式 1> / <表达式 2>
```

幂运算符 (^)

使用此运算符可对一个值进行幂运算。语法如下所示：

```
<结果> = <表达式 1> ^ <表达式 2>
```

注:

在此语法中，如果表达式 2（指数）是整数，则表达式 1 不能为负数。当表达式要连续执行多个幂运算时，则各个运算的解释顺序是从左到右。

模运算符 (Mod)

此运算符将两个值进行整除操作之后返回其余数。语法如下所示：

```
<结果> = <表达式 1> Mod <表达式 2>
```

注:

浮点数自动舍入为整数。

下例将返回 4（6.8 舍入为最接近的整数 7）：

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

关系运算符

使用关系运算符，可以将两个值进行比较。下表归纳了关系运算符：

运算符	名称	描述	语法
=	等于运算符	判断两个操作数是否相等	<表达式 1> = <表达式 2>
<	小于运算符	判断一个值是否确实小于另一个值	<表达式 1> < <表达式 2>
<=	小于或等于运算符	判断一个值是否小于或等于另一个值	<表达式 1> <= <表达式 2>
>	大于运算符	判断一个值是否确实大于另一个值	<表达式 1> > <表达式 2>
>=	大于或等于运算符	判断一个值是否大于或等于另一个值	<表达式 1> >= <表达式 2>
<>	不等于运算符	判断一个值是否与另一个值不相等	<表达式 1> <> <表达式 2>

逻辑运算符

使用逻辑运算符，可以对多个条件进行逻辑判断。

与运算符 (And)

此运算符对两个表达式执行逻辑与（条件必须都为真）运算。语法如下所示：

```
<结果> = <表达式 1> And <表达式 2>
```

如果每个表达式（操作数）的判断结果为 True**，则其结果为**True**。如果有一个表达式的计算结果为** False**，则其结果为** False**。**

或运算符 (Or)

此运算符对两个表达式执行逻辑或（必须有一个条件为真）运算。语法如下所示：

```
<结果> = <表达式 1> Or <表达式 2>
```

如果有一个表达式（操作数）的判断结果为 True**，则其结果为** True**。**

异或运算符 (Xor)

此运算符对两个表达式执行异或（两个条件中有且只有一个为真）运算。语法如下所示：

```
<结果> = <表达式 1> Xor <表达式 2>
```

如果有且只有一个表达式（操作数）的计算结果为 True**，则其结果为** True**。**

非运算符 (Not)

此运算符用于对某个表达式执行逻辑非运算。语法如下所示：

```
<结果> = Not <表达式 1>
```

如果该表达式的判断结果为 True**，则其结果为** False**。如果该表达式的判断结果为** False**，则其结果为** True**。**

运算符的优先级

当混合使用一个以上的运算符时，在判断表达式时将使用下面的优先级顺序。运算符按优先级的降序排列如下：

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

文件管理

AssetCenter Basic 支持简单的文件管理。支持标准的常见操作（读取、写入等）。

文件概念

文件是一种信息存储形式，程序及外部对象借此可以查看文件包含的数据。文件是逻辑记录的集合，可能存在也可能不存在一定的结构，程序可以对其执行一系列基本操作（读取、写入等）。逻辑记录代表了单个基本操作所可处理的最小数据集。

AssetCenter 只能处理所谓的顺序文件。在顺序文件中，操作主要是读取下一个记录或者在文件结尾追加新记录。不能同时读写记录。

读取记录时，光标位于顺序文件中的第一条逻辑记录上。每个读取操作将把记录传输到程序的内部区域（通常是一个变量）并将光标置于文件的下一记录上。有一个操作可用于判断是否有剩余的要读取的记录（EOF 子句）。

写入时，要么顺序文件为空，要么光标位于文件的最后一个记录之后。每个写入操作将程序内部区域（通常是一个变量）存储的数据传输到文件的记录中，然后将光标移动到此记录之后。

注:

顺序文件的一个主要特点是记录将按照写入顺序读取。

打开和关闭文件

Open 子句

此子句是处理文件所用的主要子句。使用此子句，可以读取、创建和写入文件。语法如下所示：

```
Open <文件路径> For <模式> [Access <访问类型>] As [#]<文件编号>
```

下表中详细列出了该子句的参数：

参数	描述
<文件路径>	指定操作所涉及文件的字符串。此字符串可以包含文件的完整路径。

参数	描述
<模式>	指定文件的处理模式。此参数可以包含下列值之一： ■ Input：文件以读取模式打开。 ■ Output：文件以写入模式打开。如果文件已存在并且已有内容，则将被覆盖。 ■ Append：文件以写入模式打开。如果文件已存在并且已有内容，则新内容将追加到文件末尾。 ■ Binary：文件以二进制读取模式打开。
<访问类型>	指定对已打开文件所能执行的操作。如果该文件已由其他进程打开，并且未被授予指定的访问权限，该文件打开命令将会失败。此参数可被设置为下列值之一： ■ Read：文件以只读访问权限打开 ■ Write：文件以只写访问权限打开 ■ Read Write：文件以读写模式打开。此访问类型只适用于 Binary 和 Append 访问模式。
<文件编号>	使用 1 和 511 之间的唯一数字标识文件。使用 FreeFile() 函数可以确定下一可用的文件编号。

注:

请牢记以下几点：

- 在对文件进行读取或写入操作之前，必须先使用 Open 子句打开该文件。
- 在 Append、Binary 或 Output 模式下，如果引用的文件不存在，则会创建文件。
- 在 Binary 或 Input 模式下，无需先关闭该文件，既打开其他编号的文件。在 Append 或 Output 模式下，必须先关闭文件，才能再打开其他编号的文件。

Close 子句

使用该子句，可以关闭使用 Open() 打开的文件。语法如下所示：

```
Close [<文件列表>]
```

可选的 <文件列表> 参数可以包含一个或多个文件编号。此可选参数的语法如下：

```
[[#]<文件编号>][,[#]<文件编号>]...
```

注:

如果在该子句中省略此参数，则将关闭使用 Open() 子句打开的所有活动文件。

从文件中读取数据

有两个子句可用于从文件中读取数据。使用哪一个子句要取决于为该文件指定的访问模式。这两个子句分别是：

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

Input 子句

此子句用于从以 Binary 或 Input 模式打开的文件中读取指定数目的字符。此子句的语法如下：

```
Input (<要读取的字符数>,[#]<文件编号>)
```

Line Input 子句

此子句用于从顺序文件中读取一行数据，并将其存储在 String 或 Variant 类型的变量中。此子句的语法如下：

```
Line Input #<文件编号>, <变量名>
```

重要:

此子句逐一读取字符，直至遇到回车或者遇到回车 - 新行。

向文件写入数据

只需使用一个 **Print** 子句，即可向文件写入数据。此子句的语法如下：

```
Print #<文件编号>, [<数据>]
```

2 函数的分类

函数可按照三种级别标准分类。对于给定的函数，可按照以下标准分类：

函数分类

AssetCenter 环境中的函数可归入几大类中：

- AssetCenter 可识别的函数：这些函数是可在软件的脚本部分（在 Basic 中）使用的基本函数。
- AssetCenter API 可识别的函数：可被外部工具调用的这些函数，或者是高级语言编写的程序。

函数的这几大分类并不是绝对的。例如，某些 AssetCenter API 函数就可在软件的 Basic 脚本中使用。此类源自 AssetCenter API 的函数在 AssetCenter 内部 Basic 脚本中是“公开”的。此类函数的语法可能会有变动，但其行为保持不变。

函数的作用域

此文档中介绍的函数至少可用于以下上下文之一：

- AssetCenter API 库。特别地，这些函数可用于 Get-It 应用程序的开发。

- **字段或链接配置脚本**（配置对象**弹出菜单项或** AssetCenter Database Administrator）**以及计算字段的扩展**计算脚本（SQL **名称：**memScript）**：**
  - **默认值，**
  - **强制属性，**
  - **历史记录，**
  - **只读属性，**
  - ...
- **脚本类型操作：**
  - **在脚本操作的**操作脚本（SQL **名称：**Script）**中定义的脚本。**
- AssetCenter **向导：**
  - **向导的** "FINISH.DO" **脚本。**
  - **节点属性的值定义脚本。**

应用程序模块

每个函数都与一个或多个应用程序模块相关联。应用程序模块说明了函数执行的操作的性质。下面列出了各种不同的应用程序模块：

- **内置模块：传统的** Basic **函数，转换和字符串处理函数，等等。**
- **技术模块：连接到数据库，处理表、字段、链接、索引、记录和查询对象。**
- **功能模块：通用的业务范围函数。**
- **电缆**
- **采购**
- **费用分摊**
- **向导**
- **操作**
- **图形**

3 约定

本章介绍：

符号

本手册的示例中使用了以下符号：

[]	方括号表示可选的参数。在命令中不要键入方括号。 例外：在 Basic 脚本中，当方括号表示数据库中数据的路径时，脚本中必须为以下形式： [链接.链接.字段]
<>	尖括号使用文本语言表示参数。不要键入这些括号。请使用恰当的信息替换该文本。
{}	花括号中包含节点的定义或属性跨多行的脚本块。
|	竖线用于分隔花括号内包含的一系列可能的参数。

下面的文本样式具有特殊的含义：

固定宽度字符	DOS 命令、函数参数为数据格式化。
示例	代码或命令的示例。

...	省略的代码或命令。
对象名称	字段、选项卡、菜单和文件的名称以粗体显示。

日期和时间常量在脚本中的格式

不管用户如何定义显示选项，脚本中引用的日期都是以国际格式表示的：

yyyy/mm/dd hh:mm:ss

示例：

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

注:

连字符"-"也可用作日期分隔符。

关于日期

日期在 Basic 内部和通过外部工具表示方法并不相同：

- 在 Basic 中，日期采用国际格式表示，或者以浮点数（双精度类型）表示。在这种情形下，数字的整数部分表示自从 1899 年 12 月 30 日零点至今已过去的天数，小数部分表示当前日期剩余时间（该日开始后过去的秒数除以 86400）。
- 在外部，日期表示为长整型（Long 类型），表示自 1970 年 1 月 1 日零点起过去的秒数，它与时区（UTC 时间）无关。

持续时间类型常量在脚本中的格式

在脚本中，持续时间(Duration 类型)以秒为单位进行存储和表示。例如，要将"持续时间"类型字段的默认值设置为 3 天，可使用以下脚本：

```
RetVal=259200
```

同样，计算持续时间的函数，例如 AmWorkTimeSpanBetween() 函数返回的是秒数。

注:

在财务计算中，AssetCenter 采用的是最常见的简化计算。只是在这种情形下，一年才被视作 12 个月，1 月视作 30 天（因此：1 年 = 360 天）。

4 定义

本章系统地介绍几个关键术语的定义。

包括以下定义：

函数的定义

函数是指执行操作并将值返回给用户的程序。此值称为“返回值”或者“返回代码”。

下面是调用 AssetCenter 内部函数所用的语法示例：

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

下面是通过 AssetCenter API 调用同一函数的语法：

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

CurrentUser 虚拟链接的定义

定义

CurrentUser 可被视作这样的链接：可在所有表中启动，指向当前用户在部门和员工表中对应的记录。

- **在 CurrentUser 格式中，它指向与当前用户对应的记录，返回来自"员工"和"部门"表中的描述字符串。**
- **在 CurrentUser.Field 格式中，返回当前用户的字段值。**

注:

在字段和链接列表中不会显示此虚拟链接；因此在 AssetCenter **的内部脚本生成器中无法直接访问它。必须手动输入此表达式。**

等价函数

AmLoginName() 和 AmLoginId() 函数，分别返回当前用户的名称（**SQL** 名称：**Name** ）**和** ID（SQL **名称：**lPersId），**可被视作从 CurrentUser 派生的函数。下面两个表达式在效果上是等价的：**

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

限制

只有定义上下文（上下文是表）之后，CurrentUser 才会起作用。

如果没有上下文，则必须使用其他函数。

示例：

要创建与上下文无关的操作，执行的文件路径取决于连接到 AssetCenter **数据库的用户。**

倘若此操作有上下文，则可以创建可执行文件**类型的操作，并设置**文件夹**字段，例如：**c:\scripts\[CurrentUser.Name]\。

但是，如果可执行文件**类型的操作没有上下文，则** [CurrentUser.Name] **将被视作固定文本。**

因此，必须找到另一种解决方案，例如使用脚本创建与上下文无关脚本**类型的操作：**

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

句柄的定义

句柄表示对象的唯一标识符。在 AssetCenter 环境中，对象可以是字段、链接、索引、查询、记录、表或连接。句柄是 32 位整数（长整型）。

注:

NULL 值不是有效的句柄。

也可以通过外部工具访问（数据库）连接句柄。

错误代码的定义

当函数失败时，将返回错误代码。

通过外部工具

外部工具可以通过 AmLastError() 和 AmLastErrorMsg() 分别恢复错误代码和与之关联的消息。可以使用 AmClearLastError() 函数清理这些代码和消息。

注:

任何新的函数调用都会清除错误代码和之前的消息。

通过内部

在内部（例如在 Basic 脚本中），可以使用 Err.Number 和 Err.Description 函数恢复最新的错误代码及其描述。

注:

在内部无需设计您自己的错误处理程序。有问题的脚本会停止，必要时会执行数据库回滚。

但也可以使用 Err.Raise 函数特意生成错误。其语法如下所示：

```
Err.Raise (<错误代码>, <错误消息>)
```

注:

如果该表中"有效性"字段的值导致了记录的创建或修改无效，为了警告用户，有一个较好的方法是使用 Err.Raise 函数生成一则错误消息（代码 12006 或 12007）。如果不这样做，用户未必会理解为什么不能修改或创建该记录。

下表列出了最常见的一些错误代码：

错误代码	含义
12001	未定义
12002	函数的参数不正确
12003	句柄无效或对象已被删除
12004	没有其他可用的数据。此错误通常出现在执行查询时。当查询未返回数据时，将会生成此错误。
12005	内部数据库服务器错误
12006	值无效（参数的类型不正确等）
12007	无有效的记录（例如，必填字段未填充）
12008	数据库访问权限问题
12009	已废弃或未实现的函数
12010	超过数据库的最大连接数

5 函数类型和参数

本章包含以下内容：

类型列表

下表归纳了函数或参数可用的各种类型：

类型	描述
Integer	整数（从 -32,768 到 +32,767）。
Long	整数（从 -2,147,483,647 到 +2,147,483,646）。
Single	4 字节浮点型数值（单精度）。
Double	8 字节浮点型数值（双精度）。
String	接受所有字符的文本。
日期	日期或日期+时间。
变量	可以代表任何类型的通用类型。

注:

并非所有的类型都可通过外部工具使用。通过外部工具只能使用长整型、双精度和字符串类型。不使用变量，同时整型和日期型对象使用长整型表示。

函数的类型

函数的类型对应于函数返回值的类型。建议特别注意函数的类型，因为函数类型使用不当，可能会造成程序编译和运行时错误。

例如，在字段默认值的定义中不能使用返回类型与该类型不同的函数。例如，假设将该默认值脚本分配给任何“日期”或“日期和时间”类型字段：

```
RetVal=AmLoginName()
```

AmLoginName() 函数以字符串（String 类型）的形式返回连接用户的名称。这样一来，此返回值的类型与“日期”类型字段不兼容，AssetCenter 会显示一则错误消息。

参数的类型

要正确执行函数，则函数中可以使用的参数也必须使用定义的类型。在函数语法中，参数将根据具体的类型加上前缀。为避免产生任何可能的混淆，此参考中使用的前缀将随函数语法（API 或 Basic）的不同而有所不同。下表归纳了在 API 语法和 Basic 语法中所使用的前缀的等价形式：

类型	API 语法中使用的前缀	Basic 语法中使用的前缀
整型	"i"	"i"
长整型	“h”表示句柄，“l”表示数字	"l"
双精度	"d"	"d"
字符串	"char*psz"	"str"
日期	"ltm"	"dt"
变量	"v"	"v"

II 使用 API

6 简介

AssetCenter API 是以 32 位 DLL 的形式提供的，可在 Windows 95/98、2000、XP 和 Server 2003 中使用。

支持以下环境：

- Visual Basic 4.0、5.0 和 6.0，
- Visual C++ 4.0、5.0 和 6.0，
- Visual Basic .NET 2002 和 2003，
- Visual Studio .NET 2002 和 2003，
- Visual C# .NET 2002 和 2003，
- 所有使用 VBA (Visual Basic for Applications) 的 Microsoft 产品。

警告:

未提供针对 .NET 环境的库入口点 (.dll)。如果希望使用这些开发环境，必须自己定义这些入口点。

注:

API 应与所有授权使用第三方 DLL 的工具相兼容。

警告

在使用 AssetCenter API 之前，用户应该熟悉 AssetCenter 概念模型中使用的术语。尤其是要对数据库的结构有最基本的了解。

有关数据库结构的信息可在手册《参考指南：管理员和高级应用》，“数据库结构”章中找到，亦可在位于 AssetCenter 安装文件夹的 doc\infos 子文件夹下的 Database.txt 和 Tables.txt 文件中找到。

安装

在使用 AssetCenter API 之前，强烈建议安装功能完整的 AssetCenter 版本。这样，可以快速测试是否可以从指定的计算机正确地访问数据库，以及创建或配置数据库连接。API 使用与 AssetCenter 相同的数据库层和配置信息访问数据源，因此通常可在 AssetCenter 内部查明问题所在。

使用 AssetCenter 设置开发环境的典型步骤如下：

- 安装带有 AssetCenter API 程序包的 32 位 AssetCenter 版本。
- 使用 AssetCenter 配置数据源，并尝试打开数据库。
- 使用您的开发环境调用 AssetCenter API 函数。

为帮助您熟悉 AssetCenter API，建议使用演示数据库或者任何非关键性的数据源，这样即使操作出错也不会带来严重的后果。

与 DLL 相关联的 .ini 配置文件

▶ 《AssetCenter - 安装》指南，.ini 和 .cfg 文件章。

请特别参考以下各节：

- 可用的 .ini 和 .cfg 文件
- .ini 文件的控制和修改

7 方法

使用 AssetCenter API **的典型操作顺序是：**

步骤 1 **使用** AQL **语句创建查询：**

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

注:

也可以使用 AssetCenter Export **生成** AQL **查询。**

步骤 2 **浏览查询结果集并检索特定项的任何有用句柄。**

步骤 3 **使用检索到的句柄更新相应对象中的信息。**

步骤 4 **提交（接受）或回滚（取消）整个事务。**

8 概念和示例

本节包含以下内容：

概念

AssetCenter 是采用面向对象设计技术构建的，API 则仍然停留在结构化视图。为突破 Windows DLL 必须使用扁平的“C 式 API”的限制，AssetCenter API 使用句柄（32 位整数）标识用户创建的每个对象。此方法的优点是允许非面向对象语言访问 AssetCenter 对象模型。

在执行其他操作之前，程序必须调用 AmStartUp() 初始化 AssetCenter 库。还必须通过调用 AmCleanUp() 函数终止程序。

在访问数据库对象之前，应该建立用户和数据库之间的连接。此连接被识别为“连接”对象的“句柄”（之后即可在所有与该数据库交互的 API 函数中使用此句柄）。它对应于参数 hApiCnxBase。之后，即可使用此对象创建查询并访问记录。

注:

所有的数据库对象将链接到一个连接上，这样可以查看有关用户特权的信息。

第一步是使用有效的数据源名称和有效的登录名和密码打开一个连接。

警告:

当通过 AssetCenter API 连接到 AssetCenter 数据库时，将占用一个连接槽。

处理日期

读取日期时，对于“日期”和“日期与时间”类型字段可选择使用以下两个函数：

- "AmGetFieldLongValue()"，返回 Unix "Long" (UTC) 形式的日期。建议使用此函数进行与日期有关的计算。
- AmGetFieldStrValue()，返回字符串形式的日期，其格式与 Windows 控制面板中的格式相同。此日期考虑到时区因素。建议在需要显示日期时使用此函数。

示例 1

下面用 C 语言编写的示例，声明了到演示数据库的连接：

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

lCnx 是一个连接句柄，可用于识别新创建的连接。

此连接现在可用于创建查询和访问数据库。下面用 C 语言编写的示例，定义了一个针对资产表的查询，然后浏览查询结果：

```
#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();
```

示例 2

查询用于找到数据库中的对象。当希望更新记录时，必须使用查询获取“记录”对象的句柄。然后可以使用其他 AssetCenter API **函数处理该记录。**

下面的示例显示了如何修改指定记录中的字段：

```
/* 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();
```

此例显示了如何使用查询机制获取唯一的记录句柄。在此示例代码中，使用查询对单一项进行了定位，但也可以使用 AmQueryExec() **来获取一组记录，从而得到一条或多条记录的句柄。**

注:

为了简明起见，此示例并未涉及所有可能的错误代码。

III 字母顺序参考

9 字母顺序参考

Abs()

返回数值的绝对值。

内部 Basic 语法

Function Abs(dValue As Double) As Double

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **dValue：想知道其绝对值的数值。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

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

AmActionDde()

此函数向 DDE 服务器应用程序发送 DDE 请求。使用此函数，AssetCenter 可以控制使用了 DDE 链接的其他应用程序。此函数等价于一个 DDE 类型的操作

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **strService：此参数包含要调用的可执行文件提供的DDE服务名。请参考可执行文件的文档，获取其提供的 DDE 服务的列表。**
- **strTopic：此参数包含主题（例如，上下文），DDE 操作必须在其中执行。**
- **strCommand：此参数包含外部应用程序必须执行的命令。必须遵守外部应用程序所定义的语法。**
- **strFileName：如果服务未在内存中驻存，则必须通过在此参数中指定用于激活此服务的可执行文件名（或与使用 Windows 文件管理器的可执行文件相关的任何文件名）来加载该服务。**
- **strDirectory：此参数包含在 strFileName 中定义的文件的路径。**
- **strParameters：此参数包含多个参数，启动服务时，这些参数将传递给用于激活服务的可执行文件。**
- **strTable：可选参数，操作与上下文相关时使用。表示应用此操作的记录所在表的 SQL 名称。**
- **lRecordId：可选参数，操作与上下文相关时使用。表示应用该操作的记录的标识符。**

输出参数

- **0：正常执行。**
- **非 0 值：错误代码。**

AmActionExec()

此函数启动 ".exe"、".com"、".bat"、".pif" 应用程序。还可参考任何类型的文档，并通过 Windows 文件管理器与可执行文件关联的文档扩展。此函数等价于 "可执行文件" 类型的操作。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strFileName：此参数包含可执行文件的名称，或通过 Windows 文件管理器与可执行文件关联的任何文档的名称。**
- **strDirectory：此参数包含在 strFileName 参数中指定的文件的路径。**
- **strParameters：该可选参数包含多个参数，启动可执行文件时，将这些参数提供给可执行文件。**
- **strTable：可选参数，操作与上下文相关时使用。表示应用此操作的记录所在表的 SQL 名称。**
- **lRecordId：可选参数，操作与上下文相关时使用。表示应用该操作的记录的标识符。**

输出参数

- **0：正常执行。**
- **非 0 值：错误代码。**

示例

下面的例子将执行 Windows NT 资源管理器（位于驱动器 "C" 的 "WinNT" 文件夹中）：

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

AmActionMail()

此函数通过由 AssetCenter **管理的邮件系统之一发送消息：**

- **内部邮件系统。**
- **基于** VIM **标准的外部邮件系统** (Lotus Notes **等**)
- **基于** MAPI **标准的外部邮件系统（**Microsoft Exchange、Microsoft Outlook **等）**
- **基于** SMTP **标准的外部邮件系统（**Internet **标准）**

API 语法

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 语法

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

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strTo：此参数包含格式为** messaging_system:address **的消息收件人的地址列表。分号用作分隔符。**
- **strCc：此参数包含地址列表，这些地址属于消息中被复制的人。分号用作分隔符。**
- **strCcc：此参数包含收到消息密送副本的人（未在收件人列表上显示）的地址列表。分号用作分隔符。**
- **strSubject：此参数包含消息主题。**

- **strMessage：此参数包含消息主体。**
- **iPriority：此参数定义发送消息的优先级**:
  - 0：**低优先级**
  - 1：**中优先级。**
  - 2：**高优先级。**
- **bAcknowledge：此参数表示消息发送者是否将接收到确认回执：**
  - 0：**消息发送者未接收到确认回执。**
  - 1：**消息发送者接收到确认回执。**
- **strRefObject：此参数仅用于通过** AssetCenter **内部邮件系统发送邮件。包含从记录得出的链接的** SQL **名称，该记录与到达参考对象的执行过程的上下文相对应。可以使用** CurrentUser **虚拟链接。**
- **strTable：可选参数，操作与上下文相关时使用。表示应用此操作的记录所在表的** SQL **名称。**
- **lRecordId：可选参数，操作与上下文相关时使用。表示应用该操作的记录的标识符。**

输出参数

- 0：**正常执行。**
- **非** 0 **值：错误代码。**

AmActionPrint()

此函数将打印与数据库中特定记录有关的报表。

内部 Basic 语法

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

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **lReportId：此参数包含要打印的报表的标识符。**
- **lRecordId：此参数包含与报表有关的记录的标识符。默认情况下，此参数设置为 "0"。通过报表隐式定义有关的表。**

输出参数

- **0：正常执行。**
- **非 0 值：错误代码。**

AmActionPrintPreview()

此函数将触发与特定数据库记录有关的报表的预览操作。

内部 Basic 语法

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

应用的字段

版本: 3.60

	可用
AssetCenter API	
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **lReportId：此参数包含相关报表的标识符。**
- **lRecordId：此参数包含与报表有关的记录的标识符。默认情况下，此参数为 "0"。**

输出参数

- 0：正常执行。
- 非 0 值：错误代码。

AmActionPrintTo()

此函数将在特定打印机上打印与特定数据库记录有关的报表。

内部 Basic 语法

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

应用的字段

版本: 3.5

	可用
AssetCenter API	
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strPrinterName**：此参数包含要使用的打印机的名称。
- **lReportId**：此参数包含要打印的报表的标识符。
- **lRecordId**：此参数包含与报表有关的记录的标识符。默认情况下，此参数设置为 "0"。

输出参数

- 0：正常执行。
- 非 0 值：错误代码。

AmAddAllPOLinesToInv()

此函数将完整的采购订单添加到现有的供应商发票中。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lPOrdId：此参数包含要添加到供应商发票的采购订单的标识符。**
- **lInvId：此参数包含要添加采购订单的供应商发票的标识符。**

输出参数

- 0：**正常执行。**
- **非** 0 **值：错误代码。**

AmAddCatRefAndCompositionToPOrder()

此函数可以将目录参考的全部内容添加到指定的采购订单。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lPOrderId：此参数包含要完成的采购订单的标识符。**
- **lCatRefId：此参数包含目录参考的标识符。**
- **dCatRefQty：此参数包含要添加的数量（其单位与产品有关）。**
- **lRequestId：此参数包含与采购订单对应的申请的标识符。**
- **dUnitPrice：此参数包含目录参考的产品单价。**
- **strCur：此参数包含用于表示单价的货币代码。**

输出参数

- **0：正常执行。**
- **非 0 值：错误代码。**

备注

注:

特别是此函数能够使用目录参考的产品的组成部分来填充采购订单。

AmAddCatRefToPOrder()

此函数可以将目录参考添加到采购订单。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lRequestLineId：此参数包含与采购订单相关的申请行的标识符。**
- **lCatRefId：此参数包含要添加的目录参考的标识符。**
- **lPOrderId：此参数包含与此操作有关的采购订单的标识符。**
- **dQty：此参数包含要添加的数量（其单位与产品有关）。**
- **bCanMerge：此参数可以指定是否将已添加的行与采购行中现有的行合并。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmAddEstimLinesToPO()

此函数将估价单的所有估价行添加到现有订单。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lEstimId：此参数包含要添加到采购订单的估价单的标识符。**
- **lPOrdId：此函数包含添加了估价单的所有估价行的采购订单的标识符。**
- **bMergeLines：此参数可以指定是否将相同的申请行 (bMergeLines**=1) **合并成特定的单行。对于要合并的行，同步增加指定的数量，并创建单行。**

输出参数

- 0：正常执行。
- 非 0 值：错误代码。

AmAddEstimLineToPO()

此函数将估价行添加到现有的采购订单。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lEstimLineId：此参数包含要添加到采购订单的估价行的标识符。**
- **lPOrdId：此参数包含要添加估价行的采购订单的标识符。**
- **bMergeLines：此参数可以指定是否将相同的申请行(bMergeLines=1)合并成特定的单行。对于要合并的行，同步增加指定的数量，并创建单行。**

输出参数

- 0：正常执行。

- **非 0 值：错误代码。**

AmAddLicContentToRequest()

此函数将许可证文件中包含的所有软件安装内容添加到采购申请。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: ?

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lRequestId：此参数包含与此操作有关的采购申请的标识符。**
- **lLicModelId：此参数包含许可证模型的标识符。**
- **lParent：此参数包含作为已创建申请行的父项的资产组合项或申请行的标识符。**
- **bExternalParent：如果此参数设置为 "1"，则已创建行的父项是申请行中现有的资产组合项。**

输出参数

- **0：正常执行。**
- **非 0 值：错误代码。**

备注

注:

仅当存在兼容性原因时才能包含此函数。建议您不要使用它。

AmAddPOLineToInv()

此函数将订单行上特定数量的项添加到供应商发票。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lPOrdLineId：此参数包含要添加到供应商发票的订单行的的标识符。**
- **lInvId：此参数包含要添加订单行项的供应商发票的标识符。**
- **dQty：此参数包含订单行上要添加到供应商发票的项数。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmAddPOrderLineToReceipt()

此函数可以将订单行添加到接收单。这样就可以接收现有接收单行中包含的订单行。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lPOrderLineId：此参数包含订单行的标识符。**
- **lRecptId：此参数包含受影响的接收单的标识符。**
- **dQty：此参数包含要接收的数量。这样就可以限制接收的数量，此数量与订购数量有关（按照产品单位计算）。**
- **bCanMerge：此参数可以指定是否将订单行与接收单中现有的行合并。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmAddReceiptLineToInvoice()

此函数可以将将接收单行添加到发票。然后，就可以为现有发票中包含的接收单行开具发票。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lRecptLineId：此参数包含接收单行的标识符。**
- **lInvoiceId：此参数包含受影响的发票的标识符。**
- **dQty：此参数包含要开具发票的数量。这样就可以限制要开具发票的数量，此数量与收货数量有关（按照产品单位计算）。**
- **bCanMerge：此参数可以指定是否将行与发票中现有的行合并。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmAddReqLinesToEstim()

此函数将申请的所有申请行添加到现有估价单。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lReqId：此参数包含要添加到估价单的申请的标识符。**
- **lEstimId：此参数包含估价单的标识符，用于添加申请的所有申请行。**
- **bMergeLines：此参数可以指定是否将相同的申请行 (bMergeLines**=1) **合并成特定的单行。对于要合并的行，同步增加指定的数量，并创建单行。**

输出参数

- 0：正常执行。
- 非 0 值：错误代码。

AmAddReqLinesToPO()

此函数将申请的所有申请行添加到现有采购订单。在申请中指定的供应商必须与相关的采购订单中的供应商相同。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lReqId：此参数包含要添加到采购订单的申请的标识符。**
- **lPOrdId：此参数包含要添加申请行的采购订单的标识符。**
- **bMergeLines：此参数可以指定是否将相同的申请行 (bMergeLines=1) 合并成特定的单行。对于要合并的行，同步增加指定的数量，并创建单行。**

输出参数

- 0：正常执行。
- 非 0 值：错误代码。

AmAddReqLineToEstim()

此函数将申请行添加到现有估价单。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lReqLineId**：此参数包含申请行的标识符以添加到估价单。
- **lEstimId**：此参数包含要添加申请行的估价单的标识符。
- **bMergeLines**：此参数可以指定是否将相同的申请行 (**bMergeLines**=1) 合并成特定的单行。对于要合并的行，同步增加指定的数量，并创建单行。

输出参数

- 0：正常执行。

- **非 0 值：错误代码。**

AmAddReqLineToPO()

此函数将申请行添加到现有采购订单。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lReqLineId：此参数包含申请行的标识符，用于添加到采购订单。**
- **lPOrdId：此参数包含采购订单的标识符，用于添加申请行。**
- **bMergeLines：此参数可以指定是否将相同的申请行(bMergeLines=1)合并成特定的单行。对于要合并的行，同步增加指定的数量，并创建单行。**

输出参数

- 0：**正常执行。**
- **非 0 值：错误代码。**

AmAddRequestLineToPOrder()

此函数可以将申请行添加到采购订单。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lRequestLineId：此参数包含申请行的标识符。**
- **lPOrderId：此参数包含受影响的采购订单的标识符。**
- **dQty：此参数包含要订购的数量。这样就可以限制要订购的数量，此数量与申请数量有关（按照产品单位计算）。**
- **bCanMerge：此参数可以指定是否将行与采购订单中现有的行合并。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmAddTemplateToPOrder()

此函数可以将标准采购订单的全部内容添加到特定的采购订单。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lRequestId：此参数包含能够满足要添加的订单行需要的申请行的标识符。**
- **lPOrderId：此参数包含受影响的采购订单的标识符。**
- **lTemplateId：此参数包含要添加的标准采购订单的标识符。**
- **lQty：此参数包含要添加的数量（按产品单位计算）。**
- **bCanMerge：此参数可以指定是否将行与采购订单中现有的行合并。**

输出参数

- **0：正常执行。**
- **非 0 值：错误代码。**

AmAddTemplateToRequest()

此函数可以将标准申请的全部内容添加到特定的申请。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lReqId：此参数包含受影响的申请行的标识符。**
- **lTemplateId：此参数包含要添加的标准申请的标识符。**
- **lQty：此参数包含要添加的数量（按产品单位计算）。**
- **bCanMerge：此参数可以指定是否将行与申请中现有的行合并。**

输出参数

- 0：**正常执行。**
- **非** 0 **值：错误代码。**

AmArchiveRecord()

此函数将归档数据库中的记录。记录将从源表中删除并移动到相应的归档表。

API 语法

long AmArchiveRecord(long hApiRecord);

内部 Basic 语法

Function AmArchiveRecord(hApiRecord As Long) As Long

应用的字段

版本: 4.3.0

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiRecord：此参数包含与此操作有关的记录的句柄。**

输出参数

- 0：**正常执行。**
- **非** 0 **值：错误代码。**

备注

注:

对链接记录的处理取决于链接的类型。对于 OWN **类型链接，链接记录的处理方式相同。对于** DEFINE **或** NORMAL **类型的链接，链接记录的外键将重置为** 0**，并将归档记录的标识符和描述字段填充到归档字段。**

重要:

此函数可用于标准表中的记录。

AmAttribCmdAvailability()

可以使用此函数确定帮助台记录单的“分配”或“不分配”按钮的可用性。

内部 Basic 语法

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

应用的字段

版本: 4.3.0

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **bAttrib：要测试“分配”按钮的可用性，将此参数设置为** "1"。 **：要测试“未分配”按钮的可用性，将此参数设置为** "0"。
- **lGroupID：此参数包含帮助台组的标识符，与相关的帮助台组关联。**
- **lInChargeID：此参数包含记录单受理人的标识符。**
- **bInChargeIsReadOnly：如果受理人只能查看记录单，则此参数设置为** "1"，**如果要具有修改权限，则设置为** "0"。

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**

- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmBackupRecord()

此函数将生成记录的备份副本。将记录复制到相应的归档表，但不从源表中删除。

API 语法

long AmBackupRecord(long hApiRecord);

内部 Basic 语法

Function AmBackupRecord(hApiRecord As Long) As Long

应用的字段

版本: 4.3.0

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiRecord：此参数包含与此操作有关的记录的句柄。**

输出参数

- 0**：正常执行。**
- **非** 0 **值：错误代码。**

备注

注:

对链接记录的处理取决于链接的类型。对于 OWN 类型链接，链接记录的处理方式相同。对于 DEFINE 或 NORMAL 类型的链接，链接记录的外键将重置为 0，并将归档记录的标识符和描述字段填充到归档字段。

重要:

此函数可用于标准表中的记录。

AmBuildNumber()

此函数将返回应用程序的内部版本号。

内部 Basic 语法

Function AmBuildNumber() As Long

应用的字段

版本: 4.3.0

	可用
AssetCenter API	
字段或链接的配置脚本	
"Script" 类型操作	
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

AmBusinessSecondsInDay()

根据日历计算一天中的工作秒数。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strCalendarSqlName：用于计算的日历的** SQL **名称。**
- **tmDate：执行计算的日期。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmCalcConsolidatedFeature()

计算通过其 SQL 名称标识的表上的合并的特征值。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lCalcFeatId：合并特征值的标识符。**
- **strSQLTableName：进行合并特征值计算的表的 SQL 名称。必须为此表定义特征值。**

输出参数

- **0：正常执行。**
- **非 0 值：错误代码。**

AmCalcDepr()

可以使用此函数计算特定日期下某个资产的折旧值。它将返回此日期的折旧值。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **iType：此参数标识折旧性质。可能是下面的值：**
  - **0：无折旧**
  - **1：直线法折旧**
  - **2：递减余额法折旧**
- **lDuration：此参数包含计算资产折旧的期间。此期间以秒为单位表示。**
- **dCoeff：此参数包含应用于递减余额折旧法的系数。在直线折旧法中不会解释此参数，但是此参数必须赋值。**
- **dPrice：此参数包含与折旧计算有关的资产的市场价值。**
- **tmStart：此参数包含计算资产折旧的起始日期。**
- **tmDate：此参数包含计算资产折旧和残值的日期。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmCalculateCatRefQty()

可以使用此函数计算要订购的目录参考的数量以完成采购订单。

API 语法

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 语法

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

应用的字段

版本: ?

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **lSetQty：此参数包含与产品中每项相应的数量（如例** 1**，产品中包含** 6 **个一升瓶装水）。**
- **lUseUnitId：此参数包含模型单位的标识符。**

- **lPurchUnitId：此参数包含产品单位的标识符。**
- **strModelDesc：此参数包含对模型的描述。**
- **strCatRefDesc：此参数包含对目录参考的描述。**
- **strPurchUnit：此参数包含对产品单位的描述。**
- **strUseUnit：此参数包含对模型单位的描述。**
- **dPkgQty：此参数包含与产品中每项相应的数量（如例 1，产品中包含 6 个一升瓶装水）。**
- **dUnitConv：此参数包含产品单位的换算比率。**
- **dReqLineQty：此参数包含采购订单中的模型的数量。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

AmCalculateReqLineQty()

可以使用此函数计算创建采购订单所需模型的数量。

API 语法

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 语法

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

应用的字段

版本: ?

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **lSetQty：此参数包含与产品中每项相应的数量（如例 1，产品中包含 6 个一升瓶装水）。**
- **lUseUnitId：此参数包含模型单位的标识符。**
- **lPurchUnitId：此参数包含产品单位的标识符。**
- **strModelDesc：此参数包含对模型的描述。**
- **strCatRefDesc：此参数包含对目录参考的描述。**
- **strPurchUnit：此参数包含对产品单位的描述。**
- **strUseUnit：此参数包含对模型单位的描述。**
- **dPkgQty：此参数包含与产品中每项相应的数量（如例 1，产品中包含 6 个一升瓶装水）。**
- **dUnitConv：此参数包含产品单位的换算比率。**
- **dCatRefQty：此参数包含订购的目录参考中模型的数量。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

AmCbkReplayEvent()

可以使用此函数，在更正由事件生成的记录后，重新应用由事件生成的费用分摊规则。

API 语法

long AmCbkReplayEvent(long hApiCnxBase, long lCbkEventId);

内部 Basic 语法

Function AmCbkReplayEvent(lCbkEventId As Long) As Long

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lCbkEventId：此参数包含相关的费用分摊事件的标识符。**

输出参数

- 0：**正常执行。**
- **非** 0 **值：错误代码。**

AmCheckTraceDone()

AmCheckTraceDone API **确定是否将端口** (lPortId) **或线束** (lBundleId) **连接到现有的跟踪。跟踪方向** (iTraceDir) **标识是应该按照用户到主机** (iTraceDir = 1) **还是按照主机到用户** (iTraceDir = 0) **的方向检查跟踪。**

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **lPortId：此参数包含要检查的端口** ID。
- **lBundleId：此参数包含要检查的线束** ID。
- **iTraceDir：此参数定义检查方向。**
  - 1：**按照主机方向检查**
  - 0：**按照主机方向检查**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmCleanup()

必须在使用了数据库修改函数的脚本的结尾处调用此函数。将释放所有已用资源。

API 语法

void AmCleanup();

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	
向导脚本	
向导的 FINISH.DO 脚本	

AmClearLastError()

此函数将清除与在最后一个函数调用期间发生的最后一个错误消息有关的信息。

API 语法

long AmClearLastError(long hApiCnxBase);

内部 Basic 语法

Function AmClearLastError() As Long

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输出参数

- **0：正常执行。**
- **非 0 值：错误代码。**

AmCloseAllChildren()

此函数将销毁在当前连接期间创建的所有对象。

API 语法

long AmCloseAllChildren(long hApiCnxBase);

内部 Basic 语法

Function AmCloseAllChildren() As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输出参数

- 0：正常执行。
- 非 0 值：错误代码。

AmCloseConnection()

结束特定连接的AssetCenter会话。在此连接中创建的所有对象（查询、记录、表、字段等）将自动销毁。其句柄将变为无效。连接句柄不再存在。

API 语法

long AmCloseConnection(long hApiCnxBase);

应用的字段

版本: 2.52

	可用
AssetCenter API	✓

	可用
字段或链接的配置脚本	
"Script" 类型操作	
向导脚本	
向导的 FINISH.DO 脚本	

输出参数

- 0：正常执行。
- 非 0 值：错误代码。

AmCommit()

此函数将提交对与此连接相关的数据库进行的所有修改。

API 语法

long AmCommit(long hApiCnxBase);

内部 Basic 语法

Function AmCommit() As Long

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输出参数

- 0：正常执行。
- 非 0 值：错误代码。

AmComputeAllLicAndInstallCounts()

此函数将对所有记录执行软件许可证和安装计数操作。

API 语法

long AmComputeAllLicAndInstallCounts(long hApiCnxBase);

内部 Basic 语法

Function AmComputeAllLicAndInstallCounts() As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输出参数

- 0：**正常执行。**
- **非 0 值：错误代码。**

AmComputeLicAndInstallCounts()

此函数将对一个记录执行软件许可证和安装计数操作。

API 语法

long AmComputeLicAndInstallCounts(long hApiCnxBase, long lSLCountId);

内部 Basic 语法

Function AmComputeLicAndInstallCounts(lSLCountId As Long) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lSLCountId：此参数包含软件许可证计数器的标识符。**

输出参数

- 0：**正常执行。**
- **非** 0 **值：错误代码。**

AmConnectionName()

此函数将返回当前的数据库连接名。

API 语法

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

内部 Basic 语法

Function AmConnectionName() As String

应用的字段

版本: 4.3.0

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
RetVal=amConnectionName()
```

AmConnectTrace()

AmConnectTrace API **用于将源设备/电缆连接到目标设备/电缆并创建跟踪历史记录和跟踪操作。**

API 语法

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 语法

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

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **iSrcLinkType：此参数确定对源设备/电缆的跟踪类型。**
  - 8：**电缆**
  - 9：**设备**
- **lSrcPortBunId：此参数包含要在源端连接的端口或线束。**
- **lSrcLabelRuleId：此参数是源链接的标签规则。**
- **iDestLinkType：此参数确定对目标设备/电缆的跟踪类型。**
  - 8：**电缆**
  - 9：**设备**
- **lDestPortBunId：此参数包含要在目标端连接的端口或线束。**
- **lDestLabelRuleId：此参数是目标链接的标签规则。**
- **iTraceDir：此参数定义连接的方向。**
  - 1：**用户到主机**
  - 0：**主机到用户**
- **lDutyId：此参数是电缆类型链接的负载。**
- **strComment：此参数是跟踪操作的标签。**
- **lCabTraceOutId：此参数是电缆跟踪输出** ID。

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

AmConvertCurrency()

此函数执行特定日期的两种货币之间的转换操作。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **tmDate：此参数包含转换日期。通过它可以了解当日的有效汇率。**
- **strSrcName：此参数包含要转换的源货币，即要转换的货币。**
- **strDstName：此参数包含转换后的目标货币，即表示源货币的货币。**
- **dVal：此参数包含要转换的金额（按源货币的单位表示）。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

备注

注:

此函数当前的参数（strSrcName 和 strDstName）必须在 AssetCenter 中定义。此外，需要执行转换时，必须存在有效的转换日（tmDate 参数）的汇率。

示例

下面的例子是在 1998 年 11 月 2 日，将 5,000 法郎转换成美元。

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

AmConvertDateBasicToUnix()

此函数将 Basic 格式日期（"日期"型）转换成 Unix 格式日期（"长整"型）。因为这两种类型等价，所以此函数不用于外部工具。

API 语法

long AmConvertDateBasicToUnix(long hApiCnxBase, long tmTime);

内部 Basic 语法

Function AmConvertDateBasicToUnix(tmTime As Date) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **tmTime：此参数包含要转换的日期。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmConvertDateIntlToUnix()

此函数将国际格式日期（"日期"型）转换成 Unix **格式日期（"长整"型）。**

API 语法

long AmConvertDateIntlToUnix(long hApiCnxBase, char *strDate);

内部 Basic 语法

Function AmConvertDateIntlToUnix(strDate As String) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **strDate：此参数包含要转换的国际格式** (yyyy-mm-dd hh:mm:ss) **日期。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmConvertDateStringToUnix()

将日期从字符串格式（在 Windows **控制面板中显示的格式）转换为** Unix **的"长整型"。**

API 语法

long AmConvertDateStringToUnix(long hApiCnxBase, char *strDate);

内部 Basic 语法

Function AmConvertDateStringToUnix(strDate As String) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **strDate：要转换的字符串格式的日期。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmConvertDateUnixToBasic()

此函数将 Unix **格式日期（"长整"型）转换成** Basic **格式日期（"日期"型）。因为这两种类型等价，所以此函数不用于外部工具。**

API 语法

long AmConvertDateUnixToBasic(long hApiCnxBase, long lTime);

内部 Basic 语法

Function AmConvertDateUnixToBasic(lTime As Long) As Date

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **ITime：此参数包含要转换的日期。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmConvertDateUnixToIntl()

此函数将Unix**格式日期（"长整型"）转换成国际格式日期（**yyyy-mm-dd hh:mm:ss**）。**

API 语法

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

内部 Basic 语法

Function AmConvertDateUnixToIntl(lUnixDate As Long) As String

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **lUnixDate：此参数包含要转换的日期。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmConvertDateUnixToString()

将"长整型"的 Unix **格式日期转换为字符串格式（在** Windows **控制面板中显示的格式）日期。**

API 语法

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

内部 Basic 语法

Function AmConvertDateUnixToString(lUnixDate As Long) As String

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **lUnixDate: 要转换的"长整型"的 Unix 格式日期。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

AmConvertDoubleToString()

此函数将双精度数值转换为字符串。根据在 Windows 控制面板中定义的区域选项（数值）确定字符串的格式。

API 语法

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

内部 Basic 语法

Function AmConvertDoubleToString(dSrc As Double) As String

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **dSrc：此参数包含要转换的双精度数值。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmConvertMonetaryToString()

此函数将货币值转换为字符串。根据在 Windows **控制面板中定义的区域选项（货币）确定字符串的格式。**

API 语法

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

内部 Basic 语法

Function AmConvertMonetaryToString(dSrc As Double) As String

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **dSrc：此参数中包含要转换的货币值。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmConvertStringToDouble()

此函数将字符串（其格式与在 Windows**控制面板中定义的格式相对应）转换成双精度数值。**

API 语法

double AmConvertStringToDouble(char *strSrc);

内部 Basic 语法

Function AmConvertStringToDouble(strSrc As String) As Double

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **strSrc：此参数包含要转换的字符串。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmConvertStringToMonetary()

此函数将字符串（其格式与在 Windows**控制面板中定义的格式相对应）转换成货币值。**

API 语法

double AmConvertStringToMonetary(char *strSrc);

内部 Basic 语法

Function AmConvertStringToMonetary(strSrc As String) As Double

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **strSrc：此参数包含要转换的字符串。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmCounter()

此函数将返回 strCounterName 计数器的值并加 1**。如果 iWidth 参数比计数器的位数多，则在计数器值的开头处添加** 0 **作为补充。如果计数器的位数比** iWidth **中存储的值多，也不会返回截断的结果。**

内部 Basic 语法

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

应用的字段

版本: 2.52

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	
向导脚本	
向导的 FINISH.DO 脚本	

输入参数

- **strCounterName：计数器的名称与在** AssetCenter **中定义的相同（通过** 管理/计数器 **菜单项访问）。**

- **iWidth：此参数值强制函数的输出格式按 n 位数值表示。仅当计数器的值小于此参数值时，才使用此参数。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

备注

注:

如果在脚本类型的操作中使用此函数，则必须指定操作的上下文。否则将生成错误。

示例

下面的例子将返回 5 **位数的** "Delivery" **计数器值**:

```
Dim MyString As String
strCounter = AmCounter("Delivery", 5)
```

例如，如果 "Delivery" **计数器等于** "18"，**则函数将返回：**

```
00019
```

AmCreateAssetPort()

AmCreateAssetPort API **将在设备上** (lAssetId) **创建新端口。新端口将包含特定电缆连接器类型** (lCabCnxTypeId) **的特定的针脚数** (iPinCount)。**针脚的状态必须为**"**可用**"。**添加到端口的针脚将按序号进行排序。根据端口方向** (iPinPortDir)，**按照升序** (iPinPortDir = 0) **或降序** (iPinPortDir = 1) **对可用的针脚进行排序。此函数将为新端口分配特定的负载** (lDutyId)。

API 语法

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 语法

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

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lAssetId：此参数是设备** ID。
- **lCabCnxTypeId：此参数是电缆连接类型** ID。
- **lDutyId：此参数是端口的负载类型** ID。
- **iPinCount：此参数是用于新端口的针脚数。**
- **bPinPortDir：此参数指定端口的方向。**
- **iConnStatus**
- **bConsecutivePins**
- **iPrevPinSeq**
- **bLogError**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmCreateAssetsAwaitingDelivery()

可以使用此函数创建等待接收的资产。

API 语法

long AmCreateAssetsAwaitingDelivery(long hApiCnxBase, long lPOrdId);

内部 Basic 语法

Function AmCreateAssetsAwaitingDelivery(lPOrdId As Long) As Long

应用的字段

版本: 3.61

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lPOrdId：此参数包含有关的采购订单的标识符。**

输出参数

- 0**：正常执行。**
- **非** 0 **值：错误代码。**

AmCreateCable()

AmCreateCable API **创建新电缆。使用指定的模型类型** (lModelId)**、电缆角色** (strCableRole)**、电缆的标签规则** (lLabelRuleId)**、电缆的用户位置** (lUserLoc) **和电缆的主机位置** (lHostLoc) **创建电缆。如果项目** (lProjectId) **和工作单** (lWorkOrderId) **已赋**

值，根据特定的备注(strComment)将新电缆添加到项目和工作单。此备注描述了要在电缆上执行的操作（如"安装新电缆"）。

API 语法

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 语法

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

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lModelId：此参数是电缆模型** ID。
- **lUserId：此参数是用户端位置** ID。
- **lHostId：此参数是主机端位置** ID。
- **strCableRole：此参数定义电缆角色。**
- **lProjectId：此参数定义与电缆放置相关的项目。**
- **lWorkOrderId：此参数定义与电缆放置相关的工作单。**
- **strComment：此参数是在工作单（由** lWorkOrderId **定义）上使用的备注。**
- **lLabelRuleId：此参数定义创建电缆标签时应用的标签规则。**
- **strLabel：此参数指定附在电缆上的标签。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmCreateCableBundle()

AmCreateCableBundle API **将在电缆上** (lCableId) **创建新线束。新线束将包含特定电缆线对类型** (lPairType) **的特定的电缆线对数** (iPairCount)**。线对的状态必须为"可用"。此函数将为新线束分配特定的负载** (lDutyId)**。**

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lCableId：此参数是电缆** ID（**必须在电缆表中存在**）。
- **lPairTypeId：此参数是电缆线对类型** ID。
- **lDutyId：此参数是电缆线束的负载** ID。
- **iPairCount：此参数定义线束的线对数目。**

- **iStartPairSeq**
- **bLogError**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmCreateCableLink()

AmCreateCableLink API **为特定电缆** (lCableId) **和线束** (lNextBundle) **创建新的电缆类型电缆链接。使用指定的负载** (lDutyId) **设置电缆链接的负载。使用指定的标签规则** (lLabelRule) **设置电缆链接的标签规则。**

注:

使用指定的标签规则不会更新标签，必须单独调用 AmRefreshLabel()。

如果指定了上一级链接 (lPrevLinkId)**，则父链接位于其上一级链接是子链接的两个记录之间。**

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	

	可用
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lCableId：此参数是连接的电缆 ID。**
- **lDutyId：此参数是连接负载。**
- **lBundleId：此参数包含要连接的电缆线束的 ID。**
- **lPrevLinkId：此参数定义用于连接的电缆链接 ID。可以选择使用值 0。**
- **iTraceDir：此参数定义连接的方向。**
  - 0=**主机到用户**
  - 1=**用户到主机**
- **lLabelRuleId：此参数包含要使用的标签规则 ID。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

AmCreateDelivFromPO()

此函数接收采购订单并返回创建的接收单的标识符。

API 语法

long AmCreateDelivFromPO(long hApiCnxBase, long lPOrdId);

内部 Basic 语法

Function AmCreateDelivFromPO(lPOrdId As Long) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lPOrdId：此参数包含要接收的采购订单的标识符。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

AmCreateDevice()

AmCreateDevice API **创建新设备。使用指定的模型类型** (lProductId) **和位置** (lLocId) **创建设备。将资产的标签规则设置为指定的规则** (lLabelRuleId)**。**

注:

使用指定的标签规则不会更新标签，必须单独调用 AmRefreshLabel**。**

如果项目 (lProjectId) **和工作单** (lWorkOrderId) **已赋值，根据** strComment **中包含的备注将新资产添加到项目和工作单。此备注描述了要在资产上执行的操作（如"安装新资产"）。**

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lModelId：此参数是新设备的模型** ID。
- **lLocationId：此参数是新设备的位置** ID。
- **lProjectId：此参数是项目** ID。**它可以为** 0。
- **lWorkOrderId：此参数定义工作单** ID。**它可以为** 0。
- **lLabelRuleId：此参数定义用于资产的标签规则** ID。
- **strComment：此参数是在工作单上使用的备注。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmCreateDeviceLink()

AmCreateDeviceLink API **为指定的设备** (lAssetId) **和端口** (lPortId) **创建新的设备类型的电缆链接。使用指定的标签规则** (lLabelRule) **设置电缆链接的标签规则。**

注:

使用指定的标签规则不会更新标签，必须单独调用 AmRefreshLabel。

如果指定了上一级链接 (lPrevLinkId)**，则父链接位于两个记录之间。如果跟踪方向是用户到主机** (iTraceDir = 1)**，则上一级链接是子链接。如果跟踪方向是主机到用户** (iTraceDir = 0)**，则上一级链接是父链接。**

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lAssetId：此参数包含要连接的资产的标识符。**
- **lPortId：此参数包含要连接的端口的标识符。**
- **lPrevLinkId：此参数包含启用连接的设备链接的标识符。**
- **iTraceDir：此参数指定连接的方向。**
  - 0=**主机到用户**
  - 1=**用户到主机**
- **lLabelRuleId：此参数包含用于新连接的标签规则的标识符。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

AmCreateEstimFromReq()

此函数创建对采购申请的估价单并返回已创建的估价单的标识符。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lReqId：此参数包含用于创建估价单的采购申请的标识符。**
- **lSuppId：此参数包含使用此函数创建的估价单对应的供应商的标识符。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

AmCreateEstimsFromAllReqLines()

此函数创建对申请的估价单并返回已创建的估价单的标识符。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lReqId：此参数包含作为估价单依据的申请的标识符。**
- **bMergeLines：此参数可以指定是否将相同的申请行 (bMergeLines=1) 合并成特定的单行。对于要合并的行，同步增加指定的数量，并创建单行。**
- **lDefSuppId：此参数包含与估价单对应的默认供应商的标识符。**

输出参数

- **0：正常执行。**
- **非 0 值：错误代码。**

AmCreateInvFromPO()

此参数创建采购订单对应的供应商发票并返回已创建的供应商发票的标识符。

API 语法

long AmCreateInvFromPO(long hApiCnxBase, long lPOrdId);

内部 Basic 语法

Function AmCreateInvFromPO(lPOrdId As Long) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lPOrdId：此参数包含作为发票依据的采购订单的标识符。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmCreateLink()

此函数修改记录的链接并使其指向目标表中的新记录 (hApiRecDest)。因此，它在两个记录之间创建了链接。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiRecord：此参数包含记录句柄，该记录包含要修改的链接。**
- **strLinkName：此参数包含要修改的链接的** SQL **名称。**
- **hApiRecDest：此参数包含链接的目标记录的句柄。**

输出参数

- 0**：正常执行。**
- **非** 0 **值：错误代码。**

AmCreateOrUpdateInvoiceFromReceipt()

此函数可以根据接收单创建或更新发票。

API 语法

long AmCreateOrUpdateInvoiceFromReceipt(long hApiCnxBase, long lRecptId);

内部 Basic 语法

Function AmCreateOrUpdateInvoiceFromReceipt(lRecptId As Long) As Long

应用的字段

版本: ?

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **lRecptId：此参数包含与此操作有关的发票的标识符。**

输出参数

此函数返回生成的发票的标识符。

备注

注:

从外部工具调用此函数不能完成更新。

AmCreatePOFromEstim()

此函数根据估价创建采购订单并返回已创建的采购订单的标识符。

API 语法

long AmCreatePOFromEstim(long hApiCnxBase, long lEstimId);

内部 Basic 语法

Function AmCreatePOFromEstim(lEstimId As Long) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lEstimId：此参数包含用于创建采购订单的估价单的标识符。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmCreatePOFromReq()

此函数根据采购申请创建采购订单并返回已创建的采购订单的标识符。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **lReqId：此参数包含用于创建采购订单的采购申请的标识符。**
- **lSuppId：此参数包含使用此函数创建的采购订单对应的供应商的标识符。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmCreatePOrderFromRequest()

此函数可以根据申请创建采购订单。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **lRequestId：此参数包含相关申请的标识符。**
- **lSupplierId：此参数包含与采购订单对应的供应商的标识符。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

AmCreatePOrdersFromRequest()

此函数可以创建满足特定申请需要的所有采购订单。

API 语法

long AmCreatePOrdersFromRequest(long hApiCnxBase, long lRequestId);

内部 Basic 语法

Function AmCreatePOrdersFromRequest(lRequestId As Long) As Long

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **lRequestId：此参数包含相关申请的标识符。**

输出参数

- **0：正常执行。**
- **非 0 值：错误代码。**

AmCreatePOsFromAllReqLines()

此函数根据申请的申请行创建所有的采购订单。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **lReqId：此参数包含创建采购订单所依据的申请的标识符。**
- **bMergeLines：此参数可以指定是否将相同的申请行 (bMergeLines=1) 合并成特定的单行。对于要合并的行，同步增加指定的数量，并创建单行。**
- **lDefSuppId：此参数包含与申请项对应的默认供应商的标识符。默认情况下，此参数可选并设置为 "0"。**

输出参数

- **0：正常执行。**
- **非 0 值：错误代码。**

AmCreateProjectCable()

AmCreateProjectCable API **将电缆** (lCableId) **添加到项目** (lProjectId) **和工作单** (lWorkOrderId)。**备注** (strComment) **描述要执行的操作（如"安装新电缆"）。**

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓

	可用
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lProjectId：此参数是获得电缆的项目** ID。
- **lWorkOrderId：此参数是用于电缆的工作单** ID。
- **lCableId：此参数是电缆** ID。
- **strComment：此参数是在工作单上使用的备注。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmCreateProjectDevice()

AmCreateProjectDevice API **将设备** (lAssetId) **添加到项目** (lProjectId) **和工作单** (lWorkOrderId)。**备注** (strComment) **描述要执行的操作（如"安装新设备"）。**

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 4.00

	可用
AssetCenter API	✓

	可用
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lProjectId：此参数定义获得新设备的项目** ID。
- **lWorkOrderId：此参数定义获得新设备的工作单** ID。
- **lAssetId：此参数是新的设备资产** ID。
- **strComment：此参数是在工作单上使用的备注。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmCreateProjectTrace()

AmCreateProjectTrace API **将跟踪** (strTrace) **添加到项目** (lProjectId) **和工作单** (lWorkOrderId)。**使用指定的负载** (lDutyId) **设置跟踪服务。跟踪类型** (lTraceType) **表示跟踪是处于连接** (iTraceType = 1) **还是断开连接** (lTraceType = 2) **状态。要修改的用户链接标签** (strModLinkLabel) **标识要修改的跟踪的部分。备注** (strComment) **描述要执行的操作（如"连接这些设备"）。**

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lProjectId：此参数定义项目** ID **以获得跟踪信息。**
- **lWorkOrderId：此参数定义工作单** ID **以获得跟踪信息。**
- **iTraceType：此参数定义跟踪类型。**
  - 1=**连接**
  - 2=**断开连接**
- **lDutyId：此参数定义负载。在工作单上显示此参数。**
- **strModLinkLabel：此参数定义在工作单上使用的备注。**
- **strTrace：此参数定义在工作单上使用的跟踪输出字符串。**
- **strComment：此参数是在工作单上使用的备注。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmCreateReceiptFromPOrder()

此函数可以根据采购订单创建接收单。

API 语法

long AmCreateReceiptFromPOrder(long hApiCnxBase, long lPOrderId);

内部 Basic 语法

Function AmCreateReceiptFromPOrder(lPOrderId As Long) As Long

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lPOrderId：此参数包含采购订单的标识符。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

AmCreateRecord()

此函数在使用默认值的表中创建空记录。在插入数据库之前，数据库中没有该新记录。

API 语法

long AmCreateRecord(long hApiCnxBase, char *strTable);

内部 Basic 语法

Function AmCreateRecord(strTable As String) As Long

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **strTable：此参数包含表的** SQL **名称，可在该表中创建记录。**

示例

通过下面的示例可以在数据库中创建员工：

```
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()

此函数可以创建采购循环中的所有对象：申请、采购订单、接收单、发票。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **dQty：此参数依次包含订购数量、收货数量和开具发票数量（按打包单位）。**
- **lCatRefId：此参数包含目录参考的标识符。**
- **dUnitPrice：此参数包含目录参考的单价。**
- **strCur：此参数包含目录参考单价的货币代码。**
- **lRequesterId：此参数包含申请人的标识符。**
- **lCostId：此参数包含受影响的成本中心的标识符。**
- **lUserId：此参数包含与订购项对应的用户标识符。**
- **lStockId：此参数包含交付库存项的标识符。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

备注

等价于依次调用：amCreateRequestToReceipt、amCreateOrUpdateInvoiceFromReceipt。

AmCreateRequestToPOrder()

此函数可以创建采购循环中的下列对象：申请、采购订单。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **dQty：此参数包含订购数量（按打包单位）。**
- **lCatRefId：此参数包含目录参考的标识符。**
- **dUnitPrice：此参数包含目录参考的单价。**
- **strCur：此参数包含目录参考单价的货币代码。**
- **lRequesterId：此参数包含申请人的标识符。**
- **lCostId：此参数包含受影响的成本中心的标识符。**
- **lUserId：此参数包含与订购项对应的用户标识符。**
- **lStockId：此参数包含交付库存项的标识符。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmCreateRequestToReceipt()

此函数可创建采购循环中的下列对象：申请、采购订单、接收单。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **dQty：此参数包含要订购和收货的数量（按打包单位）。**
- **lCatRefId：此参数包含目录参考的标识符。**
- **dUnitPrice：此参数包含目录参考的单价。**
- **strCur：此参数包含目录参考单价的货币代码。**
- **lRequesterId：此参数包含申请人的标识符。**
- **lCostId：此参数包含受影响的成本中心的标识符。**
- **lUserId：此参数包含与订购项对应的用户标识符。**
- **lStockId：此参数包含交付库存项的标识符。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

备注

等价于依次调用：amCreateRequestToPOrder、amCreateReceiptFromPOrder。

AmCreateReturnFromReceipt()

此函数可以根据接收单创建退货单。

API 语法

long AmCreateReturnFromReceipt(long hApiCnxBase, long lRecptId);

内部 Basic 语法

Function AmCreateReturnFromReceipt(lRecptId As Long) As Long

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lRecptId：此参数包含接收单行的标识符。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmCreateTraceHist()

AmCreateTraceHist API **根据现有的源设备/电缆到目标设备/电缆的连接，创建跟踪历史记录和跟踪操作。**

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lSrcLinkId：此参数是用于源链接的设备/电缆。**
- **lDestLinkId：此参数是用于目标链接的设备/电缆。**
- **iTraceDir：此参数指定连接的方向。**
  - 0=**主机到用户**
  - 1=**用户到主机**

- **lCabTraceOutId：此参数是电缆跟踪输出** ID。
- **strComment：此参数是与跟踪操作关联的备注。**

输出参数

- 0：**正常执行。**
- **非** 0 **值：错误代码。**

AmCreateTraceLink()

此函数可在电缆设备之间创建链接。

内部 Basic 语法

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

应用的字段

版本: 4.3.0

	可用
AssetCenter API	
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **iLinkType：此参数用于标识要考虑的元素类型（**"1" **表示电缆设备，**"0" **表示电缆）。**
- **lAstCabId：此参数包含与电缆设备关联的资产的标识符。**
- **lPrtBunId：此参数包含与此操作有关的记录的标识符。此标识符在** amCableBundle **表中标识电缆或在** amPort **表中标识电缆设备。**
- **lPrevLinkId：此参数包含作为链接起始点的元素的标识符。**
- **iTraceDir：此参数用于指定链接的方向。**Either "HOST_TO_USER" or "USER_TO_HOST".

- **lDutyId：此参数包含链接负载的标识符。**
- **lLabelRuleId：此参数包含链接的标签规则的标识符（默认情况下，此值为空）。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

AmCryptPassword()

此函数对用户密码进行加密，通过登录名和密码标识。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 3.5

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strUser：此参数包含用户的登录名，要对该用户的密码进行加密。**
- **strPasswd：此参数以无格式文本形式包含要加密的密码。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

AmCurrentDate()

此函数返回客户端工作站上的当前日期。

API 语法

long AmCurrentDate();

内部 Basic 语法

Function AmCurrentDate() As Date

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

备注

如果已配置数据库使用时区，则根据是否直接从 AssetCenter 或外部程序调用此函数来确定其功能。在 AssetCenter 中，此函数与 Basic 中的 Now() 函数作用相同。从外部程序调用时，此函数返回的值表示为 GMT+0 形式并且不考虑夏令时因素。

AmCurrentIsoLang()

此函数返回 AssetCenter 使用的 ISO 语言代码（"en" 表示英语，"fr" 表示法语，等等）。

API 语法

long AmCurrentIsoLang(char *pstrLanguage, long lLanguage);

内部 Basic 语法

Function AmCurrentIsoLang() As String

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输出参数

出现错误时，有两种可能性：

- 在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。
- 如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。

AmCurrentLanguage()

此函数返回 AssetCenter 的语言版本（"US" 表示英语，"FR" 表示法语，等等）。

API 语法

long AmCurrentLanguage(char *pstrLanguage, long lLanguage);

内部 Basic 语法

Function AmCurrentLanguage() As String

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

AmCurrentServerDate()

此函数返回服务器上的当前日期。

API 语法

long AmCurrentServerDate(long hApiCnxBase);

内部 Basic 语法

Function AmCurrentServerDate() As Date

应用的字段

版本: 3.5

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmDateAdd()

根据添加真实期限时的起始日期，此函数计算新的日期。

API 语法

long AmDateAdd(long tmStart, long tsDuration);

内部 Basic 语法

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

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **tmStart：此参数包含添加期限时的日期。**
- **tsDuration：此参数包含在 tmStart 日期添加的按秒表示的期限。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

下面的例子描述了 amDateAdd() **和** amDateAddLogical() **函数的不同之处。分别使用这两个函数在日期为** 1/1/1999（1999 **年** 1 **月** 1 **日）时添加为期** 30 **天的期限。**

这种情况下，AmDateAdd **将添加** 30 **天的真实期限：**

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

函数返回的值是：

```
1999/01/31
```

这种情况下，AmDateAddLogical **将添加** 30 **天（即** 1 **个月）的逻辑期限：**

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

函数返回的值是：

```
1999/02/01
```

AmDateAddLogical()

根据添加逻辑期限时的起始日期，此函数计算新的日期（1 **个月包含** 30 **天）。**

API 语法

long AmDateAddLogical(long tmStart, long tsDuration);

内部 Basic 语法

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

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **tmStart：此参数包含添加期限时的日期。**
- **tsDuration：此参数包含在 tmStart 日期添加的按秒表示的期限。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

下面的例子描述了 amDateAdd() **和** amDateAddLogical() **函数的不同之处。分别使用这两个函数在日期为** 1/1/1999（1999 **年** 1 **月** 1 **日）时添加为期** 30 **天的期限。**

这种情况下，AmDateAdd **将添加** 30 **天的真实期限：**

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

函数返回的值是：

```
1999/01/31
```

这种情况下，AmDateAddLogical **将添加** 30 **天（即** 1 **个月）的逻辑期限：**

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

函数返回的值是：

```
1999/02/01
```

AmDateDiff()

此函数计算这两个日期之间的期限或时间跨度，以秒为单位。

API 语法

long AmDateDiff(long tmEnd, long tmStart);

内部 Basic 语法

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

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **tmEnd：此参数包含要计算的期间的截止日期。**
- **tmStart：此参数包含要计算的期间的起始日期。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

下例中计算的起止时间为 01/01/98 **和** 01/01/99**。**

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

AmDbExecAql()

此函数可在数据库上执行 AQL **查询。**

API 语法

long AmDbExecAql(long hApiCnxBase, char *strAqlStatement);

内部 Basic 语法

Function AmDbExecAql(strAqlStatement As String) As Long

应用的字段

版本: 4.1.0

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **strAqlStatement：此参数包含要执行的** AQL **查询。**

输出参数

- 0**：正常执行。**
- **非** 0 **值：错误代码。**

AmDbGetDate()

此函数返回日期形式的 AQL 查询结果。如果未返回查询结果，则返回值 0 且不会触发任何错误。

API 语法

long AmDbGetDate(long hApiCnxBase, char *strQuery);

内部 Basic 语法

Function AmDbGetDate(strQuery As String) As Date

应用的字段

版本: 3.5

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strQuery：此参数包含要恢复其结果的完整 AQL 查询。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

AmDbGetDouble()

此函数返回双精度数值格式的 AQL 查询结果。如果未返回查询结果，则返回值 0 且不会触发任何错误。

API 语法

double AmDbGetDouble(long hApiCnxBase, char *strQuery);

内部 Basic 语法

Function AmDbGetDouble(strQuery As String) As Double

应用的字段

版本: 3.5

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strQuery：此参数包含要恢复其结果的完整 AQL 查询。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmDbGetList()

此函数返回列表形式的 AQL 查询结果。AQL 查询可选择的元素数目最多为 99 个。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strQuery：此参数包含要执行的** AQL **查询。**
- **strColSep：此参数包含的字符可在函数返回结果中作为列分隔符。**
- **strLineSep：此参数包含的字符可在函数返回结果中作为行分隔符。**
- **strIdSep：此参数包含的字符可在函数返回结果中作为标识符分隔符。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmDbGetListEx()

此函数返回列表形式的 AQL **查询结果。与 AmDbGetList 函数不同，此函数不受** AQL **查询可选择元素数量的限制。**

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 3.5

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strQuery：此参数包含要执行的** AQL **查询。**
- **strColSep：此参数包含的字符可在函数返回结果中作为列分隔符。**
- **strLineSep：此参数包含的字符可在函数返回结果中作为行分隔符。**
- **strIdSep：此参数包含的字符可在函数返回结果中作为标识符分隔符。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

备注

如果 AmDbGetList 函数返回的数据包含用作列、行或标识分隔符的字符，则这些字符必须用反斜杠'\'转义。

我们推荐使用 UnEscapeSeparators 函数从由 AmDbGetList 返回的字符串中删除转义字符。

AmDbGetLong()

此函数返回 AQL 查询结果。如果未返回查询结果，则返回值 0 且不会触发任何错误。

API 语法

long AmDbGetLong(long hApiCnxBase, char *strQuery);

内部 Basic 语法

Function AmDbGetLong(strQuery As String) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strQuery：此参数包含要恢复其结果的完整 AQL 查询。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

下例将返回产品供应商的标识符：

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

AmDbGetPk()

根据 AQL 中的 WHERE 子句，此函数返回表的主键。如果未返回查询结果，则返回值 0 且不会触发任何错误。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strTableName：此参数包含要恢复主键的表的 SQL 名称。**
- **strWhere：AQL 查询中的 WHERE 子句。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

AmDbGetString()

此函数返回格式化字符串形式的AQL查询结果。AQL查询可选择的元素数目最多为99个。

警告:

不要使用此函数恢复单个字符串类型字段的值。此函数与 AmDbGetList 和 AmDbGetListEx 函数类似。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strQuery：此参数包含要执行的 AQL 查询。**
- **strColSep：此参数包含的字符可在最终字符串中作为列分隔符。**
- **strLineSep：此参数包含的字符可在最终字符串中作为行分隔符。**

输出参数

出现错误时，有两种可能性：

- 在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。
- 如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。

备注

在 API 语法中，lResult 参数必须包含生成的值的期望大小。

示例

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

将返回字符串：

```
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
```

在逗号前使用转义符 \。

使用 amDbGetString() 的相同查询将不会添加转义符，因为转义符不适合用于填充列表。例如：

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

将显示：

```
/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()

此函数返回字符串形式的 AQL 查询结果。与 AmDbGetString 函数不同，此函数不受 AQL 查询可选择元素的数量的限制。

警告:

不要使用此函数恢复单个字符串类型字段的值。此函数与 AmDbGetList **和** AmDbGetListEx **函数类似。**

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 3.5

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strQuery：此参数包含要执行的** AQL **查询。**
- **strColSep：此参数包含的字符可在最终字符串中作为列分隔符。**
- **strLineSep：此参数包含的字符可在最终字符串中作为行分隔符。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmDeadLine()

此函数根据日历、起始日期和经过的工作秒数，计算最终期限。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strCalendarSqlName：此参数包含工作期间的日历的** SQL **名称，该工作期间用作计算最终期限的基础。**
- **tmStart：此参数包含期间的起始日期。**
- **tsDuration：此参数包含自期间的起始日期开始的工作期间的秒数。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

下例将根据 SQL **名称为** "Calendar_Paris" **的日历，计算起始日期设置为** 1998 **年** 9 **月** 1 **日上午** 8 **点，秒数为** 450,000 **的期间的最终期限。**

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

此例将返回最终期限值，如 1998 **年** 9 **月** 22 **日下午** 6 **点。**

AmDecrementLogLevel()

此函数可将向导的完成页面中的日志窗口的层次结构提升一级。

内部 Basic 语法

Function AmDecrementLogLevel() As Long

应用的字段

版本: 3.5

	可用
AssetCenter API	
字段或链接的配置脚本	
"Script" 类型操作	
向导脚本	
向导的 FINISH.DO 脚本	✓

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmDefAssignee()

此函数搜索指定员工组的默认记录单主管的 ID **编号。**

API 语法

long AmDefAssignee(long hApiCnxBase, long lGroupId);

内部 Basic 语法

Function AmDefAssignee(lGroupId As Long) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **lGroupId：此参数包含员工组的** ID **编号。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

一般情况下，下例将返回员工组的默认记录单主管的标识符：

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

如下例所示，可以直接输入标识符的数值：

```
AmDefAssignee(24)
```

AmDefaultCurrency()

将返回在 AssetCenter 中使用的默认货币。

API 语法

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

内部 Basic 语法

Function AmDefaultCurrency() As String

应用的字段

版本: 3.5

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

AmDefEscalationScheme()

此函数将根据帮助台工作单的位置和严重度搜索默认的升级方案。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strLocFullName：此参数包含位置的全称。**
- **lSeverityLvl：此参数包含严重度的值。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

一般情况下，下例将根据位置和严重度返回默认的升级方案：

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

如下例所示，可以直接输入参数值：

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

AmDefGroup()

此函数将根据问题类型、位置和维护合同，返回默认帮助台组的 ID **号。**

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **lProblemClassId：此参数包含问题类型的** ID **号。**
- **strLocFullName：此参数包含位置的全称。**
- **lAssetMainCntId：此参数包含维护合同的** ID **号。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

备注

用于定义默认帮助台组的方法如下：

1. **此函数搜索与工作单的问题类型相关的帮助台组。**
2. **此函数将从这些组中搜索与到资产的"最近"位置（直接位置、其他父位置等，直到根位置）相关的帮助台组。**

3 **如果未找到这些组，并且** DBMS **支持双外部连接，此函数将搜索与位置不相关的组。**

有关可支持双外部连接的 DBMS **的列表的信息，请参考《**帮助台**》指南，**参考（帮助台）**章，**支持双外部连接的 DBMS**节。**

4 **如果** DBMS **支持双外部连接，此函数将从先前找到的组中选择与涵盖资产的维护合同链接的帮助台组。**

5 **如果未找到组，此函数将从问题的层次结构的问题类型级别开始，直到问题类型树的根位置重复执行步骤** 1、2、3 **和** 4。

示例

一般情况下，此函数将根据三个参数：问题类型、位置和维护合同，计算默认帮助台组的 ID **号。**

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

如下例所示，可以直接使用 ID **号输入参数值：**

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

AmDeleteLink()

此函数将删除记录的链接。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	

	可用
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiRecord：此参数包含的记录句柄中包含要删除的链接。**
- **strLinkName：此参数包含要删除的链接的 SQL 名称。**
- **hApiRecDest：此参数包含要删除的链接的目标记录的句柄。**

输出参数

- 0：**正常执行。**
- **非 0 值：错误代码。**

AmDeleteRecord()

此函数将删除数据库中的记录。

API 语法

long AmDeleteRecord(long hApiRecord);

内部 Basic 语法

Function AmDeleteRecord(hApiRecord As Long) As Long

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiRecord：此参数包含要删除的记录的句柄。**

输出参数

- 0：正常执行。
- 非 0 值：错误代码。

AmDisconnectTrace()

AmDisconnectTrace API **断开电缆链接表中的用户节点** (lEndId) **和主机节点** (lStartId) **之间的跟踪连接。如果每个节点都在跟踪的末端，将从电缆链接表中删除它。还将基于断开操作创建跟踪历史记录和跟踪操作条目。**

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **lStartId：此参数定义要断开的主机连接 ID。**
- **lEndId：此参数定义要断开的用户连接 ID。**
- **strComment：此参数包含要显示新连接和断开连接的字符串操作符。**
- **lCabTraceOutId：此参数是电缆跟踪输出 ID。**

输出参数

- **0：正常执行。**
- **非 0 值：错误代码。**

AmDuplicateRecord()

此函数可用于复制记录。

API 语法

long AmDuplicateRecord(long hApiRecord, long bInsert);

内部 Basic 语法

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

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiRecord：此参数包含要复制的记录的句柄。**
- **bInsert：此参数可用于指定立即** (=1) **插入复制的记录或不立即插入** (=0)。

输出参数

- 0：**正常执行。**
- **非** 0 **值：错误代码。**

AmEndOfNthBusinessDay()

根据日历，在特定的日期上指定某日的最后工时（由整数型参数lDayCount标识）。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **strCalendarSqlName：计算使用的日历名称。**
- **tmStart：计算使用的起始日期。**
- **lDayCount：用于计算的添加到 dStart 的完整工作日天数。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

AmEnumValList()

此函数将返回包含所有自定义逐项列表值的字符串。按字母顺序对不同的值进行排序并使用在 strLineSep 参数中指定的分隔符对这些值进行分隔。

如果逐项列表值包含用作分隔符的字符或 "\"，则使用 "\" 作为前缀。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓

	可用
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strEnumName：此参数包含要恢复的值所在的逐项列表的** SQL **名称。**
- **bNoCase：此参数可用于指定排序是区分大小写** (=1) **还是不区分大小写** (=0)**。**
- **strLineSep：此参数包含的字符可用于分隔逐项列表值。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

AmESDAddComputers()

内部 Basic 语法

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

应用的字段

版本: 5.0.0

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmESDCreateTask()

内部 Basic 语法

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

应用的字段

版本: 5.0.0

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmEvalScript()

此函数可用于从当前上下文中按其名称对脚本进行求值。此函数具有两种用途：

- **对系统脚本求值（默认值、强制等）**
- **从脚本库中调用函数。**

内部 Basic 语法

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

应用的字段

版本: 4.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	

输入参数

- **strScriptName：此参数包含要求值的脚本的名称。在第一种情况中，它是系统脚本的名称（DefVal 等）。在第二种情况中，它是脚本库的名称。**
- **strObject：此参数包含与脚本有关的对象。它可能是字段的 SQL 名称或从库中调用的函数的名称。**
- **strPath：此参数是可选的，用于指定到脚本值的上下文位置的路径（链接.连接.链接...）。在第二种情况中不使用此参数。**
- **...：从脚本库中调用函数时，可将参数传递给被调用的函数。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

备注

下面是可用的系统脚本名称列表：

- **对于表：IsValid、IsRelevant**
- **对于字段：DefVal、Mandatory、Historized、ReadOnly、Irrelevant**
- **对于链接：Historized、Filter、Irrelevant**
- **对于特征：DefVal、Mandatory、Available、Historized**

AmExecTransition()

此函数触发从当前页面的有效转换。

内部 Basic 语法

Function AmExecTransition(strTransName As String) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	
"Script" 类型操作	
向导脚本	
向导的 FINISH.DO 脚本	

输入参数

- **strTransName：此参数包含在向导脚本中定义的转换的名称。如果未找到此转换，则返回一个错误。如果转换无效，则不使用此函数（并且不会返回错误）。**

输出参数

- 0：**正常执行。**
- **非** 0 **值：错误代码。**

AmExecuteActionById()

此函数执行由其标识符标识的操作。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lActionId：此参数包含要执行的操作的标识符。**
- **strTableName：如果操作和上下文相关，此参数包含要在其上执行操作的表的** SQL **名称。如果在与上下文相关的操作中忽略此参数，则无法执行此函数。对于和上下文无关的操作，则不解释此参数，因此此参数是可选的。**
- **lRecordId：此参数包含与此操作有关的记录的标识符。对于和上下文无关的操作，则不解释此参数，因此此参数是可选的。**

输出参数

- 0**：正常执行。**
- **非** 0 **值：错误代码。**

AmExecuteActionByName()

此函数执行由其 SQL **名称标识的操作。**

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **strSqlName：此参数包含要执行的操作的** SQL **名称。**
- **strTableName：如果操作和上下文相关，此参数包含要在其上执行操作的表的** SQL **名称。如果在与上下文相关的操作中忽略此参数，则无法执行此函数。对于和上下文无关的操作，则不解释此参数，因此此参数是可选的。**
- **lRecordId：此参数包含与此操作有关的记录的标识符。对于和上下文无关的操作，则不解释此参数，因此此参数是可选的。**

输出参数

- 0**：正常执行。**
- **非** 0 **值：错误代码。**

AmExportDocument()

此函数可用于导出与记录相关的文档。

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **lDocId：此参数包含要导出的文档的标识符。**
- **strFileName：此参数包含要导出的文档的名称，它存储在"文档"表的** FileName **字段中。**

输出参数

- 0**：正常执行。**
- **非** 0 **值：错误代码。**

AmExportReport()

此函数可用于从数据库中导出 Crystal Report **报表。**

内部 Basic 语法

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

应用的字段

版本: 4.3.0

	可用
AssetCenter API	
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **lReportId：此参数包含要导出的** Crystal Report **报表的标识符。**
- **strFileName：此参数包含要进行导出的文件的完整路径。**

输出参数

- 0**：正常执行。**
- **非** 0 **值：错误代码。**

AmFindCable()

AmFindCable API **查找下一条在指定用户位置** (lUserId) **和主机位置** (lHostId) **之间运行的可用的电缆。此电缆必须是指定的电缆类型** (strCabType) **和电缆角色** (strCableRole)**。此电缆的状态还必须为"可用"。按电缆** ID **以升序对电缆进行排序，并仅选择比先前电缆** ID (lPrevCabId) **大的电缆。**

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **lPrevCableId：此参数是先前电缆的** ID。
- **strCabType：此参数定义要搜索的电缆类型。**
- **lUserId：此参数定义用户位置** ID。
- **lHostId：此参数定义主机位置** ID。
- **strCableRole：此参数包含要分配的电缆角色。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

AmFindDevice()

AmFindDevice API **查找指定位置** (lLocId) **的指定类型** (strDevType) **的设备。按设备** ID **以升序对设备进行排序，并仅选择比先前设备** ID (lPrevDeviceId) **大的设备。**

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **lPrevDeviceId：此参数定义先前搜索的设备** ID。**值** 0 **用于启动搜索。**
- **strDeviceType：此参数定义要分配的设备类型。**
- **lLocationId：此参数包含要搜索的位置** ID。

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmFindRootLink()

此函数可用于恢复跟踪的根链接。

API 语法

long AmFindRootLink(long hApiCnxBase, long lLinkId);

内部 Basic 语法

Function AmFindRootLink(lLinkId As Long) As Long

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **lLinkId：此参数包含与此操作有关的链接的标识符。**

输出参数

此函数返回根链接的标识符。

AmFindTermDevice()

AmFindTermDevice API **查找指定端接区** (lTermField) **中指定的电缆角色** (strCableRole) **的下一条可用电缆。按序号以升序对设备进行排序，并仅选择比先前模型** (strPrevTermSeq) **大的资产。同时，对基于针脚的设备** (bPinBased = 1)**，根据设备上剩余的针脚总数，检查所需的针脚总数** (iPinPortCount)**。对基于端口的设备** (bPinBased =0) **进行检查以确保设备上至少还有一个剩余端口，并且通过检查标记（**bCheckAvail = 0 - **用户设备，**bCheckAvail = 1 - **主机设备）确保剩余端口具有可用的主机或用户端。**

API 语法

long AmFindTermDevice(long hApiCnxBase, long iPrevTermSeq, long lTermFieldId, char *strCableRole, long bPinBased, long iPinPortCount, long bCheckAvail);

内部 Basic 语法

Function AmFindTermDevice(iPrevTermSeq As Long, lTermFieldId As Long, strCableRole As String, bPinBased As Long, iPinPortCount As Long, bCheckAvail As Long) As Long

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **iPrevTermSeq：此参数是先前搜索的端接区的序列。值 0 用于启动搜索。**
- **lTermFieldId：此参数是端接区 ID。**
- **strCableRole：此参数包含要分配的电缆角色。**
- **bPinBased：此参数确定设备基于针脚或端口。**
- **iPinPortCount：对基于针脚的设备，此参数是创建虚拟端口所需的针脚总数。对基于端口的设备，因为对每个所需的端口调用此 API，所以此参数为 1。**
- **bCheckAvail：此参数用于确定需要使用的端口所在的端。**
  - **0=用户设备，检查主机端是否可用**
  - **0=主机设备，检查用户端是否可用**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

AmFindTermField()

AmFindTermField API 从指定区域 (lLocId) 查找提供指定负载 (lDutyId) 的端接区。它将在指定位置继续查找提供了指定负载的其他端接区（如果 lTermFieldId 大于 0）。

API 语法

long AmFindTermField(long hApiCnxBase, long lDutyId, long lLocationId, long lPrevTermFieldId);

内部 Basic 语法

Function AmFindTermField(lDutyId As Long, lLocationId As Long, lPrevTermFieldId As Long) As Long

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **lDutyId：此参数定义要分配的负载。**
- **lLocationId：此参数包含要搜索的位置** ID。
- **lPrevTermFieldId：此参数是端接区** ID。

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

AmFlushTransaction()

此函数清除代理的任务列表（与数据库"提交"操作类似）。

API 语法

long AmFlushTransaction(long hApiCnxBase);

内部 Basic 语法

Function AmFlushTransaction() As Long

应用的字段

版本: 4.3.0

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输出参数

- 0：**正常执行。**
- **非** 0 **值：错误代码。**

AmFormatCurrency()

此函数按指定币种显示货币值。还显示标准的货币符号。

API 语法

long AmFormatCurrency(double dAmount, char *strCurrency, char *pstrDisplay, long lDisplay);

内部 Basic 语法

Function AmFormatCurrency(dAmount As Double, strCurrency As String) As String

应用的字段

版本: 4.3.0

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **dAmount：此参数包含要显示的货币值。**
- **strCurrency：此参数包含用于操作的币种。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
RetVal=amFormatCurrency(500, "USD")
```

本例中显示：

```
US$500.00
```

AmFormatLong()

此函数将字符串中的标记替换为长整型变量中包含的值。

API 语法

long AmFormatLong(long hApiCnxBase, long lNumber, char *strFormat, char *pstrResult, long lResult);

内部 Basic 语法

Function AmFormatLong(lNumber As Long, strFormat As String) As String

应用的字段

版本: 4.3.0

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **INumber：此参数包含要插入到 strFormat 参数中包含的字符串中的长整型变量。**
- **strFormat：此参数包含要处理的字符串。将所有 "%d" 类型的标记替换为 INumber 参数中包含的值。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGeneratePlanningData()

此函数可用于生成图形化的 PLANNER 查看器。

内部 Basic 语法

Function AmGeneratePlanningData(strTableSqlName As String, strProperties As String, strIds As String) As String

应用的字段

版本: 4.3.0

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **strTableSqlName：此参数包含表的 SQL 名称，该表包含生成计划的数据。**
- **strProperties：此参数包含已创建计划的属性。**

  注:

  有关这些属性的语法的更多信息，请参考《管理》指南中的"参考：PLANNER查看器的语法参数"页面。

- **strIds：此参数包含记录的标识符列表（逗号分隔），使用这些记录的数据创建计划。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGenSqlName()

此函数根据典型字符串生成有效的 SQL 名称。使用下划线 ("_") 代替空格。根据其名称为特征值定义其 SQL 名称默认值时，此函数特别有用。

API 语法

long AmGenSqlName(char *return, long lreturn, char *strText);

内部 Basic 语法

Function AmGenSqlName(strText As String) As String

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strText：此参数用于生成** SQL **名称的字符串。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

下例将定义 AssetCenter **数据库中称为** "Label" **的对象的** SQL **名称默认值：**

```
RetVal=AmGenSQLName([Label])
```

AmGetCatRef()

此函数搜索指定模型的非目录参考（考虑有效日期）。需要遵守以下规则：

- CatProduct.lModelId=lModelId
- CatProduct.lParentId=0

此函数按照其优先级返回非实时创建的参考。如果未找到参考，并且将参数bCreate设置为 "1"，**将创建新的非目录参考和产品（指向模型）。**

API 语法

long AmGetCatRef(long hApiCnxBase, long lModelId, long bCreate);

内部 Basic 语法

Function AmGetCatRef(lModelId As Long, bCreate As Long) As Long

应用的字段

版本: 4.1.0

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lModelId：此参数包含与此操作有关的模型的** ID。
- **bCreate：此参数可用于在搜索未返回结果时，指定是否应创建非目录参考。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

备注

注:

无需为此函数指定供应商。因为仅对非目录参考执行此搜索，与供应商无关。

AmGetCatRefFromCatProduct()

除了可搜索执行产品之外，此函数与 amGetCatRef 函数相同。

API 语法

long AmGetCatRefFromCatProduct(long hApiCnxBase, long lCatProductId, long bCreate);

内部 Basic 语法

Function AmGetCatRefFromCatProduct(lCatProductId As Long, bCreate As Long) As Long

应用的字段

版本: 4.1.0

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lCatProductId：此参数包含与此操作有关的产品的** ID。
- **bCreate：此参数可用于在搜索未返回结果时，指定是否应创建非目录参考。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetComputeString()

此函数根据模板返回指定记录的描述字符串。

API 语法

long AmGetComputeString(long hApiCnxBase, char *strTableName, long lRecordId, char *strTemplate, char *pstrComputeString, long lComputeString);

内部 Basic 语法

Function AmGetComputeString(strTableName As String, lRecordId As Long, strTemplate As String) As String

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strTableName：此参数包含记录所在表的** SQL **名称，从该记录中恢复描述字符串。**
- **lRecordId：此参数包含表中记录的标识符。**
- **strTemplate：此参数包含用作描述字符串的模板（字符串格式）。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
RetVal = amGetComputeString("amEmplDept", [lEmplDeptId], "[Name], [FirstName]")
```

AmGetCurrentNTDomain()

此函数返回当前登录所在的 NT **域的名称。**

API 语法

long AmGetCurrentNTDomain(char *pstrDomain, long lDomain);

内部 Basic 语法

Function AmGetCurrentNTDomain() As String

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
RetVal = amGetCurrentNTDomain()
```

AmGetCurrentNTUser()

此函数可用于获取用户连接的 Windows（NT **或** 2000）**系统的登录名。**

API 语法

long AmGetCurrentNTUser(char *pstrUser, long lUser);

内部 Basic 语法

Function AmGetCurrentNTUser() As String

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetFeat()

此函数将根据表名的句柄来创建特征对象，并返回创建的特征对象的句柄。

API 语法

long AmGetFeat(long hApiTable, long lPos);

内部 Basic 语法

Function AmGetFeat(hApiTable As Long, lPos As Long) As Long

应用的字段

版本: 3.5

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiTable：此参数包含表的句柄。**
- **lPos：此参数包含表中特征值的位置。**

AmGetFeatCount()

此函数将返回表的特征数，这些特征在 hApiTable 参数中指定。

API 语法

long AmGetFeatCount(long hApiTable);

内部 Basic 语法

Function AmGetFeatCount(hApiTable As Long) As Long

应用的字段

版本: 3.5

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiTable：此参数包含表的句柄。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetField()

此函数将根据查询、记录或表的句柄创建特征对象并返回创建的特征对象的句柄。

API 语法

long AmGetField(long hApiObject, long lPos);

内部 Basic 语法

Function AmGetField(hApiObject As Long, lPos As Long) As Long

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiObject：此参数包含查询、记录或表的句柄。**
- **lPos：此参数包含对象中字段的位置（其索引）。**

AmGetFieldCount()

此函数返回当前对象中包含的字段的数量。

API 语法

long AmGetFieldCount(long hApiObject);

内部 Basic 语法

Function AmGetFieldCount(hApiObject As Long) As Long

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiObject：此参数包含有效的记录、查询或表的句柄。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetFieldDateOnlyValue()

此函数返回当前对象中包含的字段的值。返回的值为"日期"格式（通过外部工具返回时，是长整型）。与 AmGetFieldDateValue 函数不同，此函数仅返回"日期"部分的值，而忽略"时间"部分。

API 语法

long AmGetFieldDateOnlyValue(long hApiObject, long lFieldPos);

内部 Basic 语法

Function AmGetFieldDateOnlyValue(hApiObject As Long, lFieldPos As Long) As Date

应用的字段

版本: 4.3.0

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiObject：此参数包含查询或记录的句柄。**
- **lFieldPos：此参数包含当前对象中的字段的数量。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetFieldDateValue()

此函数返回当前对象中包含的字段的值。返回的值为"日期"格式（通过外部工具返回时，是长整型）。

API 语法

long AmGetFieldDateValue(long hApiObject, long lFieldPos);

内部 Basic 语法

Function AmGetFieldDateValue(hApiObject As Long, lFieldPos As Long) As Date

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiObject：此参数包含查询或记录的句柄。**
- **lFieldPos：此参数包含当前对象中的字段的数量。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetFieldDescription()

此函数返回字符串"字符串"格式的，由句柄标识的字段的长描述字符串。

API 语法

long AmGetFieldDescription(long hApiField, char *pstrBuffer, long lBuffer);

内部 Basic 语法

Function AmGetFieldDescription(hApiField As Long) As String

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiField：此参数包含要获取其长描述字符串的字段的有效句柄。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetFieldDoubleValue()

此函数返回当前对象中包含的字段的值。返回的值为"双精度"格式。

API 语法

double AmGetFieldDoubleValue(long hApiObject, long lFieldPos);

内部 Basic 语法

Function AmGetFieldDoubleValue(hApiObject As Long, lFieldPos As Long) As Double

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓

	可用
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiObject：此参数包含查询或记录的句柄。**
- **lFieldPos：此参数包含当前对象中的字段的数量。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetFieldFormat()

当涉及到的字段的 "UserType" **值为如下类型时（比较** "database.txt" **文件），此函数很有用：**

- **系统逐项列表**
- **逐项列表**
- **时间间隔**
- **表或字段名**

此函数返回格式为 "UserType" **的值，也即：**

UserType	Format returned by the function
System itemized list	List of system-itemized list entries.
Itemized list	Name of the itemized list associated to the field.
Time span	Display format.
Table or field name	SQL name of the field that stores the SQL name of the table.

API 语法

long AmGetFieldFormat(long hApiField, char *pstrBuffer, long lBuffer);

内部 Basic 语法

Function AmGetFieldFormat(hApiField As Long) As String

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiField：此参数包含要获取其** "UserType" **的字段的有效句柄。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetFieldFormatFromName()

此函数根据其名称返回 "UserType" **格式的字段。**

API 语法

long AmGetFieldFormatFromName(long hApiCnxBase, char *strTableName, char *strFieldName, char *pFieldFormat, long lpFieldFormat);

内部 Basic 语法

Function AmGetFieldFormatFromName(strTableName As String, strFieldName As String) As String

应用的字段

版本: 3.5

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strTableName：此参数包含与此操作有关的字段所在表的** SQL **名称。**
- **strFieldName：此参数包含字段的** SQL **名称。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetFieldFromName()

此函数根据名称创建字段对象并返回创建的字段对象的句柄。

API 语法

long AmGetFieldFromName(long hApiObject, char *strName);

内部 Basic 语法

Function AmGetFieldFromName(hApiObject As Long, strName As String) As Long

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiObject：此参数包含查询、记录或表的句柄。**
- **strName：此参数包含字段名称。**

AmGetFieldLabel()

此函数返回字符串"字符串"格式的，由句柄标识的字段的标签。

API 语法

long AmGetFieldLabel(long hApiField, char *pstrBuffer, long lBuffer);

内部 Basic 语法

Function AmGetFieldLabel(hApiField As Long) As String

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiField：此参数包含要获取其标签的字段的有效句柄。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetFieldLabelFromName()

此函数根据其 SQL **名称返回字段的标签。**

API 语法

long AmGetFieldLabelFromName(long hApiCnxBase, char *strTableName, char *strFieldName, char *pFieldLabel, long lpFieldLabel);

内部 Basic 语法

Function AmGetFieldLabelFromName(strTableName As String, strFieldName As String) As String

应用的字段

版本: 3.5

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strTableName：此参数包含与此操作有关的字段所在表的** SQL **名称。**
- **strFieldName：此参数包含字段的** SQL **名称。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetFieldLongValue()

此函数返回当前对象中包含的字段的值。

API 语法

long AmGetFieldLongValue(long hApiObject, long lFieldPos);

内部 Basic 语法

Function AmGetFieldLongValue(hApiObject As Long, lFieldPos As Long) As Long

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiObject：此参数包含查询或记录的句柄。**
- **lFieldPos：此参数包含当前对象中的字段的数量。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

备注

注:

如果使用此函数恢复日期、时间或日期+时间类型字段的值，此函数返回的长整型值将表示自 1970 年 1 月 1 日 00:00:00 以来的秒数值。

AmGetFieldName()

此函数返回当前对象中包含的字段的名称。

API 语法

long AmGetFieldName(long hApiObject, long lFieldPos, char *pstrBuffer, long lBuffer);

内部 Basic 语法

Function AmGetFieldName(hApiObject As Long, lFieldPos As Long) As String

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiObject：此参数包含查询、记录或表的句柄。**
- **lFieldPos：此参数包含当前对象中的字段的数量。例如，值 "0" 表示第一个字段。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetFieldRights()

此函数返回用户对当前对象中字段的权限。返回的这些权限为字符串格式，包含三种字符，分别指定了读取/插入/更新权限：

- "r"：**表示用户有权读取数据。**
- "i"：**表示用户有权插入数据。**
- "u"：**表示用户有权更新数据。**

例如，对只读字段，此函数将返回值 "r "。

API 语法

long AmGetFieldRights(long hApiObject, long lFieldPos, char *pstrBuffer, long lBuffer);

内部 Basic 语法

Function AmGetFieldRights(hApiObject As Long, lFieldPos As Long) As String

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiObject：此参数包含查询、记录或表的句柄。**
- **lFieldPos：此参数包含当前对象中的字段的数量。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetFieldSize()

此函数返回字段的大小值。

API 语法

long AmGetFieldSize(long hApiField);

内部 Basic 语法

Function AmGetFieldSize(hApiField As Long) As Long

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiField：此参数包含要获取其大小的字段的句柄。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetFieldSqlName()

此函数返回字符串"字符串"格式的，由句柄标识的字段的 SQL 名称。

API 语法

long AmGetFieldSqlName(long hApiField, char *pstrBuffer, long lBuffer);

内部 Basic 语法

Function AmGetFieldSqlName(hApiField As Long) As String

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiField：此参数包含要获取其 SQL 名称的字段的有效句柄。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetFieldStrValue()

此函数返回当前对象中包含的字段的值。此值为字符串格式。

警告：通过 AssetCenter API 使用此函数时，需要另外的参数 pszBuffer 和 lBuffer，这两个参数分别定义了存储恢复的字符串的缓冲区的字符串，和此缓冲区的大小。必须格式化 pszBuffer 字符串（使用字符填充），其大小必须由 lBuffer 定义。下面的这部分代码不正确，未定义用作缓冲区的字符串的大小：

```
Dim strBuffer as String
Dim lRec as Long
Dim lBuffer as Long
lBuffer=20
lRec=AmGetFieldStrValue(1, 0, strBuffer, lBuffer)
```

下面是修正以后的这部分代码：

```
Dim strBuffer as String
Dim lRec as Long
Dim lBuffer as Long
strBuffer=String(21, " ") ' The buffer is set to 21 characters (" ")
lBuffer=20
lRec=AmGetFieldStrValue(1, 0, strBuffer, lBuffer)
```

使用 "String" 函数格式化缓冲区时，请勿使用 "0" 作为填充字符。在调用 AmGetFieldStrValue 函数之前要确定缓冲区的大小（特别是此函数处于循环中并且总是使用同一个字符串作为缓冲区时）。

API 语法

long AmGetFieldStrValue(long hApiObject, long lFieldPos, char *pstrBuffer, long lBuffer);

内部 Basic 语法

Function AmGetFieldStrValue(hApiObject As Long, lFieldPos As Long) As String

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiObject：此参数包含查询或记录的句柄。**
- **lFieldPos：此参数包含当前对象中的字段的数量。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetFieldType()

此函数返回字段的类型。

API 语法

long AmGetFieldType(long hApiField);

内部 Basic 语法

Function AmGetFieldType(hApiField As Long) As Long

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiField：此参数包含要获取其类型的字段的句柄。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

备注

注:

下面的表列出了 AmGetFieldType 函数返回的各种类型字段的值：

Valeurs retournées	Type de champ correspondant
0	Non défini
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()

此函数返回由句柄标识的字段（与 database.txt **文件相比）的** "UserType"**，其格式为长整型。对于字段，返回的有效值可概括如下：**

Stored value	Plain-text value
0	Default
1	Number
2	Yes/ No
3	Money
4	Date
5	Date+Time
7	System itemized list
8	Custom itemized list
10	Percentage
11	Time span
12	Table or field SQL name

对于链接，返回的有效值可概括如下：

Stored value	Plain-text value
0	Normal
1	Comment
2	Image
3	History
4	Feature value

在 4.0.0 **版本之前，对于链接，此函数总是返回** 0。**从** AssetCenter 4.1.0 **版本开始，对于链接，此函数将返回下列值之一：**

- 0：**普通**
- 1：**备注**
- 2：**图像**
- 3：**历史记录**
- 5：**脚本**

API 语法

long AmGetFieldUserType(long hApiField);

内部 Basic 语法

Function AmGetFieldUserType(hApiField As Long) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiField：此参数包含要获取其** "UserType" **的字段的有效句柄。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetForeignKey()

恢复链接的外键的句柄，通过其句柄标识此句柄自身。

API 语法

long AmGetForeignKey(long hApiField);

内部 Basic 语法

Function AmGetForeignKey(hApiField As Long) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiField：此参数包含与此操作有关的链接的句柄。**

AmGetIndex()

此函数将根据查询、记录或表的句柄创建索引对象并返回创建的索引对象的句柄。

API 语法

long AmGetIndex(long hApiTable, long lPos);

内部 Basic 语法

Function AmGetIndex(hApiTable As Long, lPos As Long) As Long

应用的字段

版本: 3.5

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiTable：此参数包含表的句柄。**
- **lPos：此参数包含表中索引的位置。**

AmGetIndexCount()

此函数将返回在 hApiTable 参数中指定的表中包含的索引的数量。

API 语法

long AmGetIndexCount(long hApiTable);

内部 Basic 语法

Function AmGetIndexCount(hApiTable As Long) As Long

应用的字段

版本: 3.5

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiTable：此参数包含表的句柄。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetIndexField()

此函数返回由其在索引中位置标识的字段上的句柄（索引的 lpos th **字段）。**

API 语法

long AmGetIndexField(long hApiIndex, long lPos);

内部 Basic 语法

Function AmGetIndexField(hApiIndex As Long, lPos As Long) As Long

应用的字段

版本: 3.5

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiIndex：此参数包含与此操作有关的索引上的有效句柄。**
- **lPos：此参数包含索引中字段的位置。**

AmGetIndexFieldCount()

此函数返回组成索引的字段的数量。

API 语法

long AmGetIndexFieldCount(long hApiIndex);

内部 Basic 语法

Function AmGetIndexFieldCount(hApiIndex As Long) As Long

应用的字段

版本: 3.5

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiIndex：此参数包含与此操作有关的索引上的有效句柄。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetIndexFlags()

此函数返回索引的参数。

API 语法

long AmGetIndexFlags(long hApiIndex);

内部 Basic 语法

Function AmGetIndexFlags(hApiIndex As Long) As Long

应用的字段

版本: 3.5

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiIndex：此参数包含与此操作有关的索引上的有效句柄。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

备注

通过与下面的值进行逻辑或 (OR) **运算产生此函数返回的值：**

- 1：**索引已授权非唯一记录，**
- 2：**索引授权空值，**
- 4：**索引不区分大小写。**

因此，如果函数返回 3，**就可以推断索引可接受非唯一记录和空值** (1 OR 2 = 3)。

AmGetIndexName()

此函数返回索引的名称。

API 语法

long AmGetIndexName(long hApiIndex, char *pstrBuffer, long lBuffer);

内部 Basic 语法

Function AmGetIndexName(hApiIndex As Long) As String

应用的字段

版本: 3.5

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiIndex：此参数包含要获取其名称的索引的有效句柄。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetLink()

此函数将根据表的句柄创建链接对象并返回创建的链接对象的句柄。

API 语法

long AmGetLink(long hApiTable, long lPos);

内部 Basic 语法

Function AmGetLink(hApiTable As Long, lPos As Long) As Long

应用的字段

版本: 3.02

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiTable：此参数包含表的句柄。**
- **lPos：此参数包含当前对象中的链接（其索引）的位置。**

AmGetLinkCardinality()

此函数返回链接的基数。

API 语法

long AmGetLinkCardinality(long hApiField);

内部 Basic 语法

Function AmGetLinkCardinality(hApiField As Long) As Long

应用的字段

版本: 3.5

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiField：此参数包含要获取其基数的链接的句柄。**

输出参数

- 1：**链接的基数是** 1-1。
- 2：**链接的基数是** 1-n。

AmGetLinkCount()

此函数返回当前表中包含的链接的数量。

API 语法

long AmGetLinkCount(long hApiTable);

内部 Basic 语法

Function AmGetLinkCount(hApiTable As Long) As Long

应用的字段

版本: 3.02

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiTable：此参数包含有效表的句柄。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetLinkDstField()

此函数返回由 hApiField 参数定义的链接指向的字段（外键）。

API 语法

long AmGetLinkDstField(long hApiField);

内部 Basic 语法

Function AmGetLinkDstField(hApiField As Long) As Long

应用的字段

版本: 3.5

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiField：此参数包含与此操作有关的链接的句柄。**

AmGetLinkFeatureValue()

返回"链接"类型的特征值。

API 语法

long AmGetLinkFeatureValue(long hApiObject, long lFieldPos, long lRecordId);

内部 Basic 语法

Function AmGetLinkFeatureValue(hApiObject As Long, lFieldPos As Long, lRecordId As Long) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiObject：此参数包含查询或记录的句柄。**
- **lFieldPos：此参数包含当前对象中的字段的位置。**
- **lRecordId：此参数包含要恢复的特征值所在的记录的标识符。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
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()

此函数根据名称创建链接对象并返回创建的对象的句柄。

API 语法

long AmGetLinkFromName(long hApiTable, char *strName);

内部 Basic 语法

Function AmGetLinkFromName(hApiTable As Long, strName As String) As Long

应用的字段

版本: 3.02

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiTable：此参数包含表的句柄。**
- **strName：此参数包含链接的 SQL 名称。**

AmGetLinkType()

此函数链接的类型。

API 语法

long AmGetLinkType(long hApiField);

内部 Basic 语法

Function AmGetLinkType(hApiField As Long) As Long

应用的字段

版本: 3.02

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiField：此参数包含要获取其类型的链接的句柄。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetMainField()

此函数创建的字段对象与指定表中的主字段对应。它将返回所创建的字段的句柄。

API 语法

long AmGetMainField(long hApiTable);

内部 Basic 语法

Function AmGetMainField(hApiTable As Long) As Long

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiTable：此参数包含要获取其主字段的表的句柄。**

AmGetMemoField()

此函数创建的字段对象与指定表中的Memo-type字段相对应。它将返回所创建的字段的句柄。

API 语法

long AmGetMemoField(long hApiTable);

内部 Basic 语法

Function AmGetMemoField(hApiTable As Long) As Long

应用的字段

版本: 4.1.0

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiTable：此参数包含要获取其备注字段的表的句柄。**

AmGetNextAssetPin()

AmGetNextAssetPin API **将在设备** (lAssetId) **上查找下一个可用的针脚。按其序号排序针脚。根据端口方向** (iPinPortDir)**，按照升序** (iPinPortDir = 0) **或降序** (iPinPortDir = 1) **对可用的针脚进行排序。**

API 语法

long AmGetNextAssetPin(long hApiCnxBase, long lAssetId, long bPinPortDir, long iPrevPinSeq);

内部 Basic 语法

Function AmGetNextAssetPin(lAssetId As Long, bPinPortDir As Long, iPrevPinSeq As Long) As Long

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **lAssetId：此参数是设备** ID。
- **bPinPortDir：此参数确定搜索的方向。**
  - 0=**升序**
  - 1=**降序**
- **iPrevPinSeq**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**

- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetNextAssetPort()

AmGetNextAssetPort API **查找提供指定服务** (lDutyId) **或不提供任何服务的设备** (lAssetId) **上的下一可用端口。端口的状态必须为"可用"。布尔标记值用于指明是否需要检查端口的用户端** (bCheckUser) **和/或主机端** (bCheckHost)**。如果相应的布尔标记值为** true**，则函数将比较用户值** (bUserAvail) **和/或主机值** (bHostAvail)**。按序号对端口进行排序。根据端口方向** (iPinPortDir)**，按照升序** (iPinPortDir = 0) **或降序** (iPinPortDir = 1) **对可用的端口进行排序。**

API 语法

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 语法

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

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **lAssetId：此参数定义要搜索的设备** ID。
- **lCabCnxTypeId：此参数定义端口的电缆连接类型。**
- **lDutyId：此参数是端口的负载。**

- **bCheckUser：此参数是用于检查用户端的标记。**
- **bCheckHost：此参数是用于检查主机端的标记。**
- **bUserAvail：此参数定义要检查的用户端可用性状态。**
- **bHostAvail：此参数定义要检查的主机端可用性状态。**
- **bPinPortDir：此参数定义要检查的针脚方向。**
  - 0=**升序**
  - 1=**降序**
- **iPrevPortSeq**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetNextCableBundle()

AmGetNextCableBundle API **查找提供指定服务** (lDutyId) **或不提供任何服务的电缆** (lCableId) **上的下一可用线束。线对的状态必须为"可用"。布尔标记值用于指明是否需要检查线束的用户端** (bCheckUser) **和/或主机端** (bCheckHost)**。如果相应的布尔标记值为** true**，则函数将比较用户值** (bUserAvail) **和/或主机值** (bHostAvail)**。**

API 语法

long AmGetNextCableBundle(long hApiCnxBase, long lCableId, long lDutyId, long bCheckUser, long bCheckHost, long bUserAvail, long bHostAvail);

内部 Basic 语法

Function AmGetNextCableBundle(lCableId As Long, lDutyId As Long, bCheckUser As Long, bCheckHost As Long, bUserAvail As Long, bHostAvail As Long) As Long

应用的字段

版本: 4.00

	可用
AssetCenter API	✓

	可用
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **lCableId：此参数包含要检查的电缆的** ID。
- **lDutyId：此参数定义要分配的负载。**
- **bCheckUser：此参数指明要检查线束的用户端连接。**
- **bCheckHost：此参数指明要检查线束的主机端连接。**
- **bUserAvail：此参数定义要分配的用户端连接状态。**
- **bHostAvail：此参数定义要分配的主机端连接状态。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetNextCablePair()

AmGetNextCablePair API **查找指定类型** (lPairType) **的电缆** (lCableId) **中的下一可用电缆对。按电缆对的** ID **进行排序。**

API 语法

long AmGetNextCablePair(long hApiCnxBase, long lCableId, long lPairTypeId, long iStartPairSeq);

内部 Basic 语法

Function AmGetNextCablePair(lCableId As Long, lPairTypeId As Long, iStartPairSeq As Long) As Long

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **lCableId：此参数包含要搜索的电缆的** ID。
- **lPairTypeId：此参数定义要分配的电缆对类型。**
- **iStartPairSeq**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetNTDomains()

此函数可用于获取已连接到数据库的用户所在的域。

API 语法

long AmGetNTDomains(char *pstrDomains, long lDomains);

内部 Basic 语法

Function AmGetNTDomains() As String

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetNTMachinesInDomain()

此函数可用于获取形式为一列的域中的计算机列表（用逗号分隔计算机名）。如果域为空，则函数返回 ERR_CANCEL(2)**，但是执行过程不中断。**

API 语法

long AmGetNTMachinesInDomain(char *strDomain, char *pstrMachines, long lMachines, long bUseDC);

内部 Basic 语法

Function AmGetNTMachinesInDomain(strDomain As String, bUseDC As Long) As String

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **strDomain：此参数包含要浏览的域的名称。**
- **bUseDC：如果此参数设置为 1，则函数将查询域控制器以获取计算机列表。如果此参数设置为 0（默认值），则函数将使用系统函数库查找计算机列表。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetNTUsersInDomain()

此函数可用于获取域中的用户列表。返回的列表为两列（登录名，全称）。"|"用作列分隔符，","用作行分隔符。

API 语法

long AmGetNTUsersInDomain(char *strDomain, char *pstrUsers, long lUsers);

内部 Basic 语法

Function AmGetNTUsersInDomain(strDomain As String) As String

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **strDomain：此参数包含要浏览的域的名称。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetPOLinePrice()

此函数可用于计算订单行的单价。

API 语法

double AmGetPOLinePrice(long hApiCnxBase, long lPOrdLineId);

内部 Basic 语法

Function AmGetPOLinePrice(lPOrdLineId As Long) As Double

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **lPOrdLineId：此参数包含订单行的标识符。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetPOLinePriceCur()

此函数可用于查找订单行的货币代码

API 语法

long AmGetPOLinePriceCur(long hApiCnxBase, long lPOrdLineId, char *pstrPrice, long lPrice);

内部 Basic 语法

Function AmGetPOLinePriceCur(lPOrdLineId As Long) As String

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **lPOrdLineId：此参数包含订单行的标识符。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetPOLineReference()

此函数可用于获取与采购订单行对应的目录参考描述。

API 语法

long AmGetPOLineReference(long hApiCnxBase, long lPOrdLineId, char *pstrRef, long lRef);

内部 Basic 语法

Function AmGetPOLineReference(lPOrdLineId As Long) As String

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **lPOrdLineId：此参数包含订单行的标识符。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetRecordFromMainId()

此函数将返回由包含此记录的表的主键值标识的记录的 ID **号。**

API 语法

long AmGetRecordFromMainId(long hApiCnxBase, char *strTable, long lId);

内部 Basic 语法

Function AmGetRecordFromMainId(strTable As String, lId As Long) As Long

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **strTable：此参数包含与此操作有关的记录所在表的 SQL 名称。**
- **lId：此参数包含用于此记录的表的主键值。**

备注

除了未找到表的情况之外，此函数将系统地返回记录句柄。如果未在指定的表中找到记录，则在执行每个使用此函数返回句柄的函数的过程中将出现错误。

AmGetRecordHandle()

此函数将返回作为当前查询（通过其句柄标识）结果的记录的句柄。此记录可用于在数据库中写入。仅当查询包含记录的主键时，此函数才有用。

API 语法

long AmGetRecordHandle(long hApiQuery);

内部 Basic 语法

Function AmGetRecordHandle(hApiQuery As Long) As Long

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiQuery：此参数包含查询对象的有效句柄。**

AmGetRecordId()

此函数将返回由其句柄标识的记录的 ID 号。在插入记录的情况下，此值为 0。

API 语法

long AmGetRecordId(long hApiRecord);

内部 Basic 语法

Function AmGetRecordId(hApiRecord As Long) As Long

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiRecord：此参数包含要获取其 ID 号的记录的有效句柄。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**

- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetRelDstField()

此函数返回链接的目标字段上的句柄。

API 语法

long AmGetRelDstField(long hApiField);

内部 Basic 语法

Function AmGetRelDstField(hApiField As Long) As Long

应用的字段

版本: 3.5

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiField：此参数包含与此操作有关的链接上的有效句柄。**

AmGetRelSrcField()

此函数返回链接的源字段上的句柄。

API 语法

long AmGetRelSrcField(long hApiField);

内部 Basic 语法

Function AmGetRelSrcField(hApiField As Long) As Long

应用的字段

版本: 3.5

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiField：此参数包含与此操作有关的链接上的有效句柄。**

AmGetRelTable()

此函数返回 N-N 链接关系表上的句柄。

API 语法

long AmGetRelTable(long hApiField);

内部 Basic 语法

Function AmGetRelTable(hApiField As Long) As Long

应用的字段

版本: 3.5

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiField：此参数包含与此操作有关的链接上的有效句柄。**

输出参数

出现错误时，此函数返回非有效句柄 (0)。

AmGetReverseLink()

此函数将返回通过 hApiField 参数中包含的句柄指定的反向链接的句柄。

API 语法

long AmGetReverseLink(long hApiField);

内部 Basic 语法

Function AmGetReverseLink(hApiField As Long) As Long

应用的字段

版本: 3.02

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiField：此参数包含要获取其反向链接的链接的句柄。**

AmGetScriptValue()

API 语法

long AmGetScriptValue(long hApiObject, char *strScriptName, char *strObject, char *strPath);

内部 Basic 语法

Function AmGetScriptValue(hApiObject As Long, strScriptName As String, strObject As String, strPath As String) As Variant

应用的字段

版本: ?

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetSelfFromMainId()

返回指定表中记录的描述字符串。

API 语法

long AmGetSelfFromMainId(long hApiCnxBase, char *strTableName, long lId, char *pstrRecordDesc, long lRecordDesc);

内部 Basic 语法

Function AmGetSelfFromMainId(strTableName As String, lId As Long) As String

应用的字段

版本: 3.5

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strTableName：此参数包含与此操作有关的记录所在表的** SQL **名称。**
- **lId：此参数包含与此操作有关的** ID **号。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetSourceTable()

将返回在 hApiField 参数中指定的链接的源表的句柄。

API 语法

long AmGetSourceTable(long hApiField);

内部 Basic 语法

Function AmGetSourceTable(hApiField As Long) As Long

应用的字段

版本: 3.02

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiField：此参数包含要获取其源表的链接的有效句柄。**

输出参数

出现错误时，此函数返回非有效句柄 (0)。

AmGetTable()

此函数返回当前连接中通过其位置（编号）标识的表的句柄。

API 语法

long AmGetTable(long hApiCnxBase, long lPos);

内部 Basic 语法

Function AmGetTable(lPos As Long) As Long

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **IPos：此参数包含当前连接中表的位置。其值由"0"和AmGetTableCount构成。**

输出参数

出现错误时，此函数返回非有效句柄 (0)。

AmGetTableCount()

此函数将返回与当前连接有关的数据库中表的数量。

API 语法

long AmGetTableCount(long hApiCnxBase);

内部 Basic 语法

Function AmGetTableCount() As Long

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**

- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetTableDescription()

此函数返回字符串"字符串"格式的，由句柄标识的表的长描述字符串。

API 语法

long AmGetTableDescription(long hApiTable, char *pstrDesc, long lDesc);

内部 Basic 语法

Function AmGetTableDescription(hApiTable As Long) As String

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiTable：此参数包含要获取其长描述字符串的表的有效句柄。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetTableFromName()

此参数返回当前连接中通过其 SQL 名称标识的表的句柄。

API 语法

long AmGetTableFromName(long hApiCnxBase, char *strName);

内部 Basic 语法

Function AmGetTableFromName(strName As String) As Long

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strName：此参数包含要恢复其句柄的表的 SQL 名称。**

输出参数

出现错误时，此函数返回非有效句柄 (0)。

AmGetTableLabel()

此函数返回字符串"字符串"格式的，由句柄标识的表的标签。

API 语法

long AmGetTableLabel(long hApiTable, char *pstrLabel, long lLabel);

内部 Basic 语法

Function AmGetTableLabel(hApiTable As Long) As String

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiTable：此参数包含要获取其标签的表的有效句柄。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetTableName()

返回字符串形式的表的 SQL **名称。**

API 语法

long AmGetTableName(long hApiTable, char *pstrBuffer, long lBuffer);

内部 Basic 语法

Function AmGetTableName(hApiTable As Long) As String

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiTable：此参数包含要恢复其名称的表的有效句柄。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetTableRights()

此函数返回"字符串"格式的，由句柄指定的用户对表具有的权限。返回的字符串由最多两个字符组成，表示创建和删除权限的状态：

- "c" **表示用户有权创建表。**
- "d" **表示用户有权删除表。**

因此，例如：

- "c" **表示用户仅有权创建表。**
- "cd" **表示用户既有权创建表，也有权删除表。**

API 语法

long AmGetTableRights(long hApiTable, char *pstrBuffer, long lBuffer);

内部 Basic 语法

Function AmGetTableRights(hApiTable As Long) As String

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiTable：此参数包含要获取其用户权限的表的有效句柄。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetTableSqlName()

此函数返回字符串"字符串"格式的，由句柄标识的表的 SQL **名称。**

API 语法

long AmGetTableSqlName(long hApiTable, char *pstrBuffer, long lBuffer);

内部 Basic 语法

Function AmGetTableSqlName(hApiTable As Long) As String

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiTable：此参数包含要获取其 SQL 名称的表的有效句柄。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetTargetTable()

返回链接的目标表的 SQL 名称。

API 语法

long AmGetTargetTable(long hApiField);

内部 Basic 语法

Function AmGetTargetTable(hApiField As Long) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiField：此参数包含与此操作有关的链接的句柄。**

输出参数

出现错误时，此函数返回非有效句柄 (0)。

AmGetTrace()

AmGetTrace API **可用于在电缆链接表中的两个节点** (lUserId, lHostId) **之间实现跟踪。跟踪方向** (lTraceDir) **标识是应该按照用户到主机** (lTraceDir = 1) **还是按照主机到用户** (lTraceDir = 0) **的方向进行跟踪。跟踪类型** (lTraceType) **表示跟踪是处于连接** (lTraceType = 1) **还是断开连接** (lTraceType = 2) **状态。完整跟踪指示器** (bFullTrace) **标识跟踪是仅包含修改的节点** (bFullTrace = 0) **还是完整的跟踪** (bFullTrace = 1)。

API 语法

long AmGetTrace(long hApiCnxBase, long lUserId, long lHostId, long iTraceDir, long iTraceType, long bFullTrace, char *pstrTrace, long lTrace);

内部 Basic 语法

Function AmGetTrace(lUserId As Long, lHostId As Long, iTraceDir As Long, iTraceType As Long, bFullTrace As Long) As String

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **lUserId：此参数定义起始连接的链接 ID。**
- **lHostId：此参数定义终止连接的链接 ID。**
- **iTraceDir：此参数指定连接的方向。**
  - 0=**主机到用户**
  - 1=**用户到主机**
- **iTraceType：此参数定义连接的类型。**
  - 1=**连接**
  - 2=**断开连接**
- **bFullTrace：此参数指定忽略部分跟踪并返回完整的跟踪字符串。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetTraceFromHist()

AmGetTraceFromHist API **可根据使用了"跟踪"操作的"跟踪历史记录"计算字符串以显示新连接（与现有连接相对比）。**

API 语法

long AmGetTraceFromHist(long hApiCnxBase, long lProjTraceOutId, long iTraceDir, char *strDelimiter, char *pstrTraceint, long lTraceint, long bUpdateFlag);

内部 Basic 语法

Function AmGetTraceFromHist(lProjTraceOutId As Long, iTraceDir As Long, strDelimiter As String, bUpdateFlag As Long) As String

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lProjTraceOutId：此参数定义项目跟踪** ID。
- **iTraceDir：此参数指定连接的方向。**
  - 0=**主机到用户**
  - 1=**用户到主机**
- **strDelimiter：此参数是用于显示现有连接和断开的连接的字符串分隔符。**
- **bUpdateFlag：此参数对于** AmGetTraceHist API **是可选的，用于更新** amCabTraceOut.TraceString。
  - 0=false
  - 1=true

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetTypedLinkField()

返回其值为 hApiField 参数中指定的键入链接的目标表的 SQL **名称的字段的句柄。**

API 语法

long AmGetTypedLinkField(long hApiField);

内部 Basic 语法

Function AmGetTypedLinkField(hApiField As Long) As Long

应用的字段

版本: 3.02

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiField：此参数包含根据操作生成的键入链接的有效句柄。**

AmGetUserEnvSessionItem()

API 语法

long AmGetUserEnvSessionItem(long hApiCnxBase, char *return, long lreturn, char *strSection, char *strEntry);

内部 Basic 语法

Function AmGetUserEnvSessionItem(strSection As String, strEntry As String) As String

应用的字段

版本: 4.4.0

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmGetVersion()

此函数返回字符串格式的 AssetCenter **的内部版本号。**

API 语法

long AmGetVersion(char *pstrBuf, long lBuf);

内部 Basic 语法

Function AmGetVersion() As String

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmHasAdminPrivilege()

如果已连接的用户具有管理员权限，则函数返回 "TRUE"**（非** 0 **值）。**

API 语法

long AmHasAdminPrivilege(long hApiCnxBase);

内部 Basic 语法

Function AmHasAdminPrivilege() As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

AmHasRelTable()

此函数可用于检测链接是否具有关系表。

API 语法

long AmHasRelTable(long hApiField);

内部 Basic 语法

Function AmHasRelTable(hApiField As Long) As Long

应用的字段

版本: 3.5

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiField：此参数包含与此操作有关的链接上的有效句柄。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmHasRightsForCreation()

此函数可用于确定连接用户是否有权创建指定的表。

内部 Basic 语法

Function AmHasRightsForCreation(strTable As String) As Long

应用的字段

版本: 4.3.0

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strTable：此参数包含与此操作有关的表的** SQL **名称。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
RetVal=amHasRightsForCreation("amEmplDept")
```

AmHasRightsForDeletion()

此函数可用于确定连接用户是否有权删除指定的表。

内部 Basic 语法

Function AmHasRightsForDeletion(strTable As String) As Long

应用的字段

版本: 4.3.0

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **strTable：此参数包含与此操作有关的表的** SQL **名称。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
RetVal=amHasRightsForDeletion("amEmplDept")
```

AmHasRightsForFieldUpdate()

此函数可用于确定连接用户是否有权更新指定的字段。

内部 Basic 语法

Function AmHasRightsForFieldUpdate(strTable As String, strField As String) As Long

应用的字段

版本: 4.3.0

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **strTable：此参数包含与此操作有关的表的** SQL **名称。**
- **strField：此参数包含与此操作有关的字段（在 strTable 参数中指定的表）的** SQL **名称。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
RetVal=amHasRightsForFieldUpdate("amEmplDept", "Location")
```

AmHelpdeskCanCloseFile()

此函数可用于确定连接用户是否能够关闭帮助台工作单。

内部 Basic 语法

Function AmHelpdeskCanCloseFile() As Long

应用的字段

版本: 4.3.0

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

备注

注:

如果连接用户能够关闭帮助台工作单，则函数返回值 "1"。

AmHelpdeskCanProceed()

此函数可用于确定连接用户是否能够继续处理帮助台工作单。

内部 Basic 语法

Function AmHelpdeskCanProceed() As Long

应用的字段

版本: 4.3.0

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

备注

注:

如果连接用户能够继续处理帮助台工作单，则函数返回值 "1"。

AmHelpdeskCanSaveCall()

此函数可用于确定连接用户是否能够保存帮助台工作单。

内部 Basic 语法

Function AmHelpdeskCanSaveCall() As Long

应用的字段

版本: 4.3.0

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

备注

注:

如果连接用户能够保存帮助台工作单，则函数返回值 "1"。

AmImportDocument()

此函数创建并导入文件中的文档。

API 语法

long AmImportDocument(long hApiCnxBase, long lDocObjId, char *strTableName, char *strFileName, char *strCategory, char *strDesignation);

内部 Basic 语法

Function AmImportDocument(lDocObjId As Long, strTableName As String, strFileName As String, strCategory As String, strDesignation As String) As Long

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lDocObjId：此参数包含要存储在** amDocument **表的** lDocObjId **字段中的值。**
- **strTableName：此参数包含要存储在** amDocument **表的** DocObjTable **字段中的值。实际使用中，它是文档要链接的记录所在表的** SQL **名称。**
- **strFileName：此参数包含要导入的文件的名称。**
- **strCategory：此参数包含要在** AssetCenter **中显示的文档的类别。**
- **strDesignation：此参数包含要在** AssetCenter **中显示的文档的名称。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmImportReport()

此函数可用于从文件中导入 Crystal Report **报表。将其导入 amReport 表中现有的记录。**

内部 Basic 语法

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

应用的字段

版本: 4.3.0

	可用
AssetCenter API	
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lReportId：此参数包含要存储导入报表的 amReport 表中记录的标识符。**
- **strFileName：此参数包含要导入的报表所在的文件的完整路径。**

输出参数

- 0**：正常执行。**
- **非** 0 **值：错误代码。**

AmIncrementLogLevel()

此函数在历史记录窗口中显示 strMsg 消息并在向导的完成页面中创建节点。下列所有消息都将在此节点中显示。

内部 Basic 语法

Function AmIncrementLogLevel(strMsg As String, iType As Long) As Long

应用的字段

版本: 3.5

	可用
AssetCenter API	
字段或链接的配置脚本	
"Script" 类型操作	
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **strMsg：此参数包含要显示的消息的文本。**
- **iType：此参数定义与消息相关的图标。可能的值为：**"1" **表示错误，**"2" **表示警告，**"4" **表示信息。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmInsertRecord()

此函数将在数据库中插入先前创建的记录。仅可将 AmCreateRecord 函数创建的记录插入数据库。无法插入使用查询进行访问的记录。

API 语法

long AmInsertRecord(long hApiRecord);

内部 Basic 语法

Function AmInsertRecord(hApiRecord As Long) As Long

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiRecord：此参数包含要插入到数据库的记录的句柄。**

输出参数

- 0：**正常执行。**
- **非** 0 **值：错误代码。**

AmInstantiateReqLine()

可使用此函数直接实例化指定的申请行。

API 语法

long AmInstantiateReqLine(long hApiCnxBase, long lRequestLineId, long bFinal, long lPOrderLineId, double dQty);

内部 Basic 语法

Function AmInstantiateReqLine(lRequestLineId As Long, bFinal As Long, lPOrderLineId As Long, dQty As Double) As Long

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lRequestLineId：此参数包含申请行的标识符。**
- **bFinal：此参数可用于指定是否结束分配操作。**
- **lPOrderLineId：此参数包含订单行的标识符。**
- **dQty：此参数包含要实例化的数量。**

输出参数

- 0：**正常执行。**
- **非** 0 **值：错误代码。**

备注

无需逐步完成采购周期，使用此函数可创建申请元素。如果 bFinal=FALSE**，则创建状态为"等待接收中"的资产。**

AmInstantiateRequest()

可使用此函数直接实例化指定申请的全部内容。

API 语法

long AmInstantiateRequest(long hApiCnxBase, long lRequestId, long lMulFactor);

内部 Basic 语法

Function AmInstantiateRequest(lRequestId As Long, lMulFactor As Long) As Long

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lRequestId：此参数包含申请的标识符。**
- **lMulFactor：此参数可用于指定要执行实例化操作的申请数量。**

输出参数

- **0：正常执行。**
- **非 0 值：错误代码。**

AmIsConnected()

此函数测试当前连接是否有效。

API 语法

long AmIsConnected(long hApiCnxBase);

应用的字段

版本: 3.00

	可用
AssetCenter API	✓

可用
字段或链接的配置脚本
"Script" 类型操作
向导脚本
向导的 FINISH.DO 脚本

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmIsFieldForeignKey()

此函数可用于确定字段是否为数据库中的外键。

API 语法

long AmIsFieldForeignKey(long hApiField);

内部 Basic 语法

Function AmIsFieldForeignKey(hApiField As Long) As Long

应用的字段

版本: 2.52

可用
AssetCenter API
字段或链接的配置脚本
"Script" 类型操作
向导脚本
向导的 FINISH.DO 脚本

输入参数

- **hApiField：此参数包含要标识的字段的句柄。**

输出参数

- 1：字段是外键。
- 0：字段不是外键。

AmIsFieldIndexed()

此函数可用于确定字段是否已建立索引。

API 语法

long AmIsFieldIndexed(long hApiField);

内部 Basic 语法

Function AmIsFieldIndexed(hApiField As Long) As Long

应用的字段

版本: 3.5

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiField：此参数包含要标识的字段的句柄。**

输出参数

- 1：字段已建立索引。
- 0：字段未建立索引。

AmIsFieldPrimaryKey()

此函数可用于确定数据库中的字段是否为主键。

API 语法

long AmIsFieldPrimaryKey(long hApiField);

内部 Basic 语法

Function AmIsFieldPrimaryKey(hApiField As Long) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiField：此参数包含要标识的字段的句柄。**

输出参数

- 1：**字段是主键。**
- 0：**字段不是主键。**

AmIsHelpdeskAdmin()

此函数可用于确定连接用户是否为帮助台管理员。

内部 Basic 语法

Function AmIsHelpdeskAdmin() As Long

应用的字段

版本: 4.3.0

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

备注

注:

如果连接用户是帮助台管理员，则函数返回值 "1"。

AmIsHelpdeskMember()

此函数可用于确定连接用户是否属于帮助台组。

内部 Basic 语法

Function AmIsHelpdeskMember() As Long

应用的字段

版本: 4.3.0

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

备注

注:

如果连接用户属于帮助台组，则函数返回值 "1"。

AmIsHelpdeskSuper()

此函数可用于确定连接用户是否是帮助台组主管。

内部 Basic 语法

Function AmIsHelpdeskSuper() As Long

应用的字段

版本: 4.3.0

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

备注

注:

如果连接用户是帮助台主管，则函数返回值 "1"。

AmIsLink()

确定通过其句柄标识的对象是链接还是字段。

API 语法

long AmIsLink(long hApiField);

内部 Basic 语法

Function AmIsLink(hApiField As Long) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiField：此参数包含与此操作有关的对象的句柄。**

输出参数

- 1：**此对象是链接。**
- 0：**此对象是字段。**

AmIsModuleAuthorized()

此函数可用于确定连接用户是否能够访问应用程序的指定模块。

内部 Basic 语法

Function AmIsModuleAuthorized(strModuleName As String) As Long

应用的字段

版本: 4.3.0

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strModuleName：此参数包含与此操作有关的模块的名称。下面是可能的模块列表：**
  - **资产组合：资产组合管理模块**
  - **调整：调整模块**
  - **合同：合同管理模块**
  - **租赁：租赁管理模块**

- **采购：采购管理模块**
- **财务：财务模块**
- **帮助台：帮助台管理模块**
- **电缆和电路：电缆和电路模块**
- **条码：条码模块**
- **管理：管理模块**
- Visio：Visiro **集成模块**
- API：API **库**
- **向导：向导管理模块**
- **工作流：工作流管理模块**
- AutoCAD：AutoCAD **集成模块**
- Knowlix：Knowlix **集成模块**
- DA_Automation：**自动模块**
- DA_RemoteControl：**远程控制模块**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

备注

注:

并非所有的模块都可用或可在应用程序中启用。模块的可用性取决于您从 Peregrine Systems, Inc. **获得的许可证。**

AmIsTypedLink()

确定通过其句柄标识的对象是否为键入链接。

API 语法

long AmIsTypedLink(long hApiField);

内部 Basic 语法

Function AmIsTypedLink(hApiField As Long) As Long

应用的字段

版本: 3.02

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiField：此参数包含与此操作有关的对象的句柄。**

输出参数

- 1：**此对象是键入链接。**
- 0：**此对象不是键入链接。**

AmLastError()

此函数返回在相应的连接上下文中最后执行的函数产生的上一错误代码。

API 语法

long AmLastError(long hApiCnxBase);

内部 Basic 语法

Function AmLastError() As Long

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmLastErrorMsg()

此函数返回当前连接中发生的上一错误消息。

API 语法

long AmLastErrorMsg(long hApiCnxBase, char *pstrBuffer, long lBuffer);

内部 Basic 语法

Function AmLastErrorMsg() As String

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

AmListToString()

此函数将通过 AmDbGetList 函数获得的字符串形式的结果转换为可与 AmDbGetString 函数的结果按照相同方式显示的字符串。

API 语法

long AmListToString(char *return, long lreturn, char *strSource, char *strColSep, char *strLineSep, char *strIdSep);

内部 Basic 语法

Function AmListToString(strSource As String, strColSep As String, strLineSep As String, strIdSep As String) As String

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strSource：此参数包含要转换的字符串。**
- **strColSep：此参数包含的字符可在要转换的字符串中作为列分隔符。**
- **strLineSep：此参数包含的字符可在要转换的字符串中作为行分隔符。**
- **strIdSep：此参数包含的字符可在要转换的字符串中作为标识符分隔符。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmLog()

此函数在历史记录窗口中显示 strMessage 消息。

内部 Basic 语法

Function AmLog(strMessage As String, iLogType As Long) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	
"Script" 类型操作	
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **strMessage：此参数包含要显示的消息的文本。**
- **iLogType：此参数定义与消息相关的图标。可能的值为：**"1" **表示错误，**"2" **表示警告，**"4" **表示信息。**

输出参数

- 0**：正常执行。**
- **非** 0 **值：错误代码。**

示例

```
AmLog("这是消息")
```

AmLoginId()

此函数返回连接用户的标识符。

API 语法

long AmLoginId(long hApiCnxBase);

内部 Basic 语法

Function AmLoginId() As Long

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

下例将连接用户的标识符定义为数据库字段的默认值：

```
RetVal=AmLoginId()
```

AmLoginName()

此函数返回连接用户的登录名。

API 语法

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

内部 Basic 语法

Function AmLoginName() As String

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

下例将连接用户的登录名定义为数据库字段的默认值：

```
RetVal=AmLoginName()
```

AmMapSubReqLineAgent()

此函数可用于在申请行和订单行的子行之间建立可能的链接。

API 语法

long AmMapSubReqLineAgent(long hApiCnxBase, long lRequestLineId, long lPorderLineId);

内部 Basic 语法

Function AmMapSubReqLineAgent(lRequestLineId As Long, lPorderLineId As Long) As Long

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lRequestLineId：此参数包含申请行的标识符。**
- **lPorderLineId：此参数包含申请行的标识符。**

输出参数

- **0：正常执行。**
- **非 0 值：错误代码。**

AmMoveCable()

使用 AmMoveCable API **将电缆** (lCableId) **从其当前位置移动到指定的目标位置** (lToLoc)。**如果项目** (lProjectId) **和工作单** (lWorkOrderId) **已赋值，根据指定的备注** (strComment) **将电缆添加到项目和工作单。此备注描述了要在电缆上执行的操作（如"将电缆移动到其他位置"）。**

API 语法

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

内部 Basic 语法

Function AmMoveCable(lCableId As Long, lToLocId As Long, lProjectId As Long, lWorkOrderId As Long, strComment As String) As Long

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lCableId：此参数包含要移动的电缆的** ID。
- **lToLocId：此参数定义要移动的电缆。**
- **lProjectId：此参数是项目** ID。
- **lWorkOrderId：此参数定义工作单** ID。
- **strComment：此参数是在工作单上使用的备注。**

输出参数

- 0**：正常执行。**
- **非** 0 **值：错误代码。**

AmMoveDevice()

使用 AmMoveDevice API **将设备** (lAssetId) **从其当前位置移动到指定的目标位置** (lToLoc)。**如果项目** (lProjectId) **和工作单** (lWorkOrderId) **已赋值，根据指定的备注** (strComment) **将设备添加到项目和工作单。此备注描述了要在电缆上执行的操作（如"将设备移动到其他位置"）。**

API 语法

long AmMoveDevice(long hApiCnxBase, long lDeviceId, long lToLocationId, long lProjectId, long lWorkOrderId, char *strComment);

内部 Basic 语法

Function AmMoveDevice(lDeviceId As Long, lToLocationId As Long, lProjectId As Long, lWorkOrderId As Long, strComment As String) As Long

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lDeviceId：此参数定义要移动的设备的** ID。
- **lToLocationId：此参数定义设备的新位置。**
- **lProjectId：此参数是项目** ID。
- **lWorkOrderId：此参数定义工作单** ID。
- **strComment：此参数是在工作单上使用的备注。**

输出参数

- 0**：正常执行。**
- **非** 0 **值：错误代码。**

AmMsgBox()

此函数显示包含消息的对话框。

内部 Basic 语法

Function AmMsgBox(strMessage As String, lMode As Long) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strMessage：此参数包含对话框中显示的消息。**
- **lMode：此参数包含显示的对话框的类型（0 表示具有"确定"按钮的简单对话框，1 表示具有"确定"和"取消"按钮的对话框，2 表示仅具有"取消"按钮的对话框）。**

输出参数

- **0：正常执行。**
- **非 0 值：错误代码。**

示例

```
AmMsgBox("执行了移动")
```

AmOpenConnection()

根据 AC 数据库名创建会话。strDataSource 应该是有效的 AssetCenter 数据源名（在 AssetCenter 的登录框中列出了这些 AC 数据库连接）。
可以在同一个数据库或不同的数据库中打开多个连接。

API 语法

long AmOpenConnection(char *strDataSource, char *strUser, char *strPwd);

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	
向导脚本	
向导的 FINISH.DO 脚本	

输入参数

- **strDataSource：此参数包含数据源名称。**
- **strUser：此参数包含连接的用户名。**
- **strPwd：此参数包含特定用户的密码。**

AmOpenScreen()

此函数可用于在 AssetCenter 中打开屏幕或视图。

内部 Basic 语法

Function AmOpenScreen(strScreenId As String, strContext As String, strFilter As String, iMode As Long, strBindField As String, bStayReadOnly As Long) As Long

应用的字段

版本: 4.00

	可用
AssetCenter API	
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strScreenId：此参数包含系统视图或要打开的用户屏幕（按优先级顺序）的**SQL**名称。**
- **strContext：此参数是可选的，包含记录的标识符列表，打开屏幕时在列表中选择这些记录。**
- **strFilter：此参数包含打开屏幕时在列表上应用的** AQL **筛选器。**
- **iMode：此参数包含打开屏幕时的模式：查看、编辑等。可能的值有：**0**（未进行操作），**1**（未进行操作），**2**（正在进行修改操作），**3**（正在进行创建操作），**4**（正在进行复制操作），**5**（正在进行添加操作），**6**（正在进行选择操作）。**
- **strBindField：使用此参数可通过筛选器打开模式为已链接窗口的屏幕。它使用源字段的** SQL **名称或** CurrentSrcChoice **值以使用当前上下文。**
- **bStayReadOnly：此参数可用于以只读模式打开屏幕。无论用户权限如何都不允许对其进行修改。**

输出参数

- 0**：正常执行。**
- **非** 0 **值：错误代码。**

AmOverflowTables()

此函数将返回指定表的溢出表的 SQL **名称。**

API 语法

long AmOverflowTables(long hApiCnxBase, char *strBasisTable, char *strOverflowTables, long lOverflowTables);

内部 Basic 语法

Function AmOverflowTables(strBasisTable As String) As String

应用的字段

版本: 4.3.0

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strBasisTable：此参数包含与此操作有关的表的 SQL 名称。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

备注

注:

在函数返回的列表中使用逗号作为分隔符。如果指定表不存在溢出表，则函数返回空字符串。

示例

下例将返回资产组合项表 (amPortfolio) 的溢出表：

```
RetVal = AmOverflowTables("amPortfolio")
```

此例子的结果是：

```
amComputer,amSoftInstall,amPhone
```

AmPagePath()

此函数返回包含向导执行路径（如"已浏览的页面的列表"）的字符串。忽略向后回跳。

内部 Basic 语法

Function AmPagePath() As String

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	
"Script" 类型操作	
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

AmProgress()

此函数在向导的完成页面中显示的进度指示器表示完成的百分比。

内部 Basic 语法

Function AmProgress(iProgress As Long) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	
"Script" 类型操作	
向导脚本	
向导的 FINISH.DO 脚本	

输入参数

- **iProgress：此参数包含用于定义进度指示器大小的完成百分比（在 0 到 100 之间）。**

输出参数

- 0：**正常执行。**
- **非 0 值：错误代码。**

示例

```
AmProgress(85)
```

此函数显示的进度指示器表示已完成 85%。

AmPurgeRecord()

此函数可销毁记录。

API 语法

long AmPurgeRecord(long hApiRecord);

内部 Basic 语法

Function AmPurgeRecord(hApiRecord As Long) As Long

应用的字段

版本: 4.3.0

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiRecord：此参数包含与此操作有关的记录的句柄。**

输出参数

- 0：**正常执行。**
- **非** 0 **值：错误代码。**

备注

注:

对链接记录的处理取决于链接的类型。对于 OWN **类型链接，链接记录的处理方式相同。对于** DEFINE **或** NORMAL **类型的链接，链接记录的外键将重置为** 0**，并将归档记录的标识符和描述字段填充到归档字段。**

重要:

此函数可用于归档表或标准表中的记录。

AmQueryCreate()

此函数在当前连接中创建查询对象。随后，可使用此对象将AQL**语句发送到数据库服务器。**

API 语法

long AmQueryCreate(long hApiCnxBase);

内部 Basic 语法

Function AmQueryCreate() As Long

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

AmQueryExec()

此函数执行 AQL 查询。它返回查询的第一个结果。可通过 AmQueryNext 函数获取下一结果。

当使用此函数发送的查询返回"备注"类型字段时，其结果将限制在255个字符以内。

API 语法

long AmQueryExec(long hApiQuery, char *strQueryCommand);

内部 Basic 语法

Function AmQueryExec(hApiQuery As Long, strQueryCommand As String) As Long

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiQuery：此参数包含发送 AQL 语句的查询对象的有效句柄。**
- **strQueryCommand：此参数包含字符串形式的 AQL 查询的正文。**

输出参数

- **0：正常执行。**
- **非 0 值：错误代码。**

AmQueryGet()

此函数执行不带游标的 AQL 查询（单一结果）。仅返回一行结果。

API 语法

long AmQueryGet(long hApiQuery, char *strQueryCommand);

内部 Basic 语法

Function AmQueryGet(hApiQuery As Long, strQueryCommand As String) As Long

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiQuery：此参数包含发送 AQL 语句的查询对象的有效句柄。**

- **strQueryCommand：此参数包含字符串形式的 AQL 查询的正文。**

输出参数

- 0：正常执行。
- 非 0 值：错误代码。

AmQueryNext()

此函数返回预先使用 AmQueryExec 函数执行的查询的结果。

API 语法

long AmQueryNext(long hApiQuery);

内部 Basic 语法

Function AmQueryNext(hApiQuery As Long) As Long

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiQuery：此参数包含发送 AQL 语句的查询对象的有效句柄。**

输出参数

- 0：正常执行。
- 非 0 值：错误代码。

AmQuerySetAddMainField()

此函数可用于发送查询（在表的主字段可自动添加到要返回的字段列表的模式中）。这种类型的查询从不返回空标识符记录。

API 语法

long AmQuerySetAddMainField(long hApiQuery, long bAddMainField);

内部 Basic 语法

Function AmQuerySetAddMainField(hApiQuery As Long, bAddMainField As Long) As Long

应用的字段

版本: 3.5

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiQuery：此参数包含查询对象上的有效句柄。**
- **bAddMainField：此参数可以有以下两个值之一：**
  - True：**已添加表的主字段**
  - False：**未添加表的主字段**

输出参数

- 0：**正常执行。**
- **非** 0 **值：错误代码。**

AmQuerySetFullMemo()

默认情况下，执行 AmQueryExec 函数时，查询将“备注”类型字段截断为 254 个字符。此函数在“备注”字段完全恢复的模式下发送查询。

API 语法

long AmQuerySetFullMemo(long hApiQuery, long bFullMemo);

内部 Basic 语法

Function AmQuerySetFullMemo(hApiQuery As Long, bFullMemo As Long) As Long

应用的字段

版本: 3.5

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiQuery：此参数包含查询对象上的有效句柄。**
- **bFullMemo：此参数可以有以下两个值之一：**
  - True：**查询返回完整的“备注”字段**
  - False：**查询将“备注”字段截断为 254 个字符。**

输出参数

- 0：**正常执行。**
- **非 0 值：错误代码。**

AmQueryStartTable()

此函数返回与通过其句柄标识的查询有关的表的句柄。

API 语法

long AmQueryStartTable(long hApiQuery);

内部 Basic 语法

Function AmQueryStartTable(hApiQuery As Long) As Long

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiQuery：此参数包含查询对象的有效句柄。**

输出参数

出现错误时，此函数返回非有效句柄 (0)。

AmQueryStop()

此函数中断由其句柄标识的查询的执行过程。在使用 AmQueryExec 函数前必须预先启动此查询。

API 语法

long AmQueryStop(long hApiQuery);

内部 Basic 语法

Function AmQueryStop(hApiQuery As Long) As Long

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiQuery：此参数包含查询对象的有效句柄。**

输出参数

- 0：**正常执行。**
- **非** 0 **值：错误代码。**

AmReceiveAllPOLines()

此函数接收订单行上的所有的项（完整交付）。

注:

警告：提交事务时由代理创建交付行。不能提前访问这些交付行。

API 语法

long AmReceiveAllPOLines(long hApiCnxBase, long lPOrdId, long lDelivId);

内部 Basic 语法

Function AmReceiveAllPOLines(lPOrdId As Long, lDelivId As Long) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lPOrdId：此参数包含要接收项所在的订单行的标识符。**
- **lDelivId：此参数包含用于接收订单行上显示的所有项的接收单的标识符。**

输出参数

- 0：**正常执行。**
- **非** 0 **值：错误代码。**

AmReceivePOLine()

此函数将交付订单行上指定数量的项（部分交付）并返回交付行的标识符。

注:

警告：一旦提交事务即由代理创建交付行。在执行此操作之前无法访问这些交付行。

API 语法

long AmReceivePOLine(long hApiCnxBase, long lPOrdLineId, long lDelivId, double dQty);

内部 Basic 语法

Function AmReceivePOLine(lPOrdLineId As Long, lDelivId As Long, dQty As Double) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lPOrdLineId：此参数包含要接收项所在的采购订单行的标识符。**
- **lDelivId：此参数包含用于接收订单行上指定数量项的接收单的标识符。**
- **dQty：此参数包含接收单中要接收订单行上的项数。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmRefreshAllCaches()

此函数将刷新在 AssetCenter **中使用的缓存。**

API 语法

long AmRefreshAllCaches(long hApiCnxBase);

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	
向导脚本	
向导的 FINISH.DO 脚本	

输出参数

- 0：**正常执行。**
- **非** 0 **值：错误代码。**

AmRefreshLabel()

AmRefreshLabel API **刷新指定表** (strTableName) **中的指定记录** (lMainId) **的标签字符串。**

API 语法

long AmRefreshLabel(long hApiCnxBase, long lMainId, char *strTableName, char *pstrLabel, long lLabel);

内部 Basic 语法

Function AmRefreshLabel(lMainId As Long, strTableName As String) As String

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lMainId：此参数定义要刷新的记录** ID。

- **strTableName：此参数定义** lMainId **所在的表名。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmRefreshProperty()

重新对由 strVarName 参数标识的属性求值。如果此属性使用了脚本，将重新执行此脚本。

或者，将更新依赖关系数。

内部 Basic 语法

Function AmRefreshProperty(strVarName As String) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	
"Script" 类型操作	
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strVarName：此参数包含要重新求值的向导的属性名。**

输出参数

- 0**：正常执行。**
- **非** 0 **值：错误代码。**

AmRefreshTraceHist()

AmRefreshTraceHist API **刷新全部项目跟踪条目，还可通过可选参数刷新"单个"跟踪历史记录条目。如果未提供此参数，将刷新全部跟踪历史记录。**

API 语法

long AmRefreshTraceHist(long hApiCnxBase, long lCabTraceOutId, long lTraceHistId);

内部 Basic 语法

Function AmRefreshTraceHist(lCabTraceOutId As Long, lTraceHistId As Long) As Long

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lCabTraceOutId：此参数是电缆跟踪输出** ID。
- **lTraceHistId：此参数是可选的，允许刷新"单个"跟踪历史记录。**

输出参数

- 0**：正常执行。**
- **非** 0 **值：错误代码。**

AmReleaseHandle()

此函数释放对象的句柄和子句柄。

API 语法

long AmReleaseHandle(long hApiObject);

内部 Basic 语法

Function AmReleaseHandle(hApiObject As Long) As Long

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiObject：此参数包含与对象有关的句柄。**

输出参数

- 0：**正常执行。**
- **非** 0 **值：错误代码。**

AmRemoveCable()

AmRemoveCable API **将电缆** (lCableId) **从其当前位置移除。将电缆的状态更新为"不可用"。如果项目** (lProjectId) **和工作单** (lWorkOrderId) **已赋值，根据指定的备注** (strComment) **将电缆添加到项目和工作单。此备注描述了将要对电缆执行的操作（如"从当前位置移除电缆"）。**

API 语法

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

内部 Basic 语法

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

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lCableId：此参数包含要移除的电缆的** ID。
- **lProjectId：此参数是项目** ID。
- **lWorkOrderId：此参数定义工作单** ID。
- **strComment：此参数是在工作单上使用的备注。**

输出参数

- 0：**正常执行。**
- **非** 0 **值：错误代码。**

AmRemoveDevice()

AmRemoveDevice API **将设备** (lAssetId) **从其当前位置移除。将设备的状态更新为"不可用"。如果项目** (lProjectId) **和工作单** (lWorkOrderId) **已赋值，根据指定的备注** (strComment) **将设备添加到项目和工作单。此备注描述了将要对设备执行的操作（如"从当前位置移除设备"）。**

API 语法

long AmRemoveDevice(long hApiCnxBase, long lDeviceId, long lProjectId, long lWorkOrderId, char *strComment);

内部 Basic 语法

Function AmRemoveDevice(lDeviceId As Long, lProjectId As Long, lWorkOrderId As Long, strComment As String) As Long

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lDeviceId：此参数定义要移除的设备** ID。
- **lProjectId：此参数是项目** ID。
- **lWorkOrderId：此参数定义工作单** ID。
- **strComment：此参数是在工作单上使用的备注。**

输出参数

- 0**：正常执行。**
- **非** 0 **值：错误代码。**

AmResetPassword()

API 语法

long AmResetPassword(long hApiCnxBase, char *strOldPassword, char *strNewPassword);

内部 Basic 语法

Function AmResetPassword(strOldPassword As String, strNewPassword As String) As Long

应用的字段

版本: 4.4.0

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输出参数

- 0：**正常执行。**
- **非** 0 **值：错误代码。**

AmResetUserEnvSession()

API 语法

long AmResetUserEnvSession(long hApiCnxBase, char *strSection);

内部 Basic 语法

Function AmResetUserEnvSession(strSection As String) As Long

应用的字段

版本: 4.4.0

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓

	可用
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输出参数

- 0：正常执行。
- 非 0 值：错误代码。

AmResetUserPassword()

API 语法

long AmResetUserPassword(long hApiCnxBase, char *strUser, char *strPasswd, char *strNewPasswd);

内部 Basic 语法

Function AmResetUserPassword(strUser As String, strPasswd As String, strNewPasswd As String) As Long

应用的字段

版本: 4.4.0

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmRestoreRecord()

此函数可恢复归档记录。

API 语法

long AmRestoreRecord(long hApiRecord);

内部 Basic 语法

Function AmRestoreRecord(hApiRecord As Long) As Long

应用的字段

版本: 4.3.0

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiRecord：此参数包含与此操作有关的记录的句柄。**

输出参数

- 0**：正常执行。**
- **非** 0 **值：错误代码。**

备注

注:

对链接记录的处理取决于链接的类型。对于 OWN **类型链接，链接记录的处理方式相同。对于** DEFINE **或** NORMAL **类型的链接，链接记录的外键将重置为** 0**，并将归档记录的标识符和描述字段填充到归档字段。**

重要:

此函数可用于归档表或标准表中的记录。

AmReturnAsset()

此函数可用于退回资产。

API 语法

long AmReturnAsset(long hApiCnxBase, long lAstId, long lReturnId, long bCanMerge);

内部 Basic 语法

Function AmReturnAsset(lAstId As Long, lReturnId As Long, bCanMerge As Long) As Long

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lAstId：此参数包含要退回的资产的标识符。**
- **lReturnId：此参数包含退货单的标识符。**
- **bCanMerge：此参数可用于指定是否可将退回的项与退货单上现有行合并。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmReturnContract()

此函数可用于退回合同。

API 语法

long AmReturnContract(long hApiCnxBase, long lCntrId, long lReturnId, long bCanMerge);

内部 Basic 语法

Function AmReturnContract(lCntrId As Long, lReturnId As Long, bCanMerge As Long) As Long

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lCntrId：此参数包含要退回的合同的标识符。**
- **lReturnId：此参数包含退货单的标识符。**
- **bCanMerge：此参数可用于指定是否可将退回的项与退货单上现有行合并。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**

- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmReturnPortfolioItem()

此函数可用于退回资产组合项。

API 语法

long AmReturnPortfolioItem(long hApiCnxBase, long lPfId, double dQty, long lFromRecptLineId, long lReturnId, long bCanMerge);

内部 Basic 语法

Function AmReturnPortfolioItem(lPfId As Long, dQty As Double, lFromRecptLineId As Long, lReturnId As Long, bCanMerge As Long) As Long

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lPfId：此参数包含要退回的资产组合项的标识符。**
- **dQty：此参数包含要退回的数量（以模型为单位）。**
- **lFromRecptLineId：此参数包含源退货单的标识符。**
- **lReturnId：此参数包含退货单的标识符。**
- **bCanMerge：此参数可用于指定是否可将退回的项与退货单上现有行合并。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmReturnTraining()

此函数可用于返回培训。

API 语法

long AmReturnTraining(long hApiCnxBase, long lTrainingId, long lReturnId, long bCanMerge);

内部 Basic 语法

Function AmReturnTraining(lTrainingId As Long, lReturnId As Long, bCanMerge As Long) As Long

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lTrainingId：此参数包含要返回的培训的标识符。**
- **lReturnId：此参数包含退货单的标识符。**
- **bCanMerge：此参数可用于指定是否可将退回的项与退货单上现有行合并。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**

- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmReturnWorkOrder()

此函数可用于返回工作单。

API 语法

long AmReturnWorkOrder(long hApiCnxBase, long lWOId, long lReturnId, long bCanMerge);

内部 Basic 语法

Function AmReturnWorkOrder(lWOId As Long, lReturnId As Long, bCanMerge As Long) As Long

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **lWOId：此参数包含要返回的工作单的标识符。**
- **lReturnId：此参数包含退货单的标识符。**
- **bCanMerge：此参数可用于指定是否可将退回的项与退货单上现有行合并。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**

- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

AmRevCryptPassword()

此函数可加密可逆密码。对使用此函数加密的密码进行解密的函数不可用。

API 语法

long AmRevCryptPassword(long hApiCnxBase, char *return, long lreturn, char *strPassword);

内部 Basic 语法

Function AmRevCryptPassword(strPassword As String) As String

应用的字段

版本: 3.5

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strPassword：此参数包含要加密的密码。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

AmRgbColor()

此函数指定与 strText 参数相对应的颜色的 RGB 值。

API 语法

long AmRgbColor(char *strText);

内部 Basic 语法

Function AmRgbColor(strText As String) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strText：此参数包含颜色的名称：**
  - White
  - ltGray
  - Gray
  - Dkgray
  - Black
  - Red
  - Green
  - Blue
  - Yellow
  - Cyan
  - Magenta
  - Dkyellow
  - Dkgreen

- Dkcyan
- Dkblue
- Dkmagenta
- Dkred

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

AmRollback()

此函数撤销声明事务（通过 AmStartTransaction 函数执行的事务）开始之前所作的全部修改。

API 语法

long AmRollback(long hApiCnxBase);

内部 Basic 语法

Function AmRollback() As Long

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输出参数

- 0：**正常执行。**

- **非 0 值：错误代码。**

AmSetFieldDateOnlyValue()

此函数修改记录中的字段。此函数不会更新数据库。更新、插入记录，或者提交事务时进行修改。

API 语法

long AmSetFieldDateOnlyValue(long hApiRecord, char *strFieldName, long dtptmValue);

内部 Basic 语法

Function AmSetFieldDateOnlyValue(hApiRecord As Long, strFieldName As String, dtptmValue As Date) As Long

应用的字段

版本: 4.3.0

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiRecord：此参数包含要修改字段所在的记录的句柄。**
- **strFieldName：此参数包含要修改字段的 SQL 名称。**
- **dtptmValue：此参数仅包含"日期"格式的字段的新值。与 AmSetFieldDateValue 函数不同，仅处理"日期"部分而忽略"时间"部分。**

输出参数

- **0：正常执行。**
- **非 0 值：错误代码。**

AmSetFieldDateValue()

此函数修改记录中的字段。此函数不会更新数据库。更新、插入记录，或者提交事务时进行修改。

API 语法

long AmSetFieldDateValue(long hApiRecord, char *strFieldName, long tmValue);

内部 Basic 语法

Function AmSetFieldDateValue(hApiRecord As Long, strFieldName As String, tmValue As Date) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiRecord：此参数包含要修改字段所在的记录的句柄。**
- **strFieldName：此参数包含要修改字段的 SQL 名称。**
- **tmValue：此参数包含"日期"格式的字段的新值。**

输出参数

- **0：正常执行。**
- **非 0 值：错误代码。**

AmSetFieldDoubleValue()

此函数修改记录中的字段。此函数不会更新数据库。

API 语法

long AmSetFieldDoubleValue(long hApiRecord, char *strFieldName, double dValue);

内部 Basic 语法

Function AmSetFieldDoubleValue(hApiRecord As Long, strFieldName As String, dValue As Double) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiRecord：此参数包含要修改字段所在的记录的句柄。**
- **strFieldName：此参数包含要修改字段的 SQL 名称。**
- **dValue：此参数包含"双精度"格式的字段的新值。**

输出参数

- **0：正常执行。**
- **非 0 值：错误代码。**

AmSetFieldLongValue()

此函数修改记录中的字段。此函数不会更新数据库。要修改日期、时间或日期+时间的值，必须将新值的形式表示为自 1970 年 1 月 1 日 00:00:00 以来经过的秒数。

API 语法

long AmSetFieldLongValue(long hApiRecord, char *strFieldName, long lValue);

内部 Basic 语法

Function AmSetFieldLongValue(hApiRecord As Long, strFieldName As String, lValue As Long) As Long

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiRecord：此参数包含要修改字段所在的记录的句柄。**
- **strFieldName：此参数包含要修改字段的 SQL 名称。**
- **lValue：此参数包含字段的新值。**

输出参数

- **0：正常执行。**
- **非 0 值：错误代码。**

AmSetFieldStrValue()

此函数修改记录中的字段。此函数不会更新数据库。

API 语法

long AmSetFieldStrValue(long hApiRecord, char *strFieldName, char *strValue);

内部 Basic 语法

Function AmSetFieldStrValue(hApiRecord As Long, strFieldName As String, strValue As String) As Long

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiRecord：此参数包含要修改字段所在的记录的句柄。**
- **strFieldName：此参数包含要修改字段的 SQL 名称。**
- **strValue：此参数包含"字符串"格式的字段的新值。**

输出参数

- **0：正常执行。**
- **非 0 值：错误代码。**

AmSetLinkFeatureValue()

此函数用于设置指定记录的链接类型特征值。

API 语法

long AmSetLinkFeatureValue(long hApiRecord, char *strFeatSqlName, char *strDstSelfValue, long lDstId);

内部 Basic 语法

Function AmSetLinkFeatureValue(hApiRecord As Long, strFeatSqlName As String, strDstSelfValue As String, lDstId As Long) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiRecord：此参数包含与链接类型特征值关联的记录的标识符。**
- **strFeatSqlName：此参数包含要设置其值的链接类型特征值的SQL名称。此SQL名称总是以 "fv_" 作为开始。**
- **strDstSelfValue：此参数包含要显示的记录的特征值。它是标识符为 lDstId 的记录的 "Self" 值。如果传递无效或不存在的值，将造成损坏数据库完整性的风险。**
- **lDstId：此参数包含链接类型特征值指向的记录的标识符。**

输出参数

- **0：正常执行。**
- **非 0 值：错误代码。**

AmSetProperty()

此函数设置由其名称标识的属性的值。还更新此属性的依赖关系树。

内部 Basic 语法

Function AmSetProperty(strVarName As String, vValue As Variant) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	
"Script" 类型操作	
向导脚本	
向导的 FINISH.DO 脚本	

输入参数

- **strVarName：此参数包含要设置其值的属性的名称。**
- **vValue：此参数包含属性的新值。**

输出参数

- 0：**正常执行。**
- **非** 0 **值：错误代码。**

AmSetUserEnvSessionItem()

API 语法

long AmSetUserEnvSessionItem(long hApiCnxBase, char *strSection, char *strEntry, char *strValue);

内部 Basic 语法

Function AmSetUserEnvSessionItem(strSection As String, strEntry As String, strValue As String) As Long

应用的字段

版本: 4.4.0

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输出参数

- 0：正常执行。
- 非 0 值：错误代码。

AmShowCableCrossConnect()

此函数显示交叉连接屏幕。

内部 Basic 语法

Function AmShowCableCrossConnect(lCableId As Long) As Long

应用的字段

版本: 4.00

	可用
AssetCenter API	
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **lCableId：此参数包含与此操作有关的电缆的标识符。**

输出参数

- 0：正常执行。
- 非 0 值：错误代码。

AmShowDeviceCrossConnect()

此函数显示电缆设备的交叉连接屏幕。

内部 Basic 语法

Function AmShowDeviceCrossConnect(lDeviceId As Long) As Long

应用的字段

版本: 4.00

	可用
AssetCenter API	
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **lDeviceId：此参数包含与此操作有关的电缆的标识符。**

输出参数

- 0：**正常执行。**
- **非** 0 **值：错误代码。**

AmSqlTextConst()

此函数将转换在查询中使用的字符串。对字符串执行下面的操作：

- **将所有单引号 (') 转换成双引号，**
- **将单引号添加到字符串的开头和结尾处。**

API 语法

long AmSqlTextConst(char *return, long lreturn, char *str);

内部 Basic 语法

Function AmSqlTextConst(str As String) As String

应用的字段

版本: 4.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **str：此参数包含要处理的字符串。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim strReq as String
strReq="SELECT lEmplDeptId FROM amEmplDept WHERE Name=" & amSqlTextConst(s
trName)
```

即使 strName **变量中包含单引号，此查询也有效。**

AmStandIn()

此函数返回在 tmDate 日期上替换标识符为 lEmployeeId 的员工的员工的标识符。

API 语法

long AmStandIn(long hApiCnxBase, long lEmployeeId, long tmDate);

内部 Basic 语法

Function AmStandIn(lEmployeeId As Long, tmDate As Date) As Long

应用的字段

版本: 4.3.0

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **lEmployeeId：此参数包含要获取其替换人的员工的标识符。**
- **tmDate：此参数包含函数执行搜索操作的日期。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

备注

注:

如果在指定的日期 tmDate，标识符为 lEmployeeId 的员工在岗，则函数返回其标识符。

如果员工缺勤并且未指定替换人，则函数返回 0。

示例

```
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()

此函数返回在tmDate日期上替换标识符为lEmployeeId的员工的员工组的标识符。

API 语法

long AmStandInGroup(long hApiCnxBase, long lEmployeeId, long tmDate);

内部 Basic 语法

Function AmStandInGroup(lEmployeeId As Long, tmDate As Date) As Long

应用的字段

版本: 4.3.0

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **lEmployeeId：此参数包含要获取其所在的替换组的员工的标识符。**
- **tmDate：此参数包含函数执行搜索操作的日期。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

备注

注:

如果在指定的日期 tmDate，标识符为 IEmployeeId 的员工在岗，则函数返回 0。
如果员工缺勤并且未指定替换组，则函数也返回 0。

AmStartTransaction()

此函数启动与连接相关的数据库有关的新事务。接下来的 "Commit" **或** "Rollback" **语句将验证或取消对数据库进行的所有修改。**

API 语法

long AmStartTransaction(long hApiCnxBase);

内部 Basic 语法

Function AmStartTransaction() As Long

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	

	可用
向导的 FINISH.DO 脚本	✓

输出参数

- 0：正常执行。
- 非 0 值：错误代码。

AmStartup()

必须在其他所有函数之前应用此函数。它初始化对 AssetCenter 库的调用。

API 语法

void AmStartup();

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	
向导脚本	
向导的 FINISH.DO 脚本	

AmTableDesc()

此函数根据表的 SQL 名称生成"<表的描述>（<表的 SQL 名称>）"格式的字符串。

API 语法

long AmTableDesc(long hApiCnxBase, char *return, long lreturn, char *strSqlName);

内部 Basic 语法

Function AmTableDesc(strSqlName As String) As String

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strSqlName：此参数包含描述字符串所需的表的** SQL **名称。如果此参数包含无效的** SQL **名称，则函数返回一个问号** ("?")。

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

下例将生成资产表（SQL **名称：**amAsset**）的描述字符串：**

```
AmTableDesc("amAsset")
```

结果如下：

```
Assets (amAsset)
```

AmTaxRate()

此函数根据税收类型、税收管辖区和日期计算税率。

API 语法

double AmTaxRate(long hApiCnxBase, char *strTaxRateName, long lTaxLocId, long tmDate, double dValue);

内部 Basic 语法

Function AmTaxRate(strTaxRateName As String, lTaxLocId As Long, tmDate As Date, dValue As Double) As Double

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strTaxRateName：此参数包含用于计算税率的税收类型的** SQL **名称。**
- **lTaxLocId：此参数包含与税收类型有关的税收管辖区的** ID **号。**
- **tmDate：此参数包含要获取税率的日期。**
- **dValue：此参数包含过时的参数，出于兼容性原因而保留。将此参数设置为您选择的值。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

AmUpdateDetail()

此函数在数据输入向导中使用。因此可明确定义上下文（使用向导更新或填充的记录所在的表）。此函数根据值更新或填充上下文的字段或链接。在非模式向导中不允许使用此函数。

内部 Basic 语法

Function AmUpdateDetail(strFieldName As String, varValue As Variant) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	
"Script" 类型操作	
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **strFieldName：此参数包含要更新的特征值的** SQL **名称。**
- **varValue：此参数包含字段的新值。**

输出参数

- 0**：正常执行。**
- **非** 0 **值：错误代码。**

AmUpdateLossLines()

此函数可使用由 lLossValId 标识符引用的损失价值规则更新合同的所有损失价值。

内部 Basic 语法

Function AmUpdateLossLines(lLossValId As Long) As Long

应用的字段

版本: 4.3.0

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **lLossValId：此参数包含损失价值规则的标识符。**

输出参数

- 0：**正常执行。**
- **非** 0 **值：错误代码。**

AmUpdateRecord()

此函数可用于更新记录。

API 语法

long AmUpdateRecord(long hApiRecord);

内部 Basic 语法

Function AmUpdateRecord(hApiRecord As Long) As Long

应用的字段

版本: 2.52

	可用
AssetCenter API	✓
字段或链接的配置脚本	
"Script" 类型操作	✓
向导脚本	
向导的 FINISH.DO 脚本	✓

输入参数

- **hApiRecord：此参数包含要更新字段所在记录的句柄。**

输出参数

- 0：正常执行。
- 非 0 值：错误代码。

AmUpdateUser()

此信息将更新数据库中与员工有关的信息（域、NT 用户名、描述）。

API 语法

long AmUpdateUser(long hApiCnxBase, long lId, char *strNTUserName, char *strNTDomain, char *strNTUserDesc);

内部 Basic 语法

Function AmUpdateUser(lId As Long, strNTUserName As String, strNTDomain As String, strNTUserDesc As String) As Long

应用的字段

版本: 4.3.0

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **lId**：此参数包含数据库中有关员工的标识符。
- **strNTUserName**：此参数包含员工的 NT 用户名。
- **strNTDomain**：此参数包含员工所在的 NT 域名称。
- **strNTUserDesc**：此参数包含与 NT 用户相关的描述。

输出参数

- 0：正常执行。

- **非 0 值：错误代码。**

AmValueOf()

在向导中使用此函数返回由 strVarName 参数标识的属性的值。

内部 Basic 语法

Function AmValueOf(strVarName As String) As Variant

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	
"Script" 类型操作	
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strVarName：此参数包含要获取其值的属性的名称。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

下例将返回 "Page1.Label" **属性的值：**

```
AmValueOf("Page1.Label")
```

请慎重使用此函数，因为其可能破坏要处理的属性的依赖关系字符串的完整性。

AmWizChain()

此函数在向导 A 内部执行向导 B。结束执行向导 B 时，将重新接着执行向导 A。

内部 Basic 语法

Function AmWizChain(strWizSqlName As String) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	
"Script" 类型操作	
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strWizSqlName：此参数包含要执行的向导的 SQL 名称。**

输出参数

- 0：**正常执行。**
- **非 0 值：错误代码。**

AmWorkTimeSpanBetween()

此函数返回在两个日期之间的工作期间的持续时间。此持续时间按秒表示；它参考了工作期间日历的有关信息。

API 语法

long AmWorkTimeSpanBetween(long hApiCnxBase, char *strCalendarSqlName, long tmEnd, long tmStart);

内部 Basic 语法

Function AmWorkTimeSpanBetween(strCalendarSqlName As String, tmEnd As Date, tmStart As Date) As Date

应用的字段

版本: 3.00

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strCalendarSqlName：此参数包含用于计算在两个日期之间的工作期间的持续时间的工作期间日历的 SQL 名称。如果忽略此参数，则计算持续时间时将不考虑工作期间。**
- **tmEnd：此参数包含计算工作期间时使用的期间的截止日期。**
- **tmStart：此参数包含计算工作期间时使用的期间的开始日期。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

下例将计算在 1998 年 9 月 1 日上午 8 点 和 1998 年 9 月 24 日下午 7 点之间的工作期间。使用的日历的 SQL 名称是 "Calendar_Paris"，定义了如下工作期间：

- **每周一到周四，从上午 8 点到中午 12 点，然后从下午 2 点到 6 点。**
- **每周五，从上午 8 点到中午 12 点，然后从下午 2 点到 5 点。**

```
AmWorkTimeSpanBetween("Calendar_Paris", "1998/09/24 19:00:00", "1998/09/01 08:00:00")
```

此例返回的值为 507,600，表示在两个期间之间的工作期间的秒数。

AppendOperand()

根据传递给函数的参数连接字符串。给出的结果如下：

strExpr strOperator strOperand

内部 Basic 语法

Function AppendOperand(strExpr As String, strOperator As String, strOperand As String) As String

应用的字段

版本: 3.5

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strExpr：要连接的表达式。**
- **strOperator：要连接的运算符。**
- **strOperand：要连接的操作数。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

备注

注:

如果忽略一个 strExpr 或 strOperand 参数，则连接中不会使用 strOperator。

ApplyNewVals()

为 "ListBox" 控制中相同的单元赋相同的值。

内部 Basic 语法

Function ApplyNewVals(strValues As String, strNewVals As String, strRows As String, strRowFormat As String) As String

应用的字段

版本: 3.5

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strValues：包含要处理的 "ListBox" 控制的源字符串。**
- **strNewVals：要赋给所连接单元的新值。**
- **strRows：要处理的行的标识符。标识符以逗号分隔。**
- **strRowFormat：子列表的格式化指令。该指令以 "|" 字符分隔。每条指令都代表包含 strNewVals 参数的列数。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**

- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

Asc()

为字符串中的第一个字符返回 ASCII **代码数值。**

内部 Basic 语法

Function Asc(strAsc As String) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strAsc：函数运算的字符串。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim iCount as Integer
Dim strString as String
For iCount=Asc("A") To Asc("Z")
strString = strString & Str(iCount)
```

```
Next iCount
RetVal=strString
```

Atn()

返回以弧度表示的数值的反正切。

内部 Basic 语法

Function Atn(dValue As Double) As Double

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **dValue：想知道其反正切的数值。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim dPi as Double
Dim strString as String
dPi=4*Atn(1)
```

```
strString = Str(dPi)
RetVal=strString
```

BasicToLocalDate()

此函数将基本格式日期转换为字符串格式（在Windows控制面板中显示的格式）日期。

内部 Basic 语法

Function BasicToLocalDate(strDateBasic As String) As String

应用的字段

版本: 3.5

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strDateBasic：要转换的基本格式日期。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

BasicToLocalTime()

此函数将基本格式时间转换为字符串格式（在Windows控制面板中显示的格式）时间。

内部 Basic 语法

Function BasicToLocalTime(strTimeBasic As String) As String

应用的字段

版本: 3.5

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strTimeBasic：要转换的基本格式时间。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

BasicToLocalTimeStamp()

此函数将基本格式的日期和时间转换为字符串格式（在 Windows**控制面板中显示的格式）的日期和时间。**

内部 Basic 语法

Function BasicToLocalTimeStamp(strTSBasic As String) As String

应用的字段

版本: 3.5

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strTSBasic：要转换的基本格式的日期和时间。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

Beep()

在机器上播放嘟嘟声。

内部 Basic 语法

Function Beep()

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

CDbl()

将表达式转换为 "Double"。

内部 Basic 语法

Function CDbl(dValue As Double) As Double

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **dValue：要转换的表达式。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim dNumber As Double
Dim iInteger as Integer
iInteger = 25
```

```
dNumber=CDbl(iInteger)
RetVal=dNumber
```

ChDir()

更改当前目录。

内部 Basic 语法

Function ChDir(strDirectory As String)

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strDirectory：新的当前目录。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

ChDrive()

更改当前驱动器。

内部 Basic 语法

Function ChDrive(strDrive As String)

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strDrive：新驱动器名称。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

Chr()

返回由 iChr 参数传递的对应 ASCII **的字符串。**

内部 Basic 语法

Function Chr(lChr As Long) As String

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **IChr：字符的** ASCII **代码。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
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()

将任何有效表达式都转换为 Integer**。**

内部 Basic 语法

Function CInt(iValue As Long) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **iValue：要转换的表达式。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim iNumber As Integer
Dim dDouble as Double
dDouble = 25.24589
iNumber=CInt(dDouble)
RetVal=iNumber
```

CLng()

将任何有效表达式都转换为 Long**。**

内部 Basic 语法

Function CLng(lValue As Long) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **lValue：要转换的表达式。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim lNumber As Long
Dim iInteger as Integer
iInteger = 25
lNumber=CLng(iInteger)
RetVal=iNumber
```

Cos()

返回以弧度表示的值的余弦。

内部 Basic 语法

Function Cos(dValue As Double) As Double

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **dValue：想知道其余弦的数值。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

备注

注:

角度转换为弧度的转换公式如下：

弧度 = **角度** * Pi / 180

示例

```
Dim dCalc as Double
dCalc=Cos(2.79)
RetVal=dCalc
```

CountOccurences()

计算某字符串在其他字符串内出现的次数。

内部 Basic 语法

Function CountOccurences(strSearched As String, strPattern As String, strEscChar As String) As Long

应用的字段

版本: 3.5

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strSearched：执行搜索的字符串。**
- **strPattern：要在 strSearched 参数中寻找的字符串。**
- **strEscChar：转义字符。如果该函数在 strSearched 字符串中遇到此字符，将停止搜索。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim MyStr
MyStr=CountOccurences("你|我|你,我|你", "你", ",") :'返回 "2"
MyStr=CountOccurences("你|我|你,我|你", "我", "|") :'返回 "1"
```

CountValues()

计算字符串中的元素数，分隔符和转义字符也计算在内。

内部 Basic 语法

Function CountValues(strSearched As String, strSeparator As String, strEscChar As String) As Long

应用的字段

版本: 3.5

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strSearched：要处理的字符串。**
- **strSeparator：用于划分元素的分隔符。**
- **strEscChar：转义字符。如果此字符前面有分隔符，则将被忽略。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim MyStr
MyStr=CountValues("你|我|你\|我|你", "|", "\") :'返回 4
MyStr=CountValues("你|我|你\|我|你", "|", "") :'返回 5
```

CSng()

将任何有效表达式转换为浮点型数值 ("Float")。

内部 Basic 语法

Function CSng(fValue As Single) As Single

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **fValue：要转换的表达式。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim dNumber As Double
Dim iInteger as Integer
iInteger = 25
dNumber=CSng(iInteger)
RetVal=dNumber
```

CStr()

将任何有效表达式都转换为字符串。

内部 Basic 语法

Function CStr(strValue As String) As String

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strValue：要转换的表达式。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim dNumber As Double
Dim strMessage as String
dNumber = 2,452873
strMessage=CStr(dNumber)
RetVal=strMessage
```

CurDir()

返回当前路径。

内部 Basic 语法

Function CurDir() As String

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

CVar()

将任何有效表达式都转换为变量。

内部 Basic 语法

Function CVar(vValue As Variant) As Variant

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **vValue：要转换的表达式。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

Date()

返回当前系统日期。

内部 Basic 语法

Function Date() As Date

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

DateAdd()

此函数根据添加真实期限时的起始日期计算新日期。

内部 Basic 语法

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

应用的字段

版本: 2.51

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **tmStart：此参数包含添加期限的日期。**
- **tsDuration：此参数包含在 tmStart 日期添加的按秒表示的期限。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

DateAddLogical()

此函数根据添加逻辑期限（1 **个月包含** 30 **天）时的起始日期计算新日期。**

内部 Basic 语法

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

应用的字段

版本: 2.51

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **tmStart：此参数包含添加期限的日期。**
- **tsDuration：此参数包含在 tmStart 日期添加的期限（以秒表示）。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

DateDiff()

此函数计算这两个日期之间的期限或时间跨度（以秒为单位）。

内部 Basic 语法

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

应用的字段

版本: 2.51

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **tmEnd：此参数包含要计算的期间的结束日期。**
- **tmStart：此参数包含要计算的期间的起始日期。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

下例中计算的起止时间为 1998 **年** 1 **月** 1 **日和** 1999 **年** 1 **月** 1 **日。**

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

DateSerial()

此函数根据 iYear、iMonth 和 iDay 参数返回格式化日期。

内部 Basic 语法

Function DateSerial(iYear As Long, iMonth As Long, iDay As Long) As Date

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **iYear**: 年。如果值介于 0 和 99 之间，此参数将描述从 1900 年至 1999 年之间的年份。对于所有其他年份，必须指定四位数（如 1800）。
- **iMonth**: 月。
- **iDay**: 日。

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

这些参数中的每一个都可以设置为表示日、月或年的数值表达式。例如：

```
DateSerial(1999-10, 3-2, 15-8)
```

返回值：

```
1989/1/7
```

当参数值超出预期范围（例如，1 至 31 天，1 至 12 个月等）时，函数会返回空日期。

DateValue()

此函数返回 "日期+时间" 值的日期部分。

内部 Basic 语法

Function DateValue(tmDate As Date) As Date

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **tmDate：**"日期+时间" 格式日期。

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

例如：

```
DateValue ("1999/09/24 15:00:00")
```

返回值：

```
1999/09/24
```

Day()

返回 tmDate 参数中包含的日期。

内部 Basic 语法

Function Day(tmDate As Date) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **tmDate：要处理的"日期+时间"格式的参数。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim strDay as String
strDay=Day(Date())
RetVal=strDay
```

EnumToComboBox()

此函数重组了可用的逐项列表项以使其格式能够与向导控制列表兼容。这样就可以在向导的下拉列表中显示可用的逐项列表的值。

内部 Basic 语法

Function EnumToComboBox(strFormat As String) As String

应用的字段

版本: 4.3.0

	可用
AssetCenter API	

	可用
字段或链接的配置脚本	
"Script" 类型操作	
向导脚本	
向导的 FINISH.DO 脚本	

输入参数

- **strFormat：此参数包含系统逐项列表的条目列表。如果此参数包含 AmDbGetList() 函数的执行结果更好。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

下例将获取标识符为 amWOPriority **的可用逐项列表的值并按照与向导控制列表兼容的格式对其进行重组：**

```
Dim strValues As String
strValues = AmDbGetList("SELECT Value FROM amItemListVal WHERE ItemizedList.Identifier = 'amWOPriority'", "", ",", "")
RetVal = EnumToComboBox(strValues)
```

EscapeSeparators()

前面带有一个或多个带有转义字符的分隔符字符。

内部 Basic 语法

Function EscapeSeparators(strSource As String, strSeparators As String, strEscChar As String) As String

应用的字段

版本: 3.5

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strSource：要处理的字符串。**
- **strSeparators：要加上前缀的分隔符列表。如果要声明多个分隔符，必须用转义字符将它们分开（在 strEscChar 参数中指出）。**
- **strEscChar：转义字符。将用于 strSeparators 中所有分隔符的前缀。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim MyStr
MyStr=EscapeSeparators("你|我|你,我|你", "|\,", "\") :'返回 "你\|我\|你\,我\|你"
```

ExeDir()

此函数返回可执行文件的完整路径。

内部 Basic 语法

Function ExeDir() As String

应用的字段

版本: 3.60

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim strPath as string
strPath=ExeDir()
```

Exp()

返回数值的指数。

内部 Basic 语法

Function Exp(dValue As Double) As Double

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **dValue：想知道其指数的数值。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

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

ExtractValue()

提取字符串的值，并用分隔符分隔。从源字符串中删除恢复的值。此操作可能会使用转义字符。如果源字符串中没有分隔符，则会返回整个字符串，并删除整个源字符串。

内部 Basic 语法

Function ExtractValue(pstrData As String, strSeparator As String, strEscChar As String) As String

应用的字段

版本: 3.5

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓

	可用
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **pstrData：要处理的源字符串。**
- **strSeparator：在源字符串中用作分隔符的字符。**
- **strEscChar：转义字符。如果此字符前面有分隔符，则将被忽略。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim MyStr
MyStr=ExtractValue("你,我", ",", "\") :'返回 "你" 并在源字符串中保留 "我"
MyStr=ExtractValue(",你,我", ",", "\") :'返回 "" 并在源字符串中保留 "你,我"
MyStr=ExtractValue("你", ",", "\") :'返回 "你" 并在源字符串中保留 ""
MyStr=ExtractValue("你\,我", ",", "\") :'返回 "你\,我" 并在源字符串中保留 ""
MyStr=ExtractValue("你\,我", ",", "") :'返回 "你\" 并在源字符串中保留 "我"
RetVal=""
```

FileCopy()

复制文件或文件夹。

内部 Basic 语法

Function FileCopy(strSource As String, strDest As String) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strSource：要复制的文件或目录的完整路径。**
- **strDest：目标文件或目录的完整路径。**

输出参数

- 0：**正常执行。**
- **非** 0 **值：错误代码。**

FileDateTime()

以长整型返回文件的时间和日期。

内部 Basic 语法

Function FileDateTime(strFileName As String) As Date

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strFileName：操作相关文件的完整路径名。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

FileExists()

此函数测试文件是否存在。此函数将返回下列值：

- **0：找不到文件。**
- **1：已找到文件。**

内部 Basic 语法

Function FileExists(strFileName As String) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strFileName：此参数包含要测试的文件的完整路径。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
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()

返回文件的大小。

内部 Basic 语法

Function FileLen(strFileName As String) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strFileName：操作相关文件的完整路径名。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

Fix()

返回数值的整数部分（如果是负数，则返回第一个最大的整数）。

内部 Basic 语法

Function Fix(dValue As Double) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **dValue：想知道其整数部分的数值。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim dSeed as Double
dSeed = (10*Rnd)-5
RetVal = Fix(dSeed)
```

FormatDate()

根据 strFormat 参数中包含的表达式格式化日期。

内部 Basic 语法

Function FormatDate(tmFormat As Date, strFormat As String) As String

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **tmFormat：要格式化的日期。**
- **strFormat：包含格式化指令的表达式。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

下列代码示例说明如何格式化日期：

```
Dim MyDate
MyDate="2000/03/14"
RetVal=FormatDate(MyDate, "dddd d mmmm yyyy") :'返回 "Tuesday 14 March 2000"
```

FormatResString()

此函数处理源字符串，并用下列参数传递的字符串替换 $1、$2、$3、$4 **和** $5 **变量：strParamOne、strParamTwo、strParamThree、strParamFour、和 strParamFive 参数。**

内部 Basic 语法

Function FormatResString(strResString As String, strParamOne As String, strParamTwo As String, strParamThree As String, strParamFour As String, strParamFive As String) As String

应用的字段

版本: 3.5

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strResString：要处理的源字符串。**
- **strParamOne：变量** $1 **的替换字符串。**
- **strParamTwo：变量** $2 **的替换字符串。**
- **strParamThree：变量** $3 **的替换字符串。**
- **strParamFour：变量** $4 **的替换字符串。**
- **strParamFive：变量** $5 **的替换字符串。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

例如：

```
FormatResString("我$1他$2你$3", "你", "我们", "他们")
```

返回 "我你他你他们"。

FV()

此函数根据常量和定期租金以设定的利率返回将来的年金数额。

内部 Basic 语法

Function FV(dblRate As Double, iNper As Long, dblPmt As Double, dblPV As Double, iType As Long) As Double

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **dblRate：此参数表示付款日期的利率。例如，如果某项贷款的年利率为 6%，那么每月付款日期偿还的利率为：**

```
0.06/12=0.005 或 0.5%
```

- **iNper：此参数包含财务操作的付款日期总数。**
- **dblPmt：此参数指出每个付款日期要支付的数额。租金一般包含本金和利息。**
- **dblPV：此参数包含将来要支付的一系列租金的实际值或总和。**
- **iType：此参数表示付款的最终期限。其值可以为以下值之一：**
  - **0 如果拖欠付款（如期间结束）**
  - **1 如果提前付款（如期间开始）**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

备注

注:

- **Rate 和 Nper 参数必须用以相同单位表示的租金计算。**
- **支付金额（特别是以 Pmt 参数表示的金额）以负数表示。收到的总和以正数表示。**

GetEnvVar()

此函数返回环境变量的值。如果环境变量不存在，则返回空值。

内部 Basic 语法

Function GetEnvVar(strVar As String, bExpand As Long) As String

应用的字段

版本: 3.2.0

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strVar：此参数包含环境变量的名称。**

- **bExpand：当环境变量参考一个或多个环境变量时，可使用此布尔参数。在这种情况下，当此参数设为1（默认值）时，会用其值替换每个参考变量。否则，环境变量不变。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
RetVal = getEnvVar("PROMPT")
```

GetListItem()

返回用分隔符分隔的字符串的第 INb 部分。

内部 Basic 语法

Function GetListItem(strFrom As String, strSep As String, lNb As Long, strEscChar As String) As String

应用的字段

版本: 3.5

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strFrom：要处理的源字符串。**
- **strSep：在源字符串中用作分隔符的字符。**

- **INb：要恢复的字符串的位置。**
- **strEscChar：转义字符。如果此字符前面有分隔符，则忽略此字符。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

例如：

```
GetListItem("this_is_a_test", "_", 2, "%")
```

返回 "is"。

```
GetListItem("this%_is_a_test", "_", 2, "%")
```

返回 "a"。

Hex()

返回数字参数的十六进制值。

内部 Basic 语法

Function Hex(dValue As Double) As String

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **dValue：想知道其十六进制值的数字。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

Hour()

返回 tmTime 参数中包含的时间值。

内部 Basic 语法

Function Hour(tmTime As Date) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **tmTime：要处理的"日期+时间"格式的参数。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim strHour as String
strHour=Hour(Date())
RetVal=strHour
```

InStr()

返回某字符串在另一字符串中第一次出现时的字符位置。

内部 Basic 语法

Function InStr(lPosition As Long, strSource As String, strPattern As String, bCaseSensitive As Long) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **lPosition：搜索的起点。此参数不可选，必须是有效的正整数，且不得大于 65,535。**
- **strSource：执行搜索的字符串。**
- **strPattern：搜索的字符串。**

- **bCaseSensitive：根据此参数确定搜索是区分大小写**(=1)**或不区分大小写**(=0)**。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

备注

注:

第一次出现时的位置为 1**。如果未找到搜索的字符串，函数会返回** 0**。**

示例

```
Dim strSource as String
Dim strToSearch as String
Dim iPosition
strSource = "Good Bye"
strToSearch = "Bye"
iPosition = Instr(2, strSource, strToSearch)
RetVal=iPosition
```

Int()

返回数值的整数部分（如果是负数，则返回第一个较小的整数）。

内部 Basic 语法

Function Int(dValue As Double) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **dValue：想知道其整数部分的数值。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

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

IPMT()

此函数返回年金指定付款日期的利息额。

内部 Basic 语法

Function IPMT(dblRate As Double, iPer As Long, iNper As Long, dblPV As Double, dblFV As Double, iType As Long) As Double

应用的字段

版本: 3.00

	可用
AssetCenter API	

	可用
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **dblRate：此参数表示付款日期的利率。例如，如果某项贷款的年利率为** 6%，**那么每月付款日期偿还的利率为：**

0.06/12=0.005 **或** 0.5%

- **iPer：此参数指出计算期间，介于** 1 **和 Nper 参数值之间。**
- **iNper：此参数包含财务操作的付款日期总数。**
- **dblPV：此参数包含将来要支付的一系列租金的实际值或总和。**
- **dblFV：此参数包含最后付款日期支付之后要获得的未来值或余额。通常情况下，尤其是偿还贷款时，此参数设为** "0"。**实际上，指定所有付款日期之后，贷款的值为零。**
- **iType：此参数表示付款的最终期限。其值可以为以下值之一：**
  - **0 如果拖欠付款（如期间结束）**
  - **1 如果提前付款（如期间开始）**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

备注

注:

- **Rate 和 Nper 参数必须以相同单位表示的租金计算。**
- **支付金额（尤其是以 Pmt 参数表示的金额）以负数表示。收到的总和以正数表示。**

IsNumeric()

此函数可用来确定字符串是否包含数字值。

内部 Basic 语法

Function IsNumeric(strString As String) As Long

应用的字段

版本: 3.5

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strString：此参数包含要分析的字符串。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

Kill()

删除文件。

内部 Basic 语法

Function Kill(strKilledFile As String) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strKilledFile：运算相关文件的完整路径。**

输出参数

- 0：**正常执行。**
- **非 0 值：错误代码。**

LCase()

返回字符串，该字符串参数的所有字母都已转换为小写字母。

内部 Basic 语法

Function LCase(strString As String) As String

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **strString：要转换为小写字母的字符串。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
' 此示例分别用 LTrim 和 RTrim 函数去除字符串变量中前导的 ' 和结尾空格。
' 单独使用 Trim 函数去除两种类型的空格。
' 此示例还将说明 LCase 和 UCase，同时说明的还有
' 嵌套函数调用的用法

Dim strString as String
Dim strTrimString as String
strString = "  <-Trim->  " :' 初始化字符串。
strTrimString = LTrim(strString) :' strTrimString = "<-Trim->  ".
strTrimString = LCase(RTrim(strString)) :' strTrimString = "  <-trim->".
strTrimString = LTrim(RTrim(strString)) :' strTrimString = "<-Trim->".
' 单独使用 Trim 函数达到同样的效果。
strTrimString = UCase(Trim(strString)) :' strTrimString = "<-TRIM->".
RetVal= "|" & strTrimString & "|"
```

Left()

返回字符串参数最左边的 iNumber **字符。**

内部 Basic 语法

Function Left(strString As String, lNumber As Long) As String

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strString：要处理的字符串。**
- **lNumber：要返回的字符数量。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim lWord, strMsg, rWord, iPos :' 声明变量。
strMsg = "Left() Test."
iPos = InStr(1, strMsg, " ") :' 查找空格。
lWord = Left(strMsg, iPos - 1) :' 获取左边的词。
rWord = Right(strMsg, Len(strMsg) - iPos) :' 获取右边的词。
strMsg=rWord+lWord :' 然后互换
RetVal=strMsg
```

LeftPart()

将字符串的某部分提取到 strSep 参数中指定的分隔符的左侧。
搜索分隔符时会从左向右进行。
使用 bCaseSensitive 参数搜索可以设置为区分大小写。

内部 Basic 语法

Function LeftPart(strFrom As String, strSep As String, bCaseSensitive As Long) As String

应用的字段

版本: 3.5

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strFrom：要处理的源字符串。**
- **strSep：在源字符串中用作分隔符的字符。**
- **bCaseSensitive：根据此参数确定搜索是区分大小写** (=1) **或不区分大小写** (=0)**。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

这些示例说明了 LeftPart、LeftPartFromRight、RightPart 和 RightPartFromLeft 函数的用法，这些函数同属一个字符串： "This_is_a_test":

```
LeftPart("This_is_a_test","_",0)
```

返回 "This"。

```
LeftPartFromRight("This_is_a_test","_",0)
```

返回 "This_is_a"。

```
RightPart("This_is_a_test","_",0)
```

返回 "test"。

```
RightPartFromLeft("This_is_a_test","_",0)
```

返回 "is_a_test"。

LeftPartFromRight()

将字符串的某部分提取到 strSep 参数中指定的分隔符的左侧。
搜索分隔符时会从右向左进行。
使用 bCaseSensitive 参数搜索可以设置为区分大小写。

内部 Basic 语法

Function LeftPartFromRight(strFrom As String, strSep As String, bCaseSensitive As Long) As String

应用的字段

版本: 3.5

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strFrom：要处理的源字符串。**
- **strSep：在源字符串中用作分隔符的字符。**
- **bCaseSensitive：根据此参数确定搜索是区分大小写** (=1) **或不区分大小写** (=0)**。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

这些示例说明了 LeftPart、LeftPartFromRight、RightPart 和 RightPartFromLeft 函数的用法，这些函数同属一个字符串："This_is_a_test":

```
LeftPart("This_is_a_test","_",0)
```

返回 "This"。

```
LeftPartFromRight("This_is_a_test","_",0)
```

返回 "This_is_a"。

```
RightPart("This_is_a_test","_",0)
```

返回 "test"。

```
RightPartFromLeft("This_is_a_test","_",0)
```

返回 "is_a_test"。

Len()

返回字符串或变量中的字符数。

内部 Basic 语法

Function Len(vValue As Variant) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **vValue：运算相关的变量。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim strTest as String
Dim iLength as Integer
strTest = "Peregrine Systems"
iLength = Len(strTest) :'iLength 的值为 17
RetVal=iLength
```

LocalToBasicDate()

此函数将字符串格式（在 Windows**控制面板中显示的格式）日期转换为基本格式日期。**

内部 Basic 语法

Function LocalToBasicDate(strDateLocal As String) As String

应用的字段

版本: 3.5

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strDateLocal：要转换的字符串格式的日期。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

LocalToBasicTime()

此函数将字符串格式（在 Windows**控制面板中显示的格式）时间转换为基本格式时间。**

内部 Basic 语法

Function LocalToBasicTime(strTimeLocal As String) As String

应用的字段

版本: 3.5

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strTimeLocal：要转换的字符串格式时间。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

LocalToBasicTimeStamp()

此函数将字符串格式（在Windows控制面板中显示的格式）的日期和时间转换为基本格式的日期和时间。

内部 Basic 语法

Function LocalToBasicTimeStamp(strTSLocal As String) As String

应用的字段

版本: 3.5

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strTSLocal：要转换的字符串格式的日期和时间。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

LocalToUTCDate()

此函数将"日期+时间"格式的日期转换为 UTC **格式日期（不受时区限制)。**

内部 Basic 语法

Function LocalToUTCDate(tmLocal As Date) As Date

应用的字段

版本: 3.5

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **tmLocal：**"日期+时间" 格式日期。

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

Log()

返回数值的自然对数。

内部 Basic 语法

Function Log(dValue As Double) As Double

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **dValue：想知道其对数的数值。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim dSeed as Double
dSeed = Int((10*Rnd)-5)
RetVal = Log(dSeed)
```

LTrim()

移除字符串中的所有前导空格。

内部 Basic 语法

Function LTrim(strString As String) As String

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **strString：要处理的字符串。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
' 此示例分别用 LTrim 和 RTrim 函数去除字符串变量中前导的 ' 和结尾空格。
' 单独使用 Trim 函数去除两种类型的空格。
' 此示例还将说明 LCase 和 UCase，同时说明的还有
' 嵌套函数调用的用法

Dim strString as String
Dim strTrimString as String
strString = "  <-Trim->  " :' 初始化字符串。
strTrimString = LTrim(strString) :' strTrimString = "<-Trim->  ".
strTrimString = LCase(RTrim(strString)) :' strTrimString = "  <-trim->".
strTrimString = LTrim(RTrim(strString)) :' strTrimString = "<-Trim->".
' 单独使用 Trim 函数达到同样的效果。
strTrimString = UCase(Trim(strString)) :' strTrimString = "<-TRIM->".
RetVal= "|" & strTrimString & "|"
```

MakeInvertBool()

此函数返回相反的布尔值；（0 变为 1，所有其他数值变为 0）。

内部 Basic 语法

Function MakeInvertBool(lValue As Long) As Long

应用的字段

版本: 3.5

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **lValue：运算相关的数值。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim MyValue
MyValue=MakeInvertBool(0) :'返回 1
MyValue=MakeInvertBool(1) :'返回 0
MyValue=MakeInvertBool(254) :'返回 0
```

Mid()

返回字符串内的子字符串。

内部 Basic 语法

Function Mid(strString As String, lStart As Long, lLen As Long) As String

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strString：运算相关的字符串。**
- **lStart：要从** strString **提取的字符串的开始部分。**
- **lLen：要提取的字符串的长度。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim strTest as String
strTest="One Two Three" :' 定义测试字符串
strTest=Mid(strTest,5,3) :' strTest="Two"
RetVal=strTest
```

Minute()

返回以 tmTime 参数表示的时间中包含的分钟数。

内部 Basic 语法

Function Minute(tmTime As Date) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **tmTime：要处理的"日期+时间"格式的参数。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim strMinute
strMinute=Minute(Date())
RetVal=strMinute :'返回当前小时数内逝去的分钟数，例如，"45" 分钟，如果时间为 15:45:30 的话
```

MkDir()

创建新目录。

内部 Basic 语法

Function MkDir(strMkDirectory As String) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strMkDirectory：要创建的目录的完整路径。**

输出参数

- 0：**正常执行。**
- **非** 0 **值：错误代码。**

示例

```
Dim lErr as Long
' 创建 c:\tmp 目录
lErr = MkDir("c:\tmp")
```

Month()

返回 tmDate 参数表示的日期中包含的月份。

内部 Basic 语法

Function Month(tmDate As Date) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **tmDate：要处理的"日期+时间"格式的参数。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim lMonth as Long
lMonth=Month(Date())
RetVal=lMonth :'返回当前月份
```

Name()

更改文件名。

内部 Basic 语法

Function Name(strSource As String, strDest As String)

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓

	可用
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strSource：要重命名的文件的完整路径。**
- **strDest：新文件名。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim lErr as Long
' 将 "C:\tmp\src.txt" 重命名为 "D:\tmp\dst.txt"
lErr = Name("C:\tmp\src.txt", "D:\tmp\dst.txt")
```

Now()

返回当前日期和时间。

内部 Basic 语法

Function Now() As Date

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓

	可用
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

NPER()

此函数根据常量和定期租金以设定的利率返回年金的支付数额。

内部 Basic 语法

Function NPER(dblRate As Double, dblPmt As Double, dblPV As Double, dblFV As Double, iType As Long) As Double

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **dblRate：此参数表示付款日期的利率。例如，如果某项贷款的年利率为 6%，那么每月付款日期偿还的利率为：**

  0.06/12=0.005 **或** 0.5%

- **dblPmt：此参数指出每个付款日期要支付的数额。租金一般包含本金和利息。**
- **dblPV：此参数包含将来要支付的一系列租金的实际值或总和。**

- **dblFV：此参数包含最后付款日期支付之后要获得的未来值或余额。通常情况下，尤其是偿还贷款时，此参数设为** "0"**。实际上，指定所有付款日期之后，贷款的值为零。**
- **iType：此参数表示付款的最终期限。其值可以为以下值之一：**
  - **0 如果拖欠付款（如期间结束）**
  - **1 如果提前付款（如期间开始）**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

备注

注:

支付金额（尤其是以Pmt参数表示的金额）以负数表示。收到的总和以正数表示。

Oct()

返回数字参数的八进制值。

内部 Basic 语法

Function Oct(dValue As Double) As String

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **dValue：想知道其八进制值的数值。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim dSeed as Double
dSeed = Int((10*Rnd)-5)
RetVal = Oct(dSeed)
```

ParseDate()

此函数将表示为字符串的日期转换为基本日期对象。

内部 Basic 语法

Function ParseDate(strDate As String, strFormat As String, strStep As String) As Date

应用的字段

版本: 3.6.0

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **strDate：字符串格式的日期。**
- **strFormat：此参数包含字符串中包含的日期格式。可能的值如下：**
  - DD/MM/YY
  - DD/MM/YYYY
  - MM/DD/YY
  - MM/DD/YYYY
  - YYYY/MM/DD
  - Date：**根据客户端计算机设置表示的日期。**
  - DateInter：**以国际格式表示的日期**
- **strStep：此可选参数包含字符串中使用的日期分隔符。认可的分隔符为** "\" **和** "-"**。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim dDate as date
dDate=ParseDate("2001/05/01", "YYYY/MM/DD")
```

ParseDMYDate()

此函数从如下格式的日期中返回 Date **对象（同** Basic **格式）：**

```
dd/mm/yyyy
```

内部 Basic 语法

Function ParseDMYDate(strDate As String) As Date

应用的字段

版本: 3.5

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strDate：存储为字符串的日期。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim dDate as Date
dDate = ParseDMYDate("31/02/2003")
```

ParseMDYDate()

此函数从如下格式的日期中返回 Date **对象（同** Basic **格式）：**

```
mm/dd/yyyy
```

内部 Basic 语法

Function ParseMDYDate(strDate As String) As Date

应用的字段

版本: 3.5

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strDate：存储为字符串的日期。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim dDate as Date
dDate = ParseMDYDate("02/31/2003")
```

ParseYMDDate()

此函数返回 yyyy/mm/dd **格式的** Date **对象（同** Basic **格式）。**

内部 Basic 语法

Function ParseYMDDate(strDate As String) As Date

应用的字段

版本: 3.5

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓

	可用
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strDate：存储为字符串的日期。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim dDate as Date
dDate = ParseYMDDate("2003/02/31")
```

PMT()

此函数根据常量和定期租金以设定的利率返回年金数额。

内部 Basic 语法

Function PMT(dblRate As Double, iNper As Long, dblPV As Double, dblFV As Double, iType As Long) As Double

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **dblRate：此参数表示付款日期的利率。例如，如果某项贷款的年利率为 6%，那么每月付款日期偿还的利率为：**

0.06/12=0.005 **或** 0.5%

- **iNper：此参数包含财务操作的付款日期总数。**
- **dblPV：此参数包含将来要支付的一系列租金的实际值或总和。**
- **dblFV：此参数包含最后付款日期支付之后要获得的未来值或余额。通常情况下，尤其是偿还贷款时，此参数设为 "0"。实际上，指定所有付款日期之后，贷款的值为零。**
- **iType：此参数表示付款的最终期限。其值可以为以下值之一：**
  - **0 如果拖欠付款（如期间结束）**
  - **1 如果提前付款（如期间开始）**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

备注

注:

- **Rate 和 Nper 参数必须以相同单位表示的租金计算。**
- **支付金额（尤其是以 Pmt 参数表示的金额）以负数表示。收到的总和以正数表示。**

PPMT()

此函数根据常量和定期租金以设定的利率，返回指定年金付款日期所偿还的本金数额。

内部 Basic 语法

Function PPMT(dblRate As Double, iPer As Long, iNper As Long, dblPV As Double, dblFV As Double, iType As Long) As Double

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **dblRate：此参数表示付款日期的利率。例如，如果某项贷款的年利率为** 6%，**那么每月付款日期偿还的利率为：**

0.06/12=0.005 **或** 0.5%

- **iPer：此参数指出计算期间，介于** 1 **和 Nper 参数值之间。**
- **iNper：此参数包含财务操作的付款日期总数。**
- **dblPV：此参数包含将来要支付的一系列租金的实际值或总和。**
- **dblFV：此参数包含最后付款日期支付之后要获得的未来值或余额。通常情况下，尤其是偿还贷款时，此参数设为** "0"。**实际上，指定所有付款日期之后，贷款的值为零。**
- **iType：此参数表示付款的最终期限。其值可以为以下值之一：**
  - **0 如果拖欠付款（如期间结束）**
  - **1 如果提前付款（如期间开始）**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

备注

注:

- **Rate 和 Nper 参数必须以相同单位表示的租金计算。**
- **支付金额（尤其是以 Pmt 参数表示的金额）以负数表示。收到的总和以正数表示。**

PV()

此函数根据常量和定期的未来最终期限，以固定的利率返回年金的实际数额。

内部 Basic 语法

Function PV(dblRate As Double, iNper As Long, dblPmt As Double, dblFV As Double, iType As Long) As Double

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **dblRate：此参数表示付款日期的利率。例如，如果某项贷款的年利率为 6%，那么每月付款日期偿还的利率为：**

0.06/12=0.005 **或** 0.5%

- **iNper：此参数包含财务操作的付款日期总数。**
- **dblPmt：此参数指出每个付款日期要支付的数额。租金一般包含本金和利息。**
- **dblFV：此参数包含最后付款日期支付之后要获得的未来值或余额。通常情况下，尤其是偿还贷款时，此参数设为 "0"。实际上，指定所有付款日期之后，贷款的值为零。**
- **iType：此参数表示付款的最终期限。其值可以为以下值之一：**

- **0 如果拖欠付款（如期间结束）**
- **1 如果提前付款（如期间开始）**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

备注

注:

- **Rate 和 Nper 参数必须以相同单位表示的租金计算。**
- **支付金额（尤其是以 Pmt 参数表示的金额）以负数表示。收到的总和以正数表示。**

Randomize()

初始化随机数发生器。

内部 Basic 语法

Function Randomize(lValue As Long)

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **IValue：用于通过指定新的初始值来初始化 Rnd 函数的随机数发生器的可选参数。如果忽略此参数，将用系统时钟返回的值作为初始值。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

备注

另请参阅：

- Rnd() [**页** 429]

示例

```
Dim MyNumber
Randomize
MyNumber= Int((10*Rnd)+1) :'返回介于 1 和 10 之间的随机值。
RetVal=MyNumber
```

RATE()

此函数按年金付款日期返回利率。

内部 Basic 语法

Function RATE(iNper As Long, dblPmt As Double, dblFV As Double, dblPV As Double, iType As Long, dblGuess As Double) As Double

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **iNper：此参数包含财务操作的付款日期总数。**
- **dblPmt：此参数指出每个付款日期要支付的数额。租金一般包含本金和利息。**
- **dblFV：此参数包含最后付款日期支付之后要获得的未来值或余额。通常情况下，尤其是偿还贷款时，此参数设为** "0"。**实际上，指定所有付款日期之后，贷款的值为零。**
- **dblPV：此参数包含将来要支付的一系列租金的实际值或总和。**
- **iType：此参数表示付款的最终期限。其值可以为以下值之一：**
  - **0 如果拖欠付款（如期间结束）**
  - **1 如果提前付款（如期间开始）**
- **dblGuess：此参数包含按付款日期得出的利率的估计值。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

备注

注:

- **支付金额（尤其是以 Pmt 参数表示的金额）以负数表示。收到的总和以正数表示。**
- **此函数通过迭代以 Guess 参数中分配的值开始执行其计算。如果** 20 **次迭代之后未得到任何结果，表示该函数失败。**

RemoveRows()

在 strRowNames 参数标识的行列表中执行删除。

处理 "ListBox" 控制类型值时可用到此函数。此控制类型的值以数组表示，如下所示：

- "|" 字符用作列分隔符。
- "|" 字符用作行分隔符。
- 每一行都在 "=" 号右侧以唯一标识符结束。

内部 Basic 语法

Function RemoveRows(strList As String, strRowNames As String) As String

应用的字段

版本: 3.5

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strList：包含要处理的 "ListBox" 控制的源字符串。**
- **strRowNames：要删除的行的标识符。标识符以逗号分隔。**

输出参数

出现错误时，有两种可能性：

- 在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。
- 如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。

备注

另请参阅：

- SubList() [页 445]
- SetSubList() [页 435]
- ApplyNewVals() [页 345]

示例

```
Dim MyStr
MyStr=RemoveRows("a1|a2=a0,b1|b2=b0", "a0,c0") :'返回 "b1|b2=b0"
RetVal=MyStr
```

Replace()

对于出现的所有 strOldPattern 参数，都用 strNewPattern 参数替换，后者在 strData 参数包含的字符串内。搜索 strOldPattern 参数（通过 bCaseSensitive 参数）可设定为区分大小写。

内部 Basic 语法

Function Replace(strData As String, strOldPattern As String, strNewPattern As String, bCaseSensitive As Long) As String

应用的字段

版本: 3.5

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strData：包含要替换的参数的字符串。**
- **strOldPattern：要在 strData 参数包含的字符串中寻找的参数。**
- **strNewPattern：替换出现的每个参数的文本。**
- **bCaseSensitive：根据此参数确定搜索是区分大小写 (=1) 或不区分大小写 (=0)。默认情况下，此参数设置为 1。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**

- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim MyStr
MyStr=Replace("youmeyoumeyou", "you", "me",0) :'返回 "mememememe"
MyStr=Replace("youmeyoumeyou", "You", "me",1) :'返回 "youmeyoumeyou"
MyStr=Replace("youmeYoumeyou", "You", "me",1) :'返回 "youmememeyou"
```

Right()

返回字符串参数最右边的 iNumber **字符。**

内部 Basic 语法

Function Right(strString As String, lNumber As Long) As String

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strString：要处理的字符串。**
- **lNumber：要返回的字符数量。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**

- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim lWord, strMsg, rWord, iPos :' 声明变量。
strMsg = "Left() Test."
iPos = InStr(1, strMsg, " ") :' 查找空格。
lWord = Left(strMsg, iPos - 1) :' 获取左边的词。
rWord = Right(strMsg, Len(strMsg) - iPos) :' 获取右边的词。
strMsg=rWord+lWord :' 然后互换
RetVal=strMsg
```

RightPart()

将字符串的某部分提取到 strSep 参数中指定的分隔符的右侧。
搜索分隔符时会从右向左进行。
使用 bCaseSensitive 参数搜索可以设置为区分大小写。

内部 Basic 语法

Function RightPart(strFrom As String, strSep As String, bCaseSensitive As Long) As String

应用的字段

版本: 3.5

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strFrom：要处理的源字符串。**
- **strSep：在源字符串中用作分隔符的字符。**

- **bCaseSensitive：根据此参数确定搜索是区分大小写**(=1)**或不区分大小写**(=0)**。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

这些示例说明了 LeftPart、LeftPartFromRight、RightPart 和 RightPartFromLeft 函数的用法，这些函数同属一个字符串："This_is_a_test":

```
LeftPart("This_is_a_test","_",0)
```

返回 "This"。

```
LeftPartFromRight("This_is_a_test","_",0)
```

返回 "This_is_a"。

```
RightPart("This_is_a_test","_",0)
```

返回 "test"。

```
RightPartFromLeft("This_is_a_test","_",0)
```

返回 "is_a_test"。

RightPartFromLeft()

将字符串的某部分提取到 strSep 参数中指定的分隔符的右侧。
搜索分隔符时会从左向右进行。
使用 bCaseSensitive 参数搜索可以设置为区分大小写。

内部 Basic 语法

Function RightPartFromLeft(strFrom As String, strSep As String, bCaseSensitive As Long) As String

应用的字段

版本: 3.5

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strFrom：要处理的源字符串。**
- **strSep：在源字符串中用作分隔符的字符。**
- **bCaseSensitive：根据此参数确定搜索是区分大小写** (=1) **或不区分大小写** (=0)**。默认情况下，此参数设置为** 1**。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

这些示例说明了 LeftPart、LeftPartFromRight、RightPart 和 RightPartFromLeft 函数的用法，这些函数同属一个字符串："This_is_a_test":

```
LeftPart("This_is_a_test","_",0)
```

返回 "This"。

```
LeftPartFromRight("This_is_a_test","_",0)
```

返回 "This_is_a"。

```
RightPart("This_is_a_test","_",0)
```

返回 "test"。

```
RightPartFromLeft("This_is_a_test","_",0)
```

返回 "is_a_test"。

RmAllInDir()

此函数删除文件夹中的所有项（文件和文件夹）。该文件夹本身不会删除。

内部 Basic 语法

Function RmAllInDir(strRmDirectory As String, bStopIfError As Long) As Long

应用的字段

版本: 3.4.0

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strRmDirectory：此参数包含操作相关文件夹的完整路径。**
- **bStopIfError：如果此参数设为 1，无法删除文件或文件夹时，删除操作会中止。如果此参数设为 0，操作会继续，并移动到随后的文件或文件。**

输出参数

- **0：正常执行。**
- **非 0 值：错误代码。**

示例

```
RetVal = RmAllInDir("c:\files\test", 1)
```

RmDir()

移除现有目录。

内部 Basic 语法

Function RmDir(strRmDirectory As String) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strRmDirectory：要移除的目录的完整路径。**

输出参数

- 0：**正常执行。**
- **非** 0 **值：错误代码。**

备注

注:

要删除的目录必须为空。否则，该函数不起作用。

示例

```
RetVal = RmDir("c: mp")
```

Rnd()

返回包含随机数的值。

内部 Basic 语法

Function Rnd(dValue As Double) As Double

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **dValue：可选参数，其值定义函数的执行模式：**
  - **小于零：每次都生成相同的数值。**
  - **大于零：序列中下一个随机数。**
  - **等于零：生成的最后一个随机数。**
  - **忽略：序列中下一个随机数。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

备注

注:

调用此函数前，必须使用无参数的 Randomize 函数初始化随机数生成器。

另请参阅：

- Randomize() [**页** 419]

示例

```
Dim MyNumber
Randomize
MyNumber= Int((10*Rnd)+1) :'返回介于 1 和 10 之间的随机值。
RetVal=MyNumber
```

RoundValue()

此函数根据 iDigits 参数指定的设置，计算数值舍入到小数点后位数的舍入值。

内部 Basic 语法

Function RoundValue(dValue As Double, iDigits As Long) As Double

应用的字段

版本: 3.4.0

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **dValue：此参数包含要舍入的数值。**
- **iDigits：此参数包含保留舍入运算的小数位数。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

例如：

```
RetVal = RoundValue(1.2568, 2)
```

返回值：

```
1.26
```

例如：

```
RetVal = RoundValue(1.2568, 0)
```

返回值：

```
1
```

RTrim()

移除字符串中的所有结尾空格。

内部 Basic 语法

Function RTrim(strString As String) As String

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strString：要处理的字符串。**

输出参数

出现错误时，有两种可能性：

- 在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。
- 如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。

示例

```
' 此示例分别用 LTrim 和 RTrim 函数去除字符串变量中前导的 ' 和结尾空格。
' 单独使用 Trim 函数去除两种类型的空格。
' 此示例还将说明 LCase 和 UCase，同时说明的还有
' 嵌套函数调用的用法

Dim strString as String
Dim strTrimString as String
strString = "  <-Trim->  " :' 初始化字符串。
strTrimString = LTrim(strString) :' strTrimString = "<-Trim->  ".
strTrimString = LCase(RTrim(strString)) :' strTrimString = "  <-trim->".
strTrimString = LTrim(RTrim(strString)) :' strTrimString = "<-Trim->".
' 单独使用 Trim 函数达到同样的效果。
strTrimString = UCase(Trim(strString)) :' strTrimString = "<-TRIM->".
RetVal= "|" & strTrimString & "|"
```

Second()

返回以 tmTime 参数表示的时间中包含的秒数。

内部 Basic 语法

Function Second(tmTime As Date) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输入参数

- **tmTime：要处理的"日期+时间"格式的参数。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim strSecond
strSecond=Second(Date())
RetVal=strSecond :'返回当前小时内过去的秒数，例如，"30" 秒，如果时间为 15:45:30 的话
```

SetMaxInst()

可通过此函数设置 Basic **脚本可以执行的最大指令数。默认情况下，指令数限制为** 10000。

API 语法

long SetMaxInst(long lMaxInst);

内部 Basic 语法

Function SetMaxInst(lMaxInst As Long) As Long

应用的字段

版本: 4.3.0

	可用
AssetCenter API	✓
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **IMaxInst：此参数包含可由脚本执行的最大指令数。**

输出参数

- 0：**正常执行。**
- **非** 0 **值：错误代码。**

备注

注:

如果将 IMaxInst 参数设为 "0"，**脚本可以执行的指令数将不受限制。**

SetSubList()

定义 "ListBox" **控制的子列表值。**

内部 Basic 语法

Function SetSubList(strValues As String, strRows As String, strRowFormat As String) As String

应用的字段

版本: 3.5

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓

	可用
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strValues：包含要处理的** "ListBox" **控制的源字符串。**
- **strRows：要添加到 strValues 参数中或替换该参数中字符串所包含的字符的值列表。该值以** "|" **字符分隔。处理的行用其标识符进行标识，位于** "=" **号的右侧。不处理未知行。**
- **strRowFormat：子列表的格式化指令。该指令以** "|" **字符分隔。此参数具有以下特点：**
  - "1" **表示信息包含子列表的第一列中。**
  - "i-j" **可用于定义列组。**
  - "-" **表示所有列。**
  - **如果列未知，则不会返回任何值。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim MyStr
MyStr=SetSubList("a1|a2|a3=a0,b1|b2|b3=b0,c1|c2|c3=c0", "A2|A1=a0, B2|B1=b0", "2|1
") :'返回 "A1|A2|a3=a0,B1|B2|b3=b0,c1|c2|c3=c0"
MyStr=SetSubList("a1|a2|a3=a0,b1|b2|b3=b0,c1|c2|c3=c0", "Z2=*,B2=b0", "2") :'返回 "a
1|Z2|a3=a0,b1|B2|b3=b0,c1|Z2|c3=c0"
MyStr=SetSubList("a1|a2|a3=a0,b1|b2|b3=b0,c1|c2|c3=c0", "B5|B6|B7=b0,C5|C6,C7=c0
", "5-7") :'返回 "a1|a2|a3=a0,b1|b2|b3||B5|B6|B7=b0,c1|c2|c3||C5|C6|C7=c0"
MyStr=SetSubList("a1|a2|a3=a0,b1|b2|b3=b0,c1|c2|c3=c0", "B1|B2|B3|B4=b0", "-") :'返回
"a1|a2|a3=a0,B1|B2|B3|B4=b0,c1|c2|c3=c0"
MyStr=SetSubList("A|B|C,D|E|F", "X=*", "2") :'返回 "A|X|C,D|X|F"
RetVal=""
```

Sgn()

返回表示数值符号的值。

内部 Basic 语法

Function Sgn(dValue As Double) As Double

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **dValue：想知道其符号的数值。**

输出参数

此函数返回以下值之一：

- 1: **大于零的数值。**
- 0: **等于零的数值。**
- -1: **小于零的数值。**

示例

```
Dim dNumber as Double
dNumber = -256
RetVal=Sgn(dNumber)
```

Shell()

启动可执行程序。

内部 Basic 语法

Function Shell(strExec As String, bShowWindow As Long, bBackground As Long) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strExec：要启动的可执行程序的完整路径。**
- **bShowWindow：如果此参数设为 1（默认值），则启动程序时显示命令框。如果此参数设为 0，则不显示命令框。**
- **bBackground：如果此参数设为 1（默认值），则函数在赋予后退控制（同步执行）之前会等待程序执行结束。如果此参数设为 0，则不同步执行程序。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim MyId
MyId=Shell("C:\WinNT\explorer.exe")
RetVal=""
```

Sin()

返回以弧度表示的值的正弦。

内部 Basic 语法

Function Sin(dValue As Double) As Double

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **dValue：想知道其正弦的数值。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

备注

注:

角度转换为弧度的转换公式如下：

弧度 = **角度** * Pi / 180

示例

```
Dim dCalc as Double
dCalc=Sin(2.79)
RetVal=dCalc
```

Space()

创建字符串，该字符串中包含 iSpace 参数指明的空格数。

内部 Basic 语法

Function Space(iCount As Long) As String

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **iCount：要在字符串中插入的空格数。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

备注

注:

此函数可以用于格式化字符串或删除固定长度字符串中的日期。

示例

```
Dim MyString
' 返回带有 10 个空格的字符串。
MyString = Space(10)
' 在两个字符串之间插入 10 个空格。
MyString = "Space" & Space(10) &  "inserted"
RetVal=MyString
```

Sqr()

返回数值的平方根。

内部 Basic 语法

Function Sqr(dValue As Double) As Double

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **dValue：想知道其平方根的数值。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim dCalc as Double
dCalc=Sqr(81)
RetVal=dCalc
```

Str()

将数值转换为字符串。

内部 Basic 语法

Function Str(strValue As String) As String

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strValue：要转换为字符串的数值。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim dNumber as Double
dNumber=Cos(2.79)
RetVal=Str(dCalc)
```

StrComp()

比较两个字符串。

内部 Basic 语法

Function StrComp(strString1 As String, strString2 As String, iOptionCompare As Long) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strString1：第一个字符串。**
- **strString2：第二个字符串。**
- **iOptionCompare：比较类型。此参数可设为** "0"，**表示二进制比较，或设为** "1"，**表示文本比较。**

输出参数

- -1: **strString1 大于 strString2。**
- 0: **strString1 等于 strString2。**
- 1: **strString1 小于 strString2。**

String()

字符串返回由 strString 字符组成的字符串，反复重复该字符 iCount 次。

内部 Basic 语法

Function String(iCount As Long, strString As String) As String

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **iCount：字符 strString 的出现次数。**
- **strString：用于构成字符串的字符。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim iCount as Integer
Dim strTest as String
strTest="T"
iCount=5
RetVal=String(iCount,strTest)
```

SubList()

返回表示 "ListBox" 控制值的字符串中所包含的值列表的子列表。

内部 Basic 语法

Function SubList(strValues As String, strRows As String, strRowFormat As String) As String

应用的字段

版本: 3.5

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strValues：包含要处理的 "ListBox" 控制的源字符串。**
- **strRows：子列表中要包括的行标识符。标识符以逗号分隔。可使用某些通配符：**
  - **"*" 包括子列表中的所有标识符。**
  - **未知标识符返回子列表的空值。**
- **strRowFormat：子列表的格式化指令。该指令以 "|" 字符分隔。此参数具有以下特点：**
  - **"1" 表示执行的子列表中列表第一列中的信息。**
  - **"0" 表示执行的子列表中列表行的标识符。**

- "*" 表示所有列中包含的信息（行标识符除外）。
- 如果列未知，则不会返回任何值。

输出参数

出现错误时，有两种可能性：

- 在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。
- 如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。

示例

```
Dim MyStr
MyStr=SubList("a1|a2|a3=a0,b1|b2|b3=b0,c1|c2|c3=c0", "a0,b0,a0", "3|2|3") :'返回 "a3|a2|a3,b3|b2|b3,a3|a2|a3"
MyStr=SubList("a1|a2|a3=a0,b1|b2|b3=b0,c1|c2|c3=c0", "*", "*|0") :'返回 "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") :'返回 "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", "*", "999=0") :'返回 "=a0,=b0,=c0"
MyStr=SubList("a1|a2|a3=a0,b1|b2|b3=b0,c1|c2|c3=c0", "z0", "*=0") :'返回 ""
MyStr=SubList("a1|a2|a3=a0,b1|b2|b3=b0,c1|c2|c3=c0", "*", "=1") :'返回 "=a1,=b1,=c1"
MyStr=SubList("A|B|C,D|E|F", "*", "2=0") :'返回 "B,E"
RetVal=""
```

SysEnumToComboBox()

此函数重组系统逐项列表项以使其格式能够与向导控制列表兼容。这样就可以在向导的下拉列表中显示系统逐项列表项的值。

内部 Basic 语法

Function SysEnumToComboBox(strFormat As String) As String

应用的字段

版本: 4.3.0

	可用
AssetCenter API	
字段或链接的配置脚本	
"Script" 类型操作	
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strFormat：此参数包含系统逐项列表的条目列表。如果此参数包含 AmGetFieldFormat() 函数的执行结果更好。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

下例将获取 amWorkOrder **表中的系统逐项列表** seStatus **的值并按照与向导控制列表兼容的格式对其进行重组：**

```
Dim strFormat As String
strFormat = AmGetFieldFormat(AmGetFieldFromName(AmGetTableFromName("amWorkOrder"), "seStatus"))
RetVal = SysEnumToComboBox(strFormat)
```

Tan()

返回以弧度表示的值的正切。

内部 Basic 语法

Function Tan(dValue As Double) As Double

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **dValue：想知道其正切的数值。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

备注

注:

角度转换为弧度的转换公式如下：

```
弧度 = 角度 * Pi / 180
```

示例

```
Dim dCalc as Double
dCalc=Tan(2.79)
RetVal=dCalc
```

Time()

返回当前时间。

内部 Basic 语法

Function Time() As Date

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
RetVal = Time()
```

Timer()

返回自 12:00 AM **开始过去的秒数。**

内部 Basic 语法

Function Timer() As Double

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓

	可用
向导的 FINISH.DO 脚本	✓

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
RetVal = Timer()
```

TimeSerial()

此函数根据 iHour、iMimute 和 iSecond 参数返回格式化时间。

内部 Basic 语法

Function TimeSerial(iHour As Long, iMinute As Long, iSecond As Long) As Date

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **iHour：小时。**
- **iMinute：分钟。**
- **iSecond：秒。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

这些参数中的每一个都可以设置为表示小时、分钟或秒的数值表达式。因此，例如：

```
TimeSerial(12-8, -10, 0)
```

返回值：

```
3:50:00
```

当参数值超出预期范围（例如，0至59分钟，0至23秒等）时，会转换为下一个参数。因此，如果输入 "75" **表示 iMinute 参数，将解释为** 1 **小时** 15 **分钟。**

例如：

```
TimeSerial (16, 50, 45)
```

返回值：

```
16:50:45
```

TimeValue()

此函数返回"日期+时间"值的时间部分。

内部 Basic 语法

Function TimeValue(tmTime As Date) As Date

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓

	可用
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **tmTime："日期+时间" 格式日期。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

例如：

```
TimeValue ("1999/09/24 15:00:00")
```

返回值：

```
15:00:00
```

ToSmart()

此函数通过将每个词的第一个字母转换为大写来重新格式化源字符串。

内部 Basic 语法

Function ToSmart(strString As String) As String

应用的字段

版本: 3.5

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓

	可用
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strString：要重新格式化的源字符串。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

例如：

```
RetVal = ToSmart ("hello world")
```

返回值：

```
Hello World
```

Trim()

返回已移除前导和结尾空格的字符串。

内部 Basic 语法

Function Trim(strString As String) As String

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓

	可用
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strString：要处理的字符串。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
' 此示例分别用 LTrim 和 RTrim 函数去除字符串变量中前导的 ' 和结尾空格。
' 单独使用 Trim 函数去除两种类型的空格。
' 此示例还将说明 LCase 和 UCase，同时说明的还有
' 嵌套函数调用的用法

Dim strString as String
Dim strTrimString as String
strString = "  <-Trim->  " :' 初始化字符串。
strTrimString = LTrim(strString) :' strTrimString = "<-Trim->  ".
strTrimString = LCase(RTrim(strString)) :' strTrimString = "  <-trim->".
strTrimString = LTrim(RTrim(strString)) :' strTrimString = "<-Trim->".
' 单独使用 Trim 函数达到同样的效果。
strTrimString = UCase(Trim(strString)) :' strTrimString = "<-TRIM->".
RetVal= "|" & strTrimString & "|"
```

UCase()

返回所有小写字符已转换为大写字符的字符串。

内部 Basic 语法

Function UCase(strString As String) As String

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strString：要转换为大写字母的字符串。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
' 此示例分别用 LTrim 和 RTrim 函数去除字符串变量中前导的 ' 和结尾空格。
' 单独使用 Trim 函数去除两种类型的空格。
' 此示例还将说明 LCase 和 UCase，同时说明的还有
' 嵌套函数调用的用法

Dim strString as String
Dim strTrimString as String
strString = "  <-Trim->  " :' 初始化字符串。
strTrimString = LTrim(strString) :' strTrimString = "<-Trim->  ".
strTrimString = LCase(RTrim(strString)) :' strTrimString = "  <-trim->".
strTrimString = LTrim(RTrim(strString)) :' strTrimString = "<-Trim->".
' 单独使用 Trim 函数达到同样的效果。
strTrimString = UCase(Trim(strString)) :' strTrimString = "<-TRIM->".
RetVal= "|" & strTrimString & "|"
```

UnEscapeSeparators()

删除字符串中所有的转义字符。

内部 Basic 语法

Function UnEscapeSeparators(strSource As String, strEscChar As String) As String

应用的字段

版本: 3.5

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strSource：要处理的字符串。**
- **strEscChar：要删除的转义字符。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页** 281] **函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim MyStr
MyStr=UnEscapeSeparators("你\|我\|你\|", "\") :'返回 "你|我|你|"
RetVal=""
```

Union()

合并用两个分隔符分隔的字符串。删除重复部分。

内部 Basic 语法

Function Union(strListOne As String, strListTwo As String, strSeparator As String, strEscChar As String) As String

应用的字段

版本: 3.5

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strListOne：第一个字符串。**
- **strListTwo：第二个字符串。**
- **strSeparator：用于划分字符串中包含的元素的分隔符。**
- **strEscChar：转义字符。如果此字符前面有分隔符，则将被忽略。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim MyStr
MyStr=Union("a1|a2,b1|b2", "a1|a3,b1|b2", ",", "\") :'返回 "a1|a2,b1|b2,a1|a3"
```

```
MyStr=Union("a1|a2,b1|b2", "a1|a3\,b1|b2", ",", "\") :'返回 "a1|a2,b1|b2,a1|a3\,b1|b2"
RetVal=""
```

UTCToLocalDate()

此函数将 UTC 格式（不受时区限制)日期转换为"日期+时间"格式日期。

内部 Basic 语法

Function UTCToLocalDate(tmUTC As Date) As Date

应用的字段

版本: 3.5

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **tmUTC：UTC 格式的日期。**

输出参数

出现错误时，有两种可能性：

- **在 AssetCenter 中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用 AmLastError() [页 280] 函数（和可选的 AmLastErrorMsg() [页 281] 函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
RetVal = UTCToLocaldate([DateTime])
```

Val()

将表示数值的字符串转换为双精度字符串。

内部 Basic 语法

Function Val(strString As String) As Double

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **strString：要转换的字符串。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim strYear
Dim dYear as Double
strYear=Year(Date())
dYear=Val(strYear)
RetVal=dYear :'返回当前年
```

WeekDay()

返回 tmDate 参数表示的日期中包含的周日期。

内部 Basic 语法

Function WeekDay(tmDate As Date) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **tmDate：要处理的"日期+时间"格式的参数。**

输出参数

返回的数值与周日期相对应，"1"**表示星期日，**"2"**表示星期二，**"7"**表示星期六等。**

示例

```
Dim strWeekDay
strWeekDay=WeekDay(Date())
RetVal=strWeekDay :'返回星期日期
```

Year()

返回 tmDate 参数表示的值中包含的年份。

内部 Basic 语法

Function Year(tmDate As Date) As Long

应用的字段

版本: 3.00

	可用
AssetCenter API	
字段或链接的配置脚本	✓
"Script" 类型操作	✓
向导脚本	✓
向导的 FINISH.DO 脚本	✓

输入参数

- **tmDate：要处理的"日期+时间"格式的参数。**

输出参数

出现错误时，有两种可能性：

- **在** AssetCenter **中挂起包含此函数的脚本并向用户发出错误消息。**
- **如果从外部程序调用，必须调用** AmLastError() [**页** 280] **函数（和可选的** AmLastErrorMsg() [**页**281]**函数）以查找是否发生错误（并获取与其相关的消息）。**

示例

```
Dim strYear
strYear=Year(Date())
RetVal=strYear :'返回当前年
```

IV 索引

可用功能 - 函数完全列表

- Abs
- AmActionDde
- AmActionExec
- AmActionMail
- AmActionPrint
- AmActionPrintPreview
- AmActionPrintTo
- AmAddAllPOLinesToInv
- AmAddCatRefAndCompositionToPOrder
- AmAddCatRefToPOrder
- AmAddEstimLinesToPO
- AmAddEstimLineToPO
- AmAddLicContentToRequest
- AmAddPOLineToInv
- AmAddPOrderLineToReceipt
- AmAddReceiptLineToInvoice
- AmAddReqLinesToEstim
- AmAddReqLinesToPO
- AmAddReqLineToEstim
- AmAddReqLineToPO
- AmAddRequestLineToPOrder
- AmAddTemplateToPOrder
- AmAddTemplateToRequest
- AmArchiveRecord
- AmAttribCmdAvailability
- AmBackupRecord
- AmBuildNumber
- AmBusinessSecondsInDay
- AmCalcConsolidatedFeature
- AmCalcDepr
- AmCalculateCatRefQty
- AmCalculateReqLineQty
- AmCbkReplayEvent
- AmCheckTraceDone
- AmCleanup
- AmClearLastError
- AmCloseAllChildren
- AmCloseConnection
- AmCommit
- AmComputeAllLicAndInstallCounts
- AmComputeLicAndInstallCounts
- AmConnectionName
- AmConnectTrace
- AmConvertCurrency

- 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

可用功能 - 正在传输数据 - 正在计算

- 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

可用功能 - 正在获取信息

- 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

可用功能 - 正在 AssetCenter 中触发内部操作

- 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

可用功能 - “财务”模块

- AmCalcDepr
- AmCbkReplayEvent
- AmConvertCurrency
- AmDefaultCurrency
- AmFormatCurrency
- AmTaxRate

可用功能 - 正在更改数据库中的数据

- 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

可用功能 - “采购”模块

- 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**

可用功能 - “合同”模块

- AmUpdateLossLines

可用功能 - “电缆”模块

- 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

可用功能 - “软件分发”模块

- AmESDAddComputers
- AmESDCreateTask

可用功能 - “资产组合”模块

- AmStandIn
- AmStandInGroup

可用功能 - 正在从 AssetCenter 触发外部操作

- AmActionDde
- AmActionExec
- AmActionMail
- AmActionPrint
- AmActionPrintPreview
- AmActionPrintTo