Peregrine

AssetCenter 4.3

Référence de programmation

PART NUMBER DAC-432-FR09

Pour toute information technique sur ce produit ou pour faire la demande d'une documentation sur un produit dont vous possédez la licence, veuillez contacter le support client Peregrine Systems, Inc. en envoyant un e-mail à l'adresse suivante : support@peregrine.com.

Pour tout commentaire ou suggestion à propos du présent document, veuillez contacter le département des publications techniques de Peregrine Systems, Inc. en envoyant un e-mail à l'adresse suivante : doc_comments@peregrine.com.

Cette édition s'applique à la version 4.3 du programme sous contrat de licence

AssetCenter

Peregrine Systems, Inc.
Worldwide Corporate Campus and Executive Briefing Center
3611 Valley Centre Drive San Diego, CA 92130
Tel 800.638.5231 or 858.481.5000
Fax 858.481.1751
www.peregrine.com

Référence de programmation

Table des matières

PARTIE I

Introduction

CHAPITRE 1

Bases essentielles de programmation

Ce chapitre présente les composants fondamentaux du langage Basic disponible sous AssetCenter. Si vous possédez des bases de programmation et que vous avez déjà pratiqué d'autres langages, la très grande majorité de ce chapitre vous sera familière. Nous vous invitons néanmoins à parcourir rapidement les différentes sections proposées, certaines fonctionnalités classiques étant volontairement limitées ou absentes du Basic de AssetCenter.

Introduction aux variables

Les variables sont utilisées pour stocker des données au cours de l'exécution d'un programme. Elles sont identifiées par :

- leur nom, utilisé pour référencer la valeur contenue par la variable.
- leur type, qui détermine quelles données peuvent être stockées dans la variable.

On distingue en général deux types de variables :

- Les tableaux,
- Les variables scalaires qui regroupent toutes les variables qui ne sont pas des tableaux.

Déclarer une variable

Toute variable doit être explicitement déclarée avant d'être utilisée. La syntaxe de déclaration est la suivante :

```
Dim <Nom de la variable> [As <Type de la variable>]
```

Note :

La déclaration explicite des variables dans le Basic AssetCenter correspond à l'utilisation du mot clé **Option Explicit** de Microsoft Visual Basic.

Le nom d'une variable doit respecter les contraintes suivantes :

- Il doit commencer par une lettre majuscule ou minuscule,
- Il ne doit pas comporter plus de 40 caractères,
- Il peut contenir les lettres de A à Z et de a à z, les chiffres de 0 à 9, ainsi que le caractère underscore ("_").

  Note :

  Les caractères accentués sont autorisés mais leur utilisation est fortement déconseillée.

- Il ne peut être un mot clé réservé. Par exemple, tous les noms de fonctions Basic ou les clauses sont des mots clés réservés.

La clause optionnelle **As** permet de définir le type de la variable déclarée. Le type précise le type d'information stocké dans la variable. Les types disponibles sont par exemple : **String**, **Integer**, **Variant**, ...

Si la clause **As** est omise, la variable est considérée comme étant de type **Variant**.

Déclaration simple

Dans le cas d'une déclaration simple, chaque ligne déclarative concerne une seule variable, comme dans l'exemple suivant :

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

Déclaration combinée

Dans le cas d'une déclaration combinée, chaque ligne déclarative peut concerner un nombre quelconque de variables, comme dans l'exemple suivant :

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

Note :

Comme il a été décrit précédemment, toute variable dont le type n'est pas précisé est considérée comme étant de type **Variant**. Ainsi, dans la deuxième ligne de l'exemple précédent, les variables **A** et **B** sont de type **Variant** et la variable **C** de type **Integer**.

Types des données

Le tableau ci-dessous récapitule les différents types possibles pour une fonction ou un paramètre :

Type	Signification
Integer	Nombre entier de -32 768 à +32 767.
Long	Nombre entier de -2 147 483 647 à +2 147 483 646.
Single	Nombre à virgule flottante de 4 octets (simple précision).
Double	Nombre à virgule flottante de 8 octets (double précision).
String	Texte pour lequel tous les caractères sont acceptés.
Date	Date ou Date+Heure.
Variant	Type générique pouvant représenter n'importe quel type.

Note :

Ces types ne sont pas tous disponibles à partir d'un outil externe. Seuls les types **Long**, **Double** et **String** sont disponibles. Le **Variant** n'existe pas et les objets de types **Integer** et **Date** sont représentés par un **Long**.

Les types numériques

Le Basic disponible sous AssetCenter propose plusieurs types de données numériques : Integer, Long, Single et Double. L'utilisation d'un type de données numérique occupe généralement moins de place en mémoire qu'un **Variant**.

Si vous avez la certitude qu'une variable stockera systématiquement des nombres entiers (par exemple 123) et non des nombres fractionnels (comme 3.14), il est préférable de la déclarer comme un **Integer** ou un **Long**. Les opérations effectuées sur ces types sont plus rapides et moins coûteuses en mémoire que pour les autres types de données. Ces types de donnée sont particulièrement adaptés pour les compteurs utilisés dans les boucles.

Si une variable doit contenir un nombre fractionnel, déclarez-la comme **Single** ou **Double**.

Note :

Les nombres à virgule flottante (**Single** ou **Double**) peuvent être sujets à des erreurs d'arrondis.

Le type String

Si vous avez la certitude qu'une variable stockera systématiquement une chaîne de caractères, déclarez-la comme **String** :

```
Dim MyString As String
```

Vous pouvez alors stocker des chaînes de caractères dans cette variable et manipuler son contenu en utilisant les fonctions dédiées au traitement des chaînes de caractères :

```
MyString = "Ceci est une chaîne"
MyString = Right(MyString,6)
```

Par défaut, une variable de type **String** possède une taille variable. L'espace réservé au stockage des chaînes de caractères augmente ou diminue en fonction de la taille des données affectées à la variable. Il est toutefois possible de déclarer une variable de type **String** dont la taille est fixe, en utilisant la syntaxe suivante :

```
Dim <Nom de la variable> As String * <Taille de la chaîne stockée>
```

L'exemple suivant déclare une variable qui contiendra 20 caractères :

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

Si vous stockez dans cette variable une chaîne de moins de 20 caractères, des espaces seront rajoutés en fin de chaîne jusqu'à concurrence de la taille prévue. A l'inverse, si vous stockez une chaîne de plus de 20 caractères, la chaîne sera tronquée à partir du vingtième et unième caractère.

Le type Variant

Le **Variant** est un type générique qui peut se substituer à tous les autres types. Vous n'avez pas à vous soucier d'éventuels problèmes de conversion entre les différents types de données qu'un **Variant** peut représenter. La conversion est réalisée automatiquement, comme dans l'exemple suivant :

```
Dim MyVariant As Variant
MyVariant = "123"
MyVariant = MyVariant - 23
MyVariant = "Les " & MyVariant & " premiers"
```

Bien que la conversion entre les différents types soit automatique, veillez à respecter les règles suivantes :

- Si vous réalisez des opérations arithmétiques sur un **Variant**, celui-ci doit impérativement contenir un nombre, même si celui-ci est représenté par une chaîne de caractères.
- Si une opération de concaténation de chaînes fait intervenir un **Variant**, utilisez l'opérateur **&** de préférence à l'opérateur +.

Un **Variant** peut également contenir deux valeurs particulières : la valeur vide et la valeur **Null**.

La valeur vide

Avant qu'une valeur soit affectée pour la première fois à une variable de type **Variant**, celle-ci contient la valeur vide. Cette valeur est une valeur particulière, différente de 0, d'une chaîne vide ou encore de la valeur **Null**. Pour tester si un **Variant** contient la valeur vide, utilisez la fonction Basic **IsEmpty**(), comme l'illustre l'exemple suivant :

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

Un **Variant** contenant la valeur vide peut être utilisé dans les expressions. Suivant les cas, il sera traîté soit comme de valeur 0, soit comme contenant une

chaîne vide. Pour réaffecter la valeur vide à un **Variant**, utilisez le mot clé **Empty**, comme dans l'exemple suivant :

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

La valeur Null

La valeur **Null** est communément utilisée dans les bases de données pour indiquer une valeur inconnue ou manquante. Cette valeur possède des caractéristiques particulières :

- Les expressions faisant intervenir la valeur **Null** renvoient toujours la valeur **Null**. On parle de propagation de la valeur **Null** dans les expressions. Si une partie de l'expression vaut **Null** alors l'intégralité de l'expression vaut **Null**.
- En règle générale, si un paramètre de fonction a la valeur **Null**, la fonction renvoie la valeur **Null**.

Tableaux de données

Un tableau permet de stocker et de référencer un ensemble de variables par un nom unique et d'utiliser un nombre (un index) pour les identifier de façon unique. Tous les éléments d'un tableau partagent obligatoirement le même type de données. Vous ne pouvez pas créer un tableau contenant à la fois des variables de type **String** et **Double**. Evidemment, cette limitation peut être levée en utilisant des variables de type **Variant**.

Déclaration d'un tableau

Un tableau est un ensemble de variables.

Par convention, les notions suivantes sont présentées comme suit :

- Limite inférieure du tableau : index du premier élément.

  Note :

  Par défaut la limite inférieure d'un tableau est 0.

- Limite supérieure du tableau : index du dernier élément.

Note :

La limite supérieure d'un tableau ne peut excéder la taille d'un **Long** (2 147 483 646 éléments).

La déclaration d'un tableau est similaire à celle d'une variable :

```
Dim <Nom du tableau>(<Limite supérieure du tableau>) [As <Type des var
iables contenues dans le tableau>]
```

Exemples :

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

Vous pouvez également préciser la limite inférieure du tableau en utilisant la déclaration suivante :

```
Dim <Nom du tableau>(<Limite inférieure du tableau> To <Limite supérie
ure du tableau>) [As <Type des variables contenues dans le tableau>]
```

Exemples :

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

Limitations

Les limitations suivantes s'appliquent à la gestion des tableaux dans le Basic AssetCenter :

- Les tableaux de taille variable ne sont pas supportés. En particulier, il est impossible de redimensionner un tableau à la voléee.
- Les tableaux à plusieurs dimensions ne sont pas supportés.

Structures de contrôle

Comme leur nom l'indique, les structures de contrôle permettent de contrôler l'exécution d'un programme. Elles sont de deux types :

- Les structures de décision : redirigent et orientent l'exécution d'un programme en fonction de la réalisation de certains critères,
- Les structures de boucles : permettent de répéter l'exécution d'un ensemble d'instructions en fonction de certains critères.

Structures de décision

Une structure de décision réalise l'exécution conditionnelle d'instructions en fonction du résultat d'un test. Les structures de décision disponibles sont les suivantes :

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

If...Then

Utilisez cette structure pour exécuter conditionnellement une ou plusieurs instructions. La syntaxe de cette structure peut être mono ou multiligne, la syntaxe monoligne ne permettant l'exécution que d'une seule instruction :

```
If <Condition> Then <Instruction>
```

```
If <Condition> Then
  <Instructions>
End If
```

La condition est généralement une comparaison, mais toute expression dont le résultat est une valeur numérique peut être utilisée. Cette valeur est alors interprétée comme étant **True** (Vraie) ou **False** (Fausse) par le Basic. **False** correspond à la valeur numérique 0, toute autre valeur étant considérée comme **True**.

Si la condition est évaluée comme **True**, la ou les instructions suivant le mot clé **Then** seront exécutées.

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

Utilisez cette structure pour définir plusieurs blocs d'instructions conditionnelles. Un seul au plus de ces blocs est exécuté (le premier évalué comme **True**).

```
If <Condition1> Then
  <Instructions1>
ElseIf <Condition2> Then
  <Instructions2>
...
Else
  <InstructionsN>
End If
```

La première condition est testée, si le résultat est évalué comme **False**, la deuxième condition est testée et ainsi de suite jusqu'à ce que l'une d'entre elles soit évaluée comme **True**. Le jeu d'instructions situé après le mot clé **Then** de cette condition est alors exécuté.

Le mot clé **Else** est optionnel. Il permet de définir un jeu d'instructions à exécuter si toutes les conditions sont évaluées comme **False**.

Note :

Vous pouvez imbriquer autant de **ElseIf** que vous le souhaitez dans la structure de décision. Néanmoins, si vous comparez systématiquement la même expression à une valeur différente, la syntaxe de la structure de décision peut devenir inutilement complexe et difficile à lire. Nous vous conseillons, dans ce cas, d'utiliser de préférence un structure de décision de type **Select...Case**.

Select...Case

La fonctionnalité de cette structure est identique à celle des structures de décision précédentes, mais le code produit est en général plus lisible. Une structure **Select...Case** effectue un test unique en début de structure et compare le résultat du test aux valeurs de chaque mot clé **Case** dans la structure. S'il y a correspondance, le jeu d'instructions associé au mot clé **Case** est exécuté.

```
Select Case <Test>
[Case <Liste de valeurs 1>
<Instructions1>]
[Case <Liste de valeurs 2>
<Instructions2>]
...
[Case Else
<Instructionsn>]
End Select
```

Chaque liste de valeurs contient une ou plusieurs valeurs, séparées par des virgules. Si plusieurs mots clés **Case** déclarent une ou plusieurs valeurs correspondant au résultat du test, seul le jeu d'instructions associé au premier mot clé **Case** correspondant est exécuté.

Le jeu d'instructions associé au mot clé **Case Else** est exécuté si aucune correspondance n'a été détectée pour les mots clés **Case**.

Structures de boucle

Une structure de boucle permet de répéter l'exécution d'une série d'instructions. Les structures de boucle disponibles sont les suivantes :

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

Do...Loop

Utilisez cette structure pour exécuter une série d'instructions un nombre de fois non défini. La sortie de la boucle est effectuée lorsqu'une condition est réalisée ou non. Cette condition est une valeur ou une expression qui est évaluée comme **False** (0) ou **True** (différent de 0).

 Note :

La sortie de la boucle peut être forcée en utilisant le mot clé **Exit Do** dans les instructions exécutées.

Il existe plusieurs variations de cette structure, mais la plus usitée est la suivante :

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

Dans ce cas, la condition est évaluée en premier. Si elle est vérifiée (**True**), les instructions sont exécutées et le programme retourne au mot clé **Do While**, teste à nouveau la condition et ainsi de suite. La sortie de boucle est réalisée si la condition est évaluée comme **False**.

L'exemple suivant teste la valeur d'un compteur, incrémenté à chaque passage dans la boucle (ou itération). La sortie de la boucle est réalisée si le compteur contient la valeur 20.

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

L'exemple suivant reprend l'exemple précédent mais force la sortie de la boucle par un **Exit Do** si le compteur contient la valeur 10.

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

```
  If iCounter = 10 Then Exit Do
Loop
```

Dans ce type de structure **Do...Loop**, la condition est évaluée avant d'exécuter les instructions. Si vous souhaitez exécuter les instructions puis tester la condition, utilisez la structure **Do...Loop** suivante :

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

Note :

Ce type de structure garantit au moins une exécution des instructions.

Les deux types de structure **Do...Loop** précédents itèrent tant que la condition est réalisée (**True**). Si vous souhaitez itérer tant que la condition n'est pas réalisée (**False**), utilisez l'une des deux structures suivantes :

```
Do Until <Condition>
  <Instructions>
Loop

Do
  <Instructions>
Loop Until <Condition>
```

En utilisant ce type de structure, l'exemple précédent peut s'écrire :

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

For...Next

Utilisez cette structure pour exécuter une série d'instructions un nombre de fois déterminé. A l'inverse des structures **Do...Loop**, une boucle **For...Next** utilise une variable appelée compteur dont la valeur augmente ou diminue à chaque itération.

Note :

La sortie de la boucle peut être forcée en utilisant le mot clé **Exit For** dans les instructions exécutées.

```
For <Compteur> = <Valeur initiale> To <Valeur finale> [Step <Incrément
>]
  <Instructions>
Next [<Compteur>]
```

IMPORTANT :

Les arguments **Compteur**, **Valeur initiale**, **Valeur finale** et **Incrément** sont tous représentés par des valeurs numériques.

Note :

Incrément peut être une valeur positive ou négative. Si elle est positive la **Valeur initiale** doit être inférieure ou égale à la **Valeur finale** pour que les instructions soient exécutées. Si elle est négative, la **Valeur initiale** doit être supérieure ou égale à la **Valeur finale** pour que les instructions soient exécutées. Si l'**Incrément** n'est pas précisé, sa valeur par défaut est 1.

Lors de l'exécution d'une boucle **For...Next**, les opérations suivantes sont réalisées :

1 Le compteur est initialisé et stocke la valeur initiale,

2 Le Basic teste si la valeur du compteur est supérieure à la valeur finale. Si tel est le cas, le programme sort de la boucle.

   Note :

   Si l'incrément est négatif, le Basic teste si la valeur du compteur est inférieure à la valeur finale.

3 Les instructions sont exécutées,

4 Le compteur est incrémenté de 1 ou de la valeur spécifiée par l'incrément,

5 Les opérations 2 à 4 sont répétées.

L'exemple suivant effectue la somme des nombres pairs jusqu'à 1000 :

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

L'exemple suivant reprend l'exemple précédent mais force la sortie de la boucle par un **Exit For** si le compteur contient la valeur 500.

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

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

Opérateurs

Les opérateurs sont des symboles qui permettent de réaliser des opérations simples (addition, multiplication, ...) entre des variables et de les évaluer ou les comparer. On distingue plusieurs types d'opérateurs :

- Les opérateurs d'affectation,
- Les opérateurs de calcul,
- Les opérateurs relationnels (également appelés opérateurs de comparaison),
- Les opérateurs logiques.

Les opérateurs d'affectation

Ce type d'opérateur permet d'affecter une valeur à une variable. Le Basic AssetCenter utilise une seule variable d'affectation, le signe "=". La syntaxe d'affectation est la suivante :

```
<Variable> = <Valeur>
```

Les opérateurs de calcul

Les opérateurs de calcul permettent de modifier mathématiquement la valeur d'une variable, ou de réaliser des opérations mathématiques simples entre deux expressions.

L'opérateur +

Cet opérateur permet d'effectuer la somme de deux valeurs. La syntaxe est la suivante :

```
<Résultat> = <Expression 1> + <Expression 2>
```

Note :

Cet opérateur est utilisé tant pour effectuer la somme de deux nombres que pour concaténer des chaînes. Pour lever toute ambiguité, nous vous conseillons de réserver l'utilisation de cet opérateur à des sommes, et d'utiliser l'opérateur **&** pour concaténer des chaînes.

L'opérateur -

Cet opérateur permet d'effectuer la différence entre deux valeurs ou de signer négativement (opérateur monadique) une valeur. L'opérateur possède donc deux syntaxes :

```
<Résultat> = <Expression 1> - <Expression 2>
```

ou

```
- <Expression>
```

L'opérateur *

Cet opérateur permet d'effectuer la multiplication de deux valeurs. La syntaxe est la suivante :

```
<Résultat> = <Expression 1> * <Expression 2>
```

L'opérateur /

Cet opérateur permet d'effectuer la division de deux valeurs. La syntaxe est la suivante :

```
<Résultat> = <Expression 1> / <Expression 2>
```

L'opérateur ^

Cet opérateur permet d'élever une valeur à la puissance d'un exposant. La syntaxe est la suivante :

```
<Résultat> = <Expression 1> ^ <Expression 2>
```

Note :

Dans cette syntaxe l'expression 1 ne peut être négative que si l'expression 2 (l'exposant) est une valeur entière. Lorsqu'une expression effecute plusieurs opérations d'exposants en série, le Basic les interprète logiquement, de gauche à droite.

L'opérateur Mod

Cet opérateur permet de calculer le reste de la division euclidienne de deux valeurs. La syntaxe est la suivante :

```
<Résultat> = <Expression 1> Mod <Expression 2>
```

Note :

Les nombres à virgule flottante sont systématiquement arrondis à l'entier le plus proche.

L'exemple suivant renvoie la valeur 4 (6.8 est arrondi à l'entier le plus proche lors du calcul, soit 7):

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

Les opérateurs relationnels

Les opérateurs relationnels permettent de comparer des valeurs. Le tableau ci-dessous propose une vue d'ensemble des opérateurs relationnels :

Opérateur	Dénomination	Description	Syntaxe
=	Opérateur d'égalité	Compare deux valeurs et vérifie leur égalité	`<Expression 1> = < Expression 2>`
<	Opérateur d'infériorité stricte	Vérifie qu'une valeur est strictement inférieure à une autre	`<Expression 1> < < Expression 2>`
<=	Opérateur d'infériorité	Vérifie qu'une valeur est inférieure ou égale à une autre	`<Expression 1> <= <Expression 2>`

Opérateur	Dénomination	Description	Syntaxe
>	Opérateur de supériorité stricte	Vérifie qu'une valeur est strictement supérieure à une autre	`<Expression 1> > <Expression 2>`
>=	Opérateur de supériorité	Vérifie qu'une valeur est supérieure ou égale à une autre	`<Expression 1> >= <Expression 2>`
<>	Opérateur de différence	Vérifie qu'une valeur est différente d'une autre	`<Expression 1> <> <Expression 2>`

Les opérateurs logiques

Les opérateurs logiques permettent de vérifier la véracité de plusieurs conditions.

L'opérateur And

Cet opérateur effectue un ET logique (les deux conditions doivent être vérifiées) entre deux expressions. La syntaxe est la suivante :

```
<Résultat> = <Expression 1> And <Expression 2>
```

Si chaque expression est évaluée comme vraie (**True**), le résultat est vrai (**True**). Si l'une des deux expressions est évaluée comme fausse (**False**), le résultat est évalué comme faux (**False**).

L'opérateur Or

Cet opérateur effectue un OU logique (une des deux conditions doit être vérifiée) entre deux expressions. La syntaxe est la suivante :

```
<Résultat> = <Expression 1> Or <Expression 2>
```

Si l'une ou les deux expressions sont évaluées comme vraies (**True**), le résultat est vrai (**True**).

L'opérateur Xor

Cet opérateur effectue un OU exclusif (une seule des deux conditions doit être vérifiée) entre deux expressions. La syntaxe est la suivante :

```
<Résultat> = <Expression 1> Xor <Expression 2>
```

Si une seule des deux expressions est évaluée comme vraie (**True**), le résultat est vrai (**True**).

L'opérateur Not

Cet opérateur effectue la négation logique d'une expression. La syntaxe est la suivante :

```
<Résultat> = Not <Expression 1>
```

Si l'expression est évaluée comme vraie (**True**), le résultat est faux (**False**). Si l'expression est évaluée comme fausse (**False**), le résultat est vrai (**True**).

Priorité des opérateurs

Lorsque plusieurs opérateurs sont associés, l'ordre de priorité suivant est respecté dans l'évaluation des expressions. La liste ordonnée suivante classe les opérateurs dans un ordre de priorité décroissant :

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

Gestion de fichiers

Le Basic AssetCenter permet de manipuler des fichiers de façon simple. Les opérations les plus usuelles (lecture, écriture,...) sont disponibles en standard.

Rappel préliminaire sur les fichiers

Un fichier est la façon dont un programme voit un objet externe. Il s'agit d'une collection d'enregistrements logiques, éventuellement structurée, sur laquelle le programme peut exécuter un ensemble d'opérations élémentaires (lecture,

écriture, ...). Un enregistrement logique représente l'ensemble minimum de données qui peut être manipulé par une seule opération élémentaire.

Le Basic AssetCenter gère uniquement les fichiers dits séquentiels. Dans un fichier séquentiel, les opérations se résument essentiellement en la lecture de l'enregistrement suivant ou l'écriture d'un nouvel enregistrement en fin du fichier séquentiel. Il n'est pas possible de réaliser simultanément lecture et écriture des enregistrements.

En lecture, le fichier séquentiel est initialement positionné sur le premier enregistrement logique. Chaque opération de lecture transfère un enregistrement dans une zone interne (en général une variable) du programme et positionne le fichier sur l'enregistrement suivant. Une opération permet de déterminer s'il reste des enregistrements à lire (clause **EOF** : End Of File).

En écriture, le fichier séquentiel peut être initialement vide ou positionné après le dernier enregistrement du fichier. Chaque opération d'écriture transfère des données stockées dans une zone interne (en général une variable) du programme, dans un enregistrement du fichier et positionne le fichier après cet enregistrement.

 Note :

Une des caractéristiques essentielles d'un fichier séquentiel est que les enregistrements sont lus dans l'ordre où ils ont été écrits.

Ouvrir et fermer des fichiers

La clause Open

Il s'agit de la clause de base pour toute manipulation de fichier. Elle permet d'ouvrir un fichier, que ce soit pour le lire, le créer, ou y écrire. La syntaxe est la suivante :

```
Open <Chemin du fichier> For <Mode> [Access <Type d'accès>] As [#]<Numéro de fichier>
```

Les paramètres de cette clause sont détaillés dans le tableau suivant :

Paramètre	Description
<Chemin du fichier>	Chaîne de caractères spécifiant le fichier concerné par l'opération. Cette chaîne peut contenir le chemin complet du fichier.
<Mode>	Précise le mode de traitement du fichier. Ce paramètre peut contenir l'une des valeurs suivantes : • **Input** : le fichier est ouvert en lecture. • **Output** : le fichier est ouvert en écriture. Si le fichier existe et possède du contenu, celui-ci est écrasé. • **Append** : le fichier est ouvert en écriture. Si le fichier existe et possède du contenu, le nouveau contenu est ajouté en fin de fichier. • **Binary** : le fichier est ouvert en lecture binaire.
<Type d'accès>	Précise les opérations réalisables sur un fichier ouvert. Si le fichier est ouvert par un autre processus et que le type d'accès précisé n'est pas autorisé, la commande d'ouverture de fichier échoue. Ce paramètre peut contenir l'une des valeurs suivantes : • **Read** : le fichier est ouvert en lecture seule • **Write** : le fichier est ouvert en écriture seule • **Read Write** : le fichier est ouvert en lecture-écriture. Ce type d'accès n'est disponible que pour des accès en mode **Binary** ou **Append**.

Paramètre	Description
<**Numéro de fichier**>	Identifie le fichier par un numéro unique compris entre 1 et 511. La fonction **FreeFile**() permet de déterminer le prochain numéro de fichier disponible.

Note :

Gardez en mémoire les points suivants :

- Tout fichier doit être ouvert par la clause **Open** avant de lire ou d'écrire des informations dans ce fichier.
- En mode **Append**, **Binary** ou **Output**, si le fichier référencé n'existe pas, il est créé.
- En mode **Binary** ou **Input**, vous pouvez ouvrir un fichier en utilisant un numéro différent sans fermer le fichier au préalable. En mode **Append** ou **Output**, vous devez impérativement fermer un fichier avant de l'ouvrir à nouveau sous un numéro différent.

La clause Close

Cette clause permet de fermer un fichier préalablement ouvert au moyen de la clause **Open**(). La syntaxe est la suivante :

```
Close [<Liste de fichiers>]
```

L'argument optionnel <**Liste de fichiers**> peut contenir un ou plusieurs numéros de fichiers. La syntaxe de cet argument optionnel est la suivante :

```
[[#]<Numéro de fichier>][,[#]<Numéro de fichier>]...
```

Note :

Si vous omettez le paramètre de cette clause, tous les fichiers actifs ouverts par une clause **Open**() sont fermés.

Lire des données d'un fichier

Deux clauses sont disponibles pour la lecture des données d'un fichier. L'utilisation de l'une ou l'autre de ces clauses dépend du mode d'accès spécifié pour le fichier. Les deux clauses sont les suivantes :

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

La clause Input

Cette clause est utilisée pour lire un nombre déterminé de caractères à partir d'un fichier ouvert en mode **Binary** ou **Input**. La syntaxe de cette clause est la suivante :

```
Input (<Nombre de caractères à lire>,[#]<Numéro de fichier>)
```

La clause Line Input

Cette clause est utilisée pour lire une ligne de données d'un fichier séquentiel, et la stocker dans une variable de type **String** ou **Variant**. La syntaxe de cette clause est la suivante :

```
Line Input #<Numéro de fichier>, <Nom de la variable>
```

IMPORTANT :

La clause lit les caractères un par un jusqu'à ce qu'un retour chariot ou un ensemble retour chariot - fin de ligne soit rencontré.

Ecrire des données dans un fichier

Une seule clause, **Print**, permet l'écriture de données dans un fichier. La syntaxe de cette clause est la suivante :

```
Print #<Numéro de fichier>, [<Données>]
```

CHAPITRE

2 Classification des fonctions

La classification des fonctions s'effectue à trois niveaux différents. Une fonction donnée peut être classée par :

Familles de fonctions

Les fonctions disponibles dans l'environnement AssetCenter peuvent être regroupées en plusieurs grandes familles :

- les fonctions reconnues par AssetCenter : il s'agit essentiellement des fonctions utilisables dans les parties scriptables (en Basic) du logiciel.
- les fonctions reconnues par la bibliothèque AssetCenter API : ces fonctions peuvent être appelées par des outils externes ou par un programme écrit dans un langage évolué.

Ces grandes familles de fonctions se recoupent. Par exemple, certaines fonctions des AssetCenter API sont utilisables dans les scripts Basic du logiciel. On dit alors qu'une telle fonction, provenant à l'origine des AssetCenter API, est

"exposée" dans les scripts Basic internes à AssetCenter. Si la syntaxe d'une telle fonction peut alors varier sensiblement, son comportement reste inchangé.

Champs d'application des fonctions

Les fonctions décrites dans ce document sont utilisables dans au moins un des contextes suivants :

- Librairies AssetCenter API. En particulier, les fonctions sont accessibles pour le développement d'applications Get-It.
- Script de configuration d'un champ ou d'un lien (menu contextuel **Configurer l'objet** ou AssetCenter Database Administrator) et par extension **Script de calcul** (Nom SQL : memScript) d'un champ calculé :
  - Valeur par défaut,
  - Obligation de saisie,
  - Historisation,
  - Lecture seule,
  - ...
- Actions de type Script :
  - Script défini dans le champ **Script de l'action** (Nom SQL : Script) d'une action Script.
- Assistants AssetCenter :
  - Script "FINISH.DO" d'un assistant.
  - Script de définition des valeurs des propriétés d'un noeud.

Modules applicatifs

Chaque fonction est associée à un ou plusieurs modules applicatifs. Un module applicatif décrit la nature des opérations effectuées par la fonction. Les modules applicatifs sont les suivants :

- Builtin : fonctions Basic classiques et fonctions de conversion, de manipulation de chaînes, etc.
- Techniques : connexion à une base de données, gestion des objets tables, champs, liens, index, enregistrements, requêtes.
- Fonctionnel : fonctions génériques, orientées métier.

- Câble.
- Achats.
- Support.
- Refacturation.
- Assistants.
- Actions.
- Graphiques.

3 Conventions

CHAPITRE

Ce chapitre décrit :

Conventions d'écriture

La syntaxe des fonctions et des exemples proposés respecte les conventions d'écriture suivantes :

[]	Ces crochets encadrent un paramètre optionnel. Ne les tapez pas dans votre commande. Exception : dans les scripts Basic, lorsque les crochets encadrent le chemin d'accès à des données de la base, ils doivent figurer dans le script, comme le montre l'exemple ci-dessous : `[Lien.Lien.Champ]`

<>	Ces crochets encadrent un paramètre décrit en langage clair. Ne les tapez pas dans votre commande et remplacez le texte qu'ils encadrent par l'information qui doit y figurer.
{}	Ces accolades encadrent la définition d'un noeud ou d'un bloc de script multilignes pour une propriété.
|	La barre verticale sépare les paramètres possibles qui figurent dans les accolades.

Les mises en forme suivantes ont des significations particulières :

`Police fixe`	Commande DOS, paramètre de fonction ou formatage de données.
`Exemple`	Exemple de code ou de commande.
...	Portion de code ou de commande omise.
Nom d'objet	Les noms de champs, d'onglets, de menus, de fichiers sont en caractères gras.

Format des constantes de type Date+Heure dans les scripts

Les dates référencées dans les scripts sont exprimées au format international, indépendamment des options d'affichage spécifiées par l'utilisateur :

```
yyyy/mm/dd hh:mm:ss
```

Exemple :

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

Note :

Le tiret ("-") peut également être utilisé comme séparateur de date.

A propos des dates

Les dates sont exprimées différemment en Basic interne et à partir d'outils externes :

- En Basic, une date peut être exprimée au format international ou sous la forme d'un nombre à virgule flottante (type "Double"). Dans ce dernier cas, la partie entière de ce nombre représente le nombre de jours écoulés depuis le 30/12/1899 à 0:00, la partie décimale représente la fraction écoulée dans le jour courant.

- En externe, les dates sont exprimées sous la forme d'un entier long (type "Long" 32 bits) qui représente le nombre de secondes écoulées depuis le 01/01/1970 à 0:00, indépendamment d'un quelconque fuseau horaire (heure UTC).

Format des constantes de type Durée

Dans les scripts, les durées sont stockées et exprimées en secondes. Par exemple, pour fixer la valeur par défaut d'un champ de type "Durée" à 3 jours, vous devez utiliser le script suivant :

```
RetVal=259200
```

De même, les fonctions qui calculent une durée, comme la fonction "AmWorkTimeSpanBetween()", fournissent un résultat en secondes.

 Note :

Dans les calculs financiers, AssetCenter tient compte des simplifications habituellement usitées. Dans ce cas seulement, une année vaut 12 mois et un mois vaut 30 jours (d'où : 1 année = 360 jours).

4 Définitions

CHAPITRE

Ce chapitre regroupe les définitions de quelques termes essentiels.

Vous y trouverez les définitions suivantes :

Définition d'une fonction

Une fonction est un programme qui effectue des opérations et renvoie à l'utilisateur une valeur, appelée "valeur de retour" ou "code de retour".

Voici un exemple de syntaxe d'appel d'une fonction par le Basic interne de AssetCenter :

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

Voici à présent la syntaxe d'appel de la même fonction au travers des AssetCenter API :

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

Définition du lien virtuel CurrentUser

Définition

CurrentUser peut être considéré comme un lien qui part de toutes les tables et pointe vers l'enregistrement de la table des services et personnes correspondant à l'utilisateur courant.

- Sous la forme **CurrentUser**, il pointe sur l'enregistrement correspondant à l'utilisateur courant et renvoie la chaîne de description de la table des services et personnes.
- Sous la forme **CurrentUser.Champ**, il renvoie la valeur du champ pour l'utilisateur courant.

 Note :

Ce lien virtuel n'est pas affiché dans la liste des champs et des liens et n'est donc pas accessible directement dans le constructeur de scripts interne à AssetCenter. Vous devez saisir cette expression à la main.

Equivalences

Les fonctions **AmLoginName**() et **AmLoginId**() qui fournissent respectivement le **Nom (Nom SQL : Name)** et le numéro d'identifiant (Nom SQL : lPersId) de l'utilisateur courant peuvent être considérées comme des fonctions dérivées de **CurrentUser**. En effet, on a les équivalences suivantes :

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

Restrictions

CurrentUser ne peut fonctionner si un contexte est défini (le contexte étant une table).

S'il n'existe pas de contexte, vous devez faire appel à une autre fonction.

Exemple :

Vous souhaitez créer une action non contextuelle qui exécute un fichier dont le chemin dépend de l'utilisateur connecté à la base AssetCenter.

Si l'action avait été contextuelle, vous auriez pu créer une action de type **Exécutable** dont le champ **Dossier** (Folder) aurait, par exemple, valu : **c:\scripts\[CurrentUser.Name]**.

Toutefois, lorsqu'une action de type **Exécutable** n'a pas de contexte, **[CurrentUser.Name]** est considéré comme du texte fixe.

Vous devez donc trouver une autre solution, comme, par exemple, de créer une action non contextuelle de type **Script** avec le script :

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

Définition d'un descripteur

Un descripteur représente un identifiant unique sur un objet. Dans le contexte de AssetCenter, cet objet peut être un champ, un lien, un index, une requête, un enregistrement, une table ou une connexion. Les descripteurs sont des entiers (type "Long") codés sur 32 bits.

Note :

Un descripteur valide ne peut avoir une valeur nulle (NULL).

A partir d'un outil externe, vous avez également accès à un descripteur de connexion (à la base de données).

Définition d'un code d'erreur

Lorsque l'exécution d'une fonction échoue, un code d'erreur est renvoyé.

A partir d'outils externes

A partir d'outils externes, le code d'erreur et le message qui lui est associé peuvent être récupérés respectivement grâce aux fonctions "AmLastError()"

et "AmLastErrorMsg()". Il peut être détruit au moyen de la fonction "AmClearLastError()".

Note :

Tout nouvel appel de fonction efface le code d'erreur et le message précédents.

En interne

En interne (dans les scripts Basic par exemple), la description et le code de la dernière erreur peuvent être retrouvés respectivement au moyen des fonctions **Err.Number** et **Err.Description**.

Note :

En interne, vous n'avez pas besoin de programmer une gestion des erreurs. Le script erroné s'arrête et un rollback est effectué, si nécessaire, sur la base de données.

Vous pouvez déclencher volontairement un message d'erreur en utilisant la fonction Err.Raise dont la syntaxe est la suivante :

```
Err.Raise (<Numéro de l'erreur>, <Message d'erreur>)
```

Note :

Lorsque la création ou la modification d'un enregistrement est invalidé par la valeur du champ "Validité" pour la table concernée, il est judicieux de déclencher un message d'erreur au moyen de la fonction **Err.Raise** afin de prévenir l'utilisateur (codes 12006 ou 12007). Si vous ne le faites pas, celui-ci ne comprendra pas nécessairement pourquoi il ne peut ni modifier, ni créer l'enregistrement.

Le tableau ci-dessous répertorie les codes d'erreur les plus fréquents :

Code d'erreur	Signification
12001	Erreur indéfinie
12002	Mauvais paramètre pour une fonction
12003	Descripteur invalide ou objet détruit

Code d'erreur	Signification
12004	Plus de données disponible. Cette erreur arrive classiquement lors de l'exécution de requêtes. Quand le résultat de la requête ne renvoie aucune donnée, cette erreur est déclenchée.
12005	Erreur interne du serveur de base de données
12006	Valeur non valide (type incorrect pour un paramètre, etc.)
12007	Enregistrement non valide (par exemple, un champ obligatoire n'a pas été renseigné)
12008	Problèmes de droits d'accès à des données de la base
12009	Fonction obsolète ou non implémentée
12010	Nombre maximum de connexions à la base de données dépassé

5 Typage des fonctions et des paramètres de fonctions

CHAPITRE

Vous trouverez dans ce chapitre les informations suivantes :

Liste des types

Le tableau ci-dessous récapitule les différents types possibles pour une fonction ou un paramètre :

Type	Signification
Integer	Nombre entier de -32 768 à +32 767.
Long	Nombre entier de -2 147 483 647 à +2 147 483 646.
Single	Nombre à virgule flottante de 4 octets (simple précision).
Double	Nombre à virgule flottante de 8 octets (double précision).

Type	Signification
String	Texte pour lequel tous les caractères sont acceptés.
Date	Date ou Date+Heure.
Variant	Type générique pouvant représenter n'importe quel type.

Note :

Ces types ne sont pas tous disponibles à partir d'un outil externe. Seuls les types Long, Double et String sont disponibles. Le Variant n'existe pas et les objets de types Integer et Date sont représentés par un Long.

Type d'une fonction

Le type d'une fonction correspond au type de la valeur retournée par la fonction. Nous vous invitons à faire particulièrement attention à cette information car elle peut être à l'origine d'erreurs de compilation et d'exécution de vos programmes.

Par exemple, vous ne pouvez pas utiliser une fonction renvoyant une valeur d'un certain type dans la définition de la valeur par défaut d'un champ d'un type différent. Essayez par exemple d'affecter ce script de valeur par défaut à n'importe quel champ de type "Date" ou "Date and time" :

```
RetVal=AmLoginName()
```

La fonction "AmLoginName()" renvoie le nom de l'utilisateur connecté sous la forme d'une chaîne de caractères (type "String"). Cette valeur de retour est donc dans un format incompatible avec celui d'un champ de type "Date" et AssetCenter affiche un message d'erreur.

Type d'un paramètre

Les paramètres utilisés dans les fonctions possèdent également un type que vous devez impérativement respecter pour la bonne exécution de la fonction. Dans la syntaxe des fonctions, les paramètres sont préfixés en fonction de leur type. Pour éviter toute confusion possible, les préfixes utilisés dans cette référence sont différents suivant la syntaxe (API ou Basic) de la fonction. Le

tableau ci-dessous propose pour chaque type une équivalence entre le préfixe utilisé dans la syntaxe API et celui utilisé dans la syntaxe Basic :

Type	Préfixe utilisé dans la syntaxe API	Préfixe utilisé dans la syntaxe Basic
Integer	"i"	"i"
Long	"h" pour un descripteur ou "l" pour un nombre	"l"
Double	"d"	"d"
String	"char*psz"	"str"
Date	"ltm"	"dt"
Variant	"v"	"v"

II Utilisation des API

PARTIE

CHAPITRE 6

Préambule

Les AssetCenter API sont fournies sous la forme d'un fichier DLL 32 bits utilisable sous Windows 95/98, Windows NT ou Windows 2000.

Elles ont été testées avec succès dans des environnements de développement suivants :

- Visual Basic 4.0, 5.0 et 6.0,
- Visual C++ 4.0, 5.0 et 6.0,
- Tous les produits de la gamme Microsoft utilisant le VBA (Visual Basic for Applications)

Note :

Les API sont à priori compatibles avec tous les outils autorisant l'utilisation de bibliothèques de fonctions externes (DLL).

Avertissement

L'utilisation des AssetCenter API est assujettie à une bonne connaissance du modèle conceptuel de données utilisé par AssetCenter en général et de la structure de la base de données en particulier.

Toutes les informations utiles sur la structure de la base de données sont regroupées dans le manuel intitulé "Manuel de référence : Administration et utilisation avancée", chapitre "Structure de la base de données" ainsi que dans les fichiers "database.txt" et "tables.txt", situés dans le sous-répertoire "doc\infos" du répertoire d'installation de AssetCenter.

Installation

Avant toute utilisation des AssetCenter API, nous vous recommandons de procéder à une installation complète de AssetCenter. Vous pourrez ainsi contrôler facilement l'accès aux bases de données à partir de votre ordinateur, créer et configurer rapidement les bases de données sur lesquelles vous souhaitez travailler. Les API utilisant les mêmes couches de base de données et les mêmes informations de configuration que AssetCenter pour l'accès aux sources de données, vous pouvez rapidement détecter, à partir de l'interface graphique de AssetCenter, les problèmes rencontrés lors de l'utilisation des API.

Les étapes classiques pour l'installation d'un environnement de développement sous AssetCenter sont les suivantes :

- Installation d'une version 32 bits de AssetCenter avec le composant AssetCenter API.
- Configuration de la source de données et test d'accès à la base de données sous AssetCenter.
- Utilisation de votre environnement de développement pour appeler les fonctions des AssetCenter API.

Nous vous recommandons de vous familiariser avec les AssetCenter API en utilisant une base de données de démonstration ou toute autre source ne contenant aucune donnée sensible.

Fichier de configuration .ini associé à la DLL

▶ Manuel intitulé **AssetCenter - Installation**, chapitre **Fichiers .ini et .cfg**.

Consultez en particulier les sections :

- **Fichiers .ini et .cfg disponibles**
- **Contrôler la modification des fichiers .ini**

7 Méthodologie

CHAPITRE

Voici, étape par étape, une approche classique de l'utilisation des AssetCenter API :

1. Création d'une requête AQL du type :
   `SELECT AssetTag, User.Name, Supervisor.Name FROM amPortfolio`

   Note :

   Une bonne solution pour composer simplement une requête AQL est d'utiliser AssetCenter Export.
2. Récupération des résultats de la requête et des descripteurs sur tous les objets que vous souhaitez utiliser.
3. Utilisation des ces descripteurs pour mettre à jour les informations contenues dans les objets correspondants.
4. Réalisation d'un "Commit" (pour accepter toutes les modifications) ou d'un "Rollback" (pour annuler toutes les modifications) pour la transaction en cours.

8 Concepts et exemples

CHAPITRE

Vous trouverez dans cette partie les informations suivantes :

Concepts

AssetCenter a été pensé et conçu dans une approche "objet" qui se retrouve également dans les API. Les DLL de Windows requièrent l'utilisation d'un modèle "flat" proche du C. Les AssetCenter API contournent cette limitation en utilisant des "descripteurs" ("handles" en anglais) sur les objets créés par l'utilisateur. Cette approche permet aux langages qui ne sont pas orientés "objet" d'accéder au modèle de AssetCenter.

Avant tout autre chose, votre programme doit utiliser la fonction "AmStartUp()" pour initialiser l'appel aux bibliothèques AssetCenter. De même, la dernière fonction appelée par votre programme doit impérativement être la fonction "AmCleanUp()".

Avant tout accès à un objet d'une base, une connexion valide doit être établie entre l'utilisateur et cette base de données. Cette connexion est identifiée par un "descripteur" sur un objet "connexion" (ce descripteur est alors utilisé dans toutes les fonctions des API qui interagissent avec la base de données. Il correspond au paramètre "hApiCnxBase"). Cet objet peut alors être utilisé pour créer des requêtes et accéder aux enregistrements.

 Note :

Notez que tous les objets d'une base de données sont liés à une connexion. Il en résulte que les informations sur les droits d'accès par exemple peuvent être contrôlées.

La première étape consiste donc en la création d'une connexion utilisant une source de données, un login et son mot de passe valides.

 Avertissement :

Lorsque vous vous connectez à une base de données AssetCenter au travers des AssetCenter API, un jeton de connexion est utilisé.

Manipuler les dates

Pour lire une date, vous avez la possibilité d'utiliser l'une des deux fonctions suivantes sur un champ de type "Date" ou "Date and time" :

- "AmGetFieldLongValue()" qui renvoie la date sous la forme d'un "Long" Unix (UTC). Préférez cette fonction pour les calculs faisant intervenir les dates.
- "AmGetFieldStrValue()" qui renvoie la date sous la même forme que le panneau de configuration de Windows. Cette date respecte les fuseaux horaires. Utilisez cette fonction pour l'affichage.

Premier exemple

L'exemple suivant, rédigé en C, déclare une connexion à la base de démonstration :

```
long lCnx ;
lCnx = AmOpenConnection(ACDemo351FRA, Admin , ) ;
```

"lCnx" représente un "descripteur" sur un objet "connexion". Ce descripteur est utilisé pour identifier la connexion que vous venez de déclarer.

Cette connexion peut dès lors être utilisée pour créer des requêtes et accéder à la base de données. L'exemple suivant, rédigé en C, définit une requête sur la table des biens et parcourt les résultats de cette requête :

```
#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();
```

Second exemple

Les requêtes sont utilisées pour localiser les objets dans la base de données. Lorsque vous devez mettre à jour un enregistrement, un descripteur sur l'objet "enregistrement" doit être récupéré au moyen d'une requête. L'enregistrement peut alors être traité au moyen des autres fonctions des AssetCenter API.

L'exemple suivant illustre la modification d'un champ d'un enregistrement donné :

```
/* 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 = 'Com
paq' and barcode='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();
```

Cet exemple illustre la récupération d'un "descripteur" sur un enregistrement par le biais d'une requête. La requête est alors analysée pour localiser un élément de l'enregistrement (dans cet exemple un champ). Il est également possible d'utiliser la fonction "AmQueryExec()" pour récupérer plusieurs enregistrements en une seule requête, puis de parcourir ces enregistrements et récupérer les "descripteurs" des éléments qui vous intéressent.

Note :

Par souci de simplification, cet exemple ne traite pas tous les codes d'erreur possibles.

III Référence alphabétique

PARTIE

CHAPITRE

9 Référence alphabétique

Abs()

Renvoie la valeur absolue d'un nombre.

Syntaxe Basic interne

Function Abs(dValue As Double) As Double

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓

	Utilisable
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **dValue** : Nombre dont vous souhaitez connaître la valeur absolue.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

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

AmActionDde()

Cette fonction lance une requête DDE à destination d'une application qui gère les liens DDE. Grâce à cette fonction, AssetCenter peut piloter une autre application par l'intermédiaire d'un lien DDE. Cette fonction est équivalente à une action de type DDE.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strService** : Ce paramètre contient le nom du service DDE proposé par l'exécutable que vous souhaitez solliciter. Veuillez vous reporter à la documentation de cet exécutable pour connaître la liste des services DDE qu'il propose.
- **strTopic** : Ce paramètre contient le thème, c'est-à-dire le contexte dans lequel l'action DDE doit être effectuée.
- **strCommand** : Ce paramètre contient les commandes que l'application externe doit exécuter. Vous devez respecter la syntaxe imposée par l'application externe.
- **strFileName** : Si le service n'est pas présent en mémoire, vous devez le charger en précisant dans ce paramètre le nom de l'exécutable (ou celui d'un fichier quelconque si celui-ci est associé à un exécutable par l'intermédiaire du gestionnaire de fichiers de Windows) qui active le service.
- **strDirectory** : Ce paramètre contient le chemin d'accès du fichier précisé dans **strFileName**.
- **strParameters** : Ce paramètre contient les différents paramètres à fournir à l'exécutable qui active le service lors de son lancement.

- **strTable** : Paramètre optionnel, utilisé dans le cas où l'action est contextuelle. Il indique le nom SQL de la table contenant l'enregistrement auquel s'applique l'action.
- **lRecordId** : Paramètre optionnel, utilisé dans le cas où l'action est contextuelle. Il indique l'identifiant de l'enregistrement auquel s'applique l'action.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmActionExec()

Cette fonction lance une application de type ".exe", ".com", ".bat", ".pif". Vous pouvez également faire référence à des documents de tous types, à condition que leur extension soit associée à un exécutable par le gestionnaire de fichiers de Windows. Cette fonction est équivalente à une action de type "Exécutable".

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓

	Utilisable
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strFileName** : Ce paramètre contient le nom de l'exécutable ou celui d'un document quelconque (associé à un exécutable par l'intermédiaire du gestionnaire de fichiers de Windows).
- **strDirectory** : Ce paramètre contient le chemin d'accès du fichier précisé dans le paramètre **strFileName**.
- **strParameters** : Ce paramètre optionnel contient les différents paramètres à fournir à l'exécutable lors de son lancement.
- **strTable** : Paramètre optionnel, utilisé dans le cas où l'action est contextuelle. Il indique le nom SQL de la table contenant l'enregistrement auquel s'applique l'action.
- **lRecordId** : Paramètre optionnel, utilisé dans le cas où l'action est contextuelle. Il indique l'identifiant de l'enregistrement auquel s'applique l'action.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

Exemple

Cet exemple exécute l'explorer de Windows NT (situé dans le dossier "WinNT" du disque "C") :

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

AmActionMail()

Cette fonction émet un message via l'un des messageries gérées par AssetCenter :

- Messagerie interne.
- Messagerie externe au standard VIM (Lotus Notes, ...).
- Messagerie externe au standard MAPI (Microsoft Exchange, Microsoft Outlook, ...).
- Messagerie externe au standard SMTP (standard Internet).

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **strTo** : Ce paramètre contient la liste des adresses des destinataires du message, sous la forme messagerie:adresse. Le point-virgule est utilisée comme séparateur.
- **strCc** : Ce paramètre contient la liste des adresses des destinataires en copie du message. Le point-virgule est utilisée comme séparateur.
- **strCcc** : Ce paramètre contient la liste des adresses des destinataires en copie cachée du message (ils n'apparaissent pas dans la liste des destinataires). Le point-virgule est utilisée comme séparateur.
- **strSubject** : Ce paramètre contient l'intitulé du message.
- **strMessage** : Ce paramètre contient le corps du message.
- **iPriority** : Ce paramètre définit la priorité d'envoi du message
  - 0 priorité basse.
  - 1 priorité normale.
  - 2 priorité haute.
- **bAcknowledge** : Ce paramètre précise si l'émetteur du message reçoit un accusé de réception :
  - 0 : l'émetteur ne reçoit pas d'accusé de réception.
  - 1 : l'émetteur reçoit un accusé de réception.
- **strRefObject** : Ce paramètre ne sert qu'aux messages adressés à la messagerie interne d'AssetCenter. Il s'agit du nom SQL du lien qu'il faut suivre depuis l'enregistrement correspondant au contexte d'exécution pour atteindre l'objet référencé. Il peut s'agir du lien virtuel CurrentUser.
- **strTable** : Paramètre optionnel, utilisé dans le cas où l'action est contextuelle. Il indique le nom SQL de la table contenant l'enregistrement auquel s'applique l'action.
- **lRecordId** : Paramètre optionnel, utilisé dans le cas où l'action est contextuelle. Il indique l'identifiant de l'enregistrement auquel s'applique l'action.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmActionPrint()

Cette fonction déclenche l'impression d'un rapport sur un enregistrement donné de la base.

Syntaxe Basic interne

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

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **lReportId** : Ce paramètre contient l'identifiant du rapport à imprimer.
- **lRecordId** : Ce paramètre contient l'identifiant de l'enregistrement concerné par le rapport. Par défaut, ce paramètre prend la valeur "0". La table concernée est implicitement définie par le rapport.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmActionPrintPreview()

Cette fonction déclenche un aperçu avant impression d'un rapport sur un enregistrement donné de la base.

Syntaxe Basic interne

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

Champ d'application

Version : 3.60

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **lReportId** : Ce paramètre contient l'identifiant du rapport concerné.
- **lRecordId** : Ce paramètre contient l'identifiant de l'enregistrement concerné par le rapport. Par défaut, ce paramètre prend la valeur "0".

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmActionPrintTo()

Cette fonction déclenche l'impression d'un rapport sur un enregistrement donnée de la base et sur une imprimante donnée.

Syntaxe Basic interne

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

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strPrinterName** : Ce paramètre contient le nom de l'imprimante sur laquelle s'effectue l'impression.
- **lReportId** : Ce paramètre contient l'identifiant du rapport à imprimer.
- **lRecordId** : Ce paramètre contient l'identifiant de l'enregistrement concerné par le rapport. Par défaut, ce paramètre prend la valeur "0".

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmAddAllPOLinesToInv()

Cette fonction ajoute l'intégralité d'une commande à une facture fournisseur existante.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lPOrdId** : Ce paramètre contient l'identifiant de commande à ajouter à la facture fournisseur.

- **lInvId** : Ce paramètre contient l'identifiant de la facture fournisseur à laquelle est ajoutée la commande.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmAddCatRefAndCompositionToPOrder()

Cette fonction permet d'ajouter le contenu complet d'une référence catalogue à une commande donnée.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	

	Utilisable
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lPOrderId** : Ce paramètre contient l'identifiant de la commande à compléter.
- **lCatRefId** : Ce paramètre contient l'identifiant de la référence catalogue.
- **dCatRefQty** : Ce paramètre contient la quantité (dans l'unité associée au produit) à ajouter.
- **lRequestId** : Ce paramètre contient l'identifiant de la demande que cette commande va satisfaire.
- **dUnitPrice** : Ce paramètre contient le prix unitaire du produit de la référence catalogue.
- **strCur** : Ce paramètre contient le code de la devise dans laquelle le prix unitaire est exprimé

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

Remarques

Note :

Cette fonction permet notamment d'utiliser la composition de produits d'une référence catalogue pour enrichir une commande.

AmAddCatRefToPOrder()

Cette fonction permet d'ajouter une référence catalogue à une commande existante.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lRequestLineId** : Ce paramètre contient l'identifiant de la ligne de demande associée à la commande.
- **lCatRefId** : Ce paramètre contient l'identifiant de la référence catalogue à ajouter.
- **lPOrderId** : Ce paramètre contient l'identifiant de la commande sur laquelle porte l'opération.
- **dQty** : Ce paramètre contient la quantité (dans l'unité associée au produit) à ajouter.
- **bCanMerge** : Ce paramètre permet de préciser si l'ajout peut être fusionné avec une ligne déjà existante dans la commande.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmAddEstimLinesToPO()

Cette fonction ajoute toutes les lignes de devis d'un devis à une commande existante.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **lEstimId** : Ce paramètre contient l'identifiant du devis à ajouter à la commande.
- **lPOrdId** : Ce paramètre contient l'identifiant de la commande à laquelle sont ajoutées toutes les lignes de devis du devis.
- **bMergeLines** : Ce paramètre permet de préciser si les lignes de demande identiques doivent être combinées (**bMergeLines**=1) pour n'en créer qu'une seule. Les quantités décrites sur les lignes à combiner sont alors ajoutées et une seule ligne est créée.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmAddEstimLineToPO()

Cette fonction ajoute une ligne de devis à une commande existante.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lEstimLineId** : Ce paramètre contient l'identifiant de la ligne de devis à ajouter à la commande.
- **lPOrdId** : Ce paramètre contient l'identifiant de la commande à laquelle est ajoutée la ligne de devis.
- **bMergeLines** : Ce paramètre permet de préciser si les lignes de demande identiques doivent être combinées (**bMergeLines**=1) pour n'en créer qu'une seule. Les quantités décrites sur les lignes à combiner sont alors ajoutées et une seule ligne est créée.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmAddLicContentToRequest()

Cette fonction ajoute à une demande d'achat toutes les installations logicielles couvertes par une licence.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : ?

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lRequestId** : Ce paramètre contient l'identifiant de la demande d'achat concernée par l'opération.
- **lLicModelId** : Ce paramètre contient l'identifiant du modèle de la licence.
- **lParent** : Ce paramètre contient l'identifiant de l'élément de parc ou de la ligne de demande qui sera le parent des lignes de demandes créées.
- **bExternalParent** : Si ce paramètre a pour valeur "1", le parent des lignes créées est un élément de parc existant sur une ligne de demande.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

Remarques

Note :

Cette fonction est présente pour des raisons de compatibilité. Son utilisation est déconseillée.

AmAddPOLineToInv()

Cette fonction ajoute une quantité donnée d'élément(s) sur une ligne de commande à une facture fournisseur.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **lPOrdLineId** : Ce paramètre contient l'identifiant de la ligne de commande à ajouter à la facture fournisseur.
- **lInvId** : Ce paramètre contient l'identifiant de la facture fournisseur à laquelle des éléments de la ligne de commande sont ajoutés.
- **dQty** : Ce paramètre contient la quantité d'éléments présents sur la ligne de commande à ajouter à la facture fournisseur.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmAddPOrderLineToReceipt()

Cette fonction permet d'ajouter une ligne de commande à une réception. Vous pouvez ainsi réceptionner une ligne de commande au sein d'une réception existante.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lPOrderLineId** : Ce paramètre contient l'identifiant de la ligne de commande.
- **lRecptId** : Ce paramètre contient l'identifiant de la réception impactée.
- **dQty** : Ce paramètre contient la quantité à réceptionner. Vous pouvez ainsi limiter la quantité réceptionnée par rapport à la quantité commandée (dans l'unité du produit).
- **bCanMerge** : Ce paramètre permet de préciser si oui ou non la ligne peut être fusionnée avec une ligne déja existante dans la reception.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction

AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmAddReceiptLineToInvoice()

Cette fonction permet d'ajouter une ligne de réception à une facture. Vous pouvez ainsi facturer une ligne de réception au sein d'une facture existante.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lRecptLineId** : Ce paramètre contient l'identifiant de la ligne de réception.
- **lInvoiceId** : Ce paramètre contient l'identifiant de la facture impactée.

- **dQty** : Ce paramètre contient la quantité à facturer. Vous pouvez ainsi limiter la quantité facturée par rapport à la quantité reçue (dans l'unité du produit).
- **bCanMerge** : Ce paramètre permet de préciser si oui ou non la ligne peut être fusionnée avec une ligne déja existante dans la facture.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmAddReqLinesToEstim()

Cette fonction ajoute toutes les lignes de demande d'une demande à un devis existant.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lReqId** : Ce paramètre contient l'identifiant de la demande à ajouter au devis.
- **lEstimId** : Ce paramètre contient l'identifiant du devis auquel sont ajoutées toutes les lignes de demande de la demande.
- **bMergeLines** : Ce paramètre permet de préciser si les lignes de demande identiques doivent être combinées (**bMergeLines**=1) pour n'en créer qu'une seule. Les quantités décrites sur les lignes à combiner sont alors ajoutées et une seule ligne est créée.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmAddReqLinesToPO()

Cette fonction ajoute toutes les lignes de demande d'une demande à une commande existante. Le fournisseur précisé dans la demande doit être identique à celui de la commande concernée.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lReqId** : Ce paramètre contient l'identifiant de la demande à ajouter à la commande.
- **lPOrdId** : Ce paramètre contient l'identifiant de la commande à laquelle sont ajoutées les lignes de demande de la demande.
- **bMergeLines** : Ce paramètre permet de préciser si les lignes de demande identiques doivent être combinées (**bMergeLines**=1) pour n'en créer qu'une seule. Les quantités décrites sur les lignes à combiner sont alors ajoutées et une seule ligne est créée.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmAddReqLineToEstim()

Cette fonction ajoute une ligne de demande à un devis existant.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lReqLineId** : Ce paramètre contient l'identifiant de la ligne de demande à ajouter au devis.
- **lEstimId** : Ce paramètre contient l'identifiant du devis auquel est ajoutée la ligne de demande.
- **bMergeLines** : Ce paramètre permet de préciser si les lignes de demande identiques doivent être combinées (**bMergeLines**=1) pour n'en créer qu'une

seule. Les quantités décrites sur les lignes à combiner sont alors ajoutées et une seule ligne est créée.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmAddReqLineToPO()

Cette fonction ajoute une ligne de demande à une commande existante.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **lReqLineId** : Ce paramètre contient l'identifiant de la ligne de demande à ajouter à la commande.
- **lPOrdId** : Ce paramètre contient l'identifiant de la commande à laquelle est ajoutée la ligne de demande.
- **bMergeLines** : Ce paramètre permet de préciser si les lignes de demande identiques doivent être combinées (**bMergeLines**=1) pour n'en créer qu'une seule. Les quantités décrites sur les lignes à combiner sont alors ajoutées et une seule ligne est créée.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmAddRequestLineToPOrder()

Cette fonction permet d'ajouter une ligne de demande à une commande.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lRequestLineId** : Ce paramètre contient l'identifiant de la ligne de demande.
- **lPOrderId** : Ce paramètre contient l'identifiant de la commande impactée.
- **dQty** : Ce paramètre contient la quantité à commander. Vous pouvez ainsi limiter la quantité commandée par rapport à la quantité demandée (dans l'unité du modèle).
- **bCanMerge** : Ce paramètre permet de préciser si oui ou non la ligne peut être fusionnée avec une ligne déja existante dans la commande.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmAddTemplateToPOrder()

Cette fonction permet d'ajouter le contenu complet d'une commande standard à une commande donnée.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lRequestId** : Ce paramètre contient l'identifiant de la ligne de demande à satisfaire par les lignes de commandes qui seront ajoutées.
- **lPOrderId** : Ce paramètre contient l'identifiant de la commande impactée.
- **lTemplateId** : Ce paramètre contient l'identifiant de la commande standard à ajouter.
- **lQty** : Ce paramètre contient la quantité (dans l'unité du produit) à ajouter.

- **bCanMerge** : Ce paramètre permet de préciser si oui ou non la ligne peut être fusionnée avec une ligne déja existante dans la commande.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmAddTemplateToRequest()

Cette fonction permet d'ajouter le contenu complet d'une demande standard à une demande donnée.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **lReqId** : Ce paramètre contient l'identifiant de la ligne de demande impactée.
- **lTemplateId** : Ce paramètre contient l'identifiant de la demande standard à ajouter.
- **lQty** : Ce paramètre contient la quantité (dans l'unité du produit) à ajouter.
- **bCanMerge** : Ce paramètre permet de préciser si oui ou non la ligne peut être fusionnée avec une ligne déja existante dans la demande.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmArchiveRecord()

Cette fonction procède à l'archivage d'un enregistrement de la base de données. L'enregistrement est effacé de sa table d'origine et déplacé vers la table d'archivage correspondante.

Syntaxe API

long AmArchiveRecord(long hApiRecord);

Syntaxe Basic interne

Function AmArchiveRecord(hApiRecord As Long) As Long

Champ d'application

Version : 4.3.0

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiRecord** : Ce paramètre contient le descripteur sur l'enregistrement concerné par l'opération.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

Remarques

Note :

Le traitement des enregistrements liés dépend du type de lien. Dans le cas d'un lien de type OWN, les enregistrements liés sont traités à l'identique. Dans le cas d'un lien DEFINE ou NORMAL, les clés étrangères des enregistrements liés sont remises à 0 et les champs d'archivage sont renseignés avec l'identifiant de l'enregistrement archivé et sa chaîne de description.

IMPORTANT :

Cette fonction n'est disponible que pour un enregistrement provenant d'une table standard.

AmAttribCmdAvailability()

Cette fonction permet de déterminer la disponibilité du bouton d'affectation ou de suppression d'affectation pour un dossier de support.

Syntaxe Basic interne

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

Champ d'application

Version : 4.3.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **bAttrib** : Si vous souhaitez tester la disponibilité du bouton d'affectation, donnez "1" comme valeur à ce paramètre. Si vous souhaitez tester la disponibilité du bouton de suppression d'affectation, donnez "0" comme valeur à ce paramètre.
- **lGroupID** : Ce paramètre contient l'identifiant du groupe de support associé au dossier de support concerné.
- **lInChargeID** : Ce paramètre contient l'identifiant du chargé du dossier de support concerné.
- **bInChargeIsReadOnly** : Ce paramètre vaut "1" si le chargé peut seulement consulter le dossier de support, "0" s'il possède les droits de modification sur le dossier.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmBackupRecord()

Cette fonction effectue une copie de sauvegarde d'un enregistrement. L'enregistrement est copié vers la table d'archivage correspondante sans être effacé de sa table d'origine.

Syntaxe API

long AmBackupRecord(long hApiRecord);

Syntaxe Basic interne

Function AmBackupRecord(hApiRecord As Long) As Long

Champ d'application

Version : 4.3.0

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiRecord** : Ce paramètre contient le descripteur sur l'enregistrement concerné par l'opération.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

Remarques

Note :

Le traitement des enregistrements liés dépend du type de lien. Dans le cas d'un lien de type OWN, les enregistrements liés sont traités à l'identique. Dans le cas d'un lien DEFINE ou NORMAL, les clés étrangères des enregistrements liés sont remises à 0 et les champs d'archivage sont renseignés avec l'identifiant de l'enregistrement archivé et sa chaîne de description.

IMPORTANT :

Cette fonction n'est disponible que pour un enregistrement provenant d'une table standard.

AmBuildNumber()

Cette fonction renvoie le numéro de compilation (build) de l'application.

Syntaxe Basic interne

Function AmBuildNumber() As Long

Champ d'application

Version : 4.3.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmBusinessSecondsInDay()

Calcule le nombre de secondes ouvrées dans une journée en fonction d'un calendrier.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strCalendarSqlName** : Nom SQL du calendrier utilisé pour le calcul.
- **tmDate** : Date à laquelle s'effectue le calcul.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmCalcConsolidatedFeature()

Calcule la valeur d'une caractéristique consolidée sur une table identifiée par son nom SQL.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lCalcFeatId** : Identifiant de la caractéristique consolidée.
- **strSQLTableName** : Nom SQL de la table pour laquelle la caractéristique consolidée est calculée. La caractéristique doit absolument être définie pour cette table.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmCalcDepr()

Cette fonction permet de calculer le montant de l'amortissement sur un bien à une date donnée. Elle renvoie la valeur de l'amortissement à cette date.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **iType** : Ce paramètre permet d'identifier la nature de l'amortissement. Les valeurs possibles de ce paramètre sont les suivantes :
  - 0 : pas d'amortissement
  - 1 : amortissement linéaire
  - 2 : amortissement dégressif
- **lDuration** : Ce paramètre contient la durée sur laquelle porte l'amortissement du bien. Cette durée est exprimée en secondes.
- **dCoeff** : Ce paramètre contient le coefficient appliqué lors du calcul de l'amortissement dégressif. Il n'est pas interprété dans le cas d'un amortissement linéaire mais doit posséder une valeur quelconque.
- **dPrice** : Ce paramètre contient la valeur initiale du bien sur lequel porte le calcul de l'amortissement.
- **tmStart** : Ce paramètre contient la date à partir de laquelle le bien est amorti.
- **tmDate** : Ce paramètre contient la date à laquelle sont évalués l'amortissement et la valeur résiduelle du bien.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmCalculateCatRefQty()

Cette fonction permet de calculer la quantité d'une référence catalogue à commander pour réaliser une demande d'achat.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : ?

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **lSetQty** : Ce paramètre contient la quantité d'éléments dans le produit (par exemple 6, dans le cas d'un produit qui serait 6 bouteilles de 1 litre d'eau).
- **lUseUnitId** : Ce paramètre contient l'identifiant de l'unité utilisée pour le modèle.
- **lPurchUnitId** : Ce paramètre contient l'identifiant de l'unité utilisée pour le produit.
- **strModelDesc** : Ce paramètre contient la description du modèle.

- **strCatRefDesc** : Ce paramètre contient la description de la référence catalogue.
- **strPurchUnit** : Ce paramètre contient la description de l'unité utilisée pour le produit.
- **strUseUnit** : Ce paramètre contient la description de l'unité utilisée pour le modèle.
- **dPkgQty** : Ce paramètre contient la quantité par élément dans le produit (par exemple 1, dans le cas d'un produit qui serait 6 bouteilles de 1 litre d'eau).
- **dUnitConv** : Ce paramètre contient le ratio de conversion des unités pour le produit.
- **dReqLineQty** : Ce paramètre contient la quantité de modèle stipulée dans la demande d'achat.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmCalculateReqLineQty()

Cette fonction permet de calculer la quantité d'un modèle requise pour réaliser une demande d'achat.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : ?

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **lSetQty** : Ce paramètre contient la quantité d'éléments dans le produit (par exemple 6, dans le cas d'un produit qui serait 6 bouteilles de 1 litre d'eau).
- **lUseUnitId** : Ce paramètre contient l'identifiant de l'unité utilisée pour le modèle.
- **lPurchUnitId** : Ce paramètre contient l'identifiant de l'unité utilisée pour le produit.
- **strModelDesc** : Ce paramètre contient la description du modèle.
- **strCatRefDesc** : Ce paramètre contient la description de la référence catalogue.
- **strPurchUnit** : Ce paramètre contient la description de l'unité utilisée pour le produit.
- **strUseUnit** : Ce paramètre contient la description de l'unité utilisée pour le modèle.

- **dPkgQty** : Ce paramètre contient la quantité par élément dans le produit (par exemple 1, dans le cas d'un produit qui serait 6 bouteilles de 1 litre d'eau).
- **dUnitConv** : Ce paramètre contient le ratio de conversion des unités pour le produit.
- **dCatRefQty** : Ce paramètre contient la quantité de modèle stipulée dans la référence catalogue commandée.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmCbkReplayEvent()

Cette fonction permet de rejouer la règle de refacturation à l'origine d'un événement, après avoir corrigé l'enregistrement à l'origine de l'événement.

Syntaxe API

long AmCbkReplayEvent(long hApiCnxBase, long lCbkEventId);

Syntaxe Basic interne

Function AmCbkReplayEvent(lCbkEventId As Long) As Long

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lCbkEventId** : Ce paramètre contient l'identifiant de l'événement de refacturation concerné.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmCheckTraceDone()

L'API AmCheckTraceDone détermine si un port (lPortId) ou un faisceau (lBundleId) est connecté à une chaîne de liaisons existante. La direction de la chaîne de liaisons (iTraceDir) indique si la chaîne de liaisons doit être vérifiée suivant la direction utilisateur vers hôte (iTraceDir = 1) ou hôte vers utilisateur (iTraceDir = 0).

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **lPortId** : ce paramètre est l'identifiant du port a vérifier.
- **lBundleId** : ce paramètre est l'identifiant du faisceau à vérifier.
- **iTraceDir** : ce paramètre précise la direction à vérifier.
  - 1 : Vérifier en direction de l'hôte
  - 0 : Vérifier en direction de l'utilisateur

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmCleanup()

Cette fonction doit être appelée à la fin de tout script utilisant les fonctions de modification de la base de données. Elle libère toutes les ressources utilisées.

Syntaxe API

void AmCleanup();

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	

AmClearLastError()

Cette fonction efface les informations concernant le dernier message d'erreur survenu lors du dernier appel à une fonction.

Syntaxe API

long AmClearLastError(long hApiCnxBase);

Syntaxe Basic interne

Function AmClearLastError() As Long

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmCloseAllChildren()

Cette fonction détruit tous les objets créés lors de la connexion courante.

Syntaxe API

long AmCloseAllChildren(long hApiCnxBase);

Syntaxe Basic interne

Function AmCloseAllChildren() As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	

	Utilisable
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmCloseConnection()

Met un terme à la session AssetCenter pour une connexion donnée. Tous les objets (requêtes, enregistrements, tables, champs...) créés au cours de la session sont automatiquement détruits et tous leurs descripteurs deviennent obsolètes et sont inutilisables. Le descripteur de la connexion n'existe plus.

Syntaxe API

long AmCloseConnection(long hApiCnxBase);

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	

Sortie

- 0 : La fonction s'est exécutée normalement.

- Non nul : Code d'erreur.

AmCommit()

Cette fonction réalise un "Commit" de toutes les modifications apportées à la base de données associée à la connexion.

Syntaxe API

long AmCommit(long hApiCnxBase);

Syntaxe Basic interne

Function AmCommit() As Long

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmComputeAllLicAndInstallCounts()

Cette fonction effectue le décompte des licences et des installations logicielles pour tous les enregistrements.

Syntaxe API

long AmComputeAllLicAndInstallCounts(long hApiCnxBase);

Syntaxe Basic interne

Function AmComputeAllLicAndInstallCounts() As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmComputeLicAndInstallCounts()

Cette fonction effectue le décompte des licences et des installations logicielles pour un enregistrement.

Syntaxe API

long AmComputeLicAndInstallCounts(long hApiCnxBase, long lSLCountId);

Syntaxe Basic interne

Function AmComputeLicAndInstallCounts(lSLCountId As Long) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lSLCountId** : Ce paramètre contient l'identifiant du compteur de licences logicielles.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmConnectionName()

Cette fonction renvoie le nom de la connexion courante à la base de données.

Syntaxe API

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

Syntaxe Basic interne

Function AmConnectionName() As String

Champ d'application

Version : 4.3.0

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
RetVal=amConnectionName()
```

AmConnectTrace()

L'API AmConnectTrace permet de connecter un dispositif/câble source à un dispositif/câble destination et de créer un historique de chaîne de liaisons ainsi qu'une opération sur chaîne de liaisons.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **iSrcLinkType** : ce paramètre détermine le type de chaîne de liaisons pour le dispositif/câble source.
  - 8 : Câble
  - 9 : Dispositif
- **lSrcPortBunId** : ce paramètre est l'identifiant du port ou faisceau à connecter côté source.
- **lSrcLabelRuleId** : ce paramètre est l'identifiant de la règle d'étiquetage utilisée pour la liaison source.
- **iDestLinkType** : ce paramètre détermine le type de chaîne de liaisons pour le dispositif/câble destination.
  - 8 : Câble
  - 9 : Dispositif
- **lDestPortBunId** : ce paramètre est l'identifiant du port ou faisceau à connecter côté destination.
- **lDestLabelRuleId** : ce paramètre est l'identifiant de la règle d'étiquetage utilisée pour la liaison destination.
- **iTraceDir** : ce paramètre indique la direction de la connexion.
  - 1 : de l'utilisateur vers l'hôte
  - 0 : de l'hôte vers l'utilisateur
- **lDutyId** : ce paramètre est l'identifiant de la fonction de la liaison de type câble.
- **strComment** : ce paramètre est l'étiquette de l'opération sur chaîne de liaisons.
- **lCabTraceOutId** : ce paramètre est l'identifiant de compte-rendu de chaîne de liaisons de câble.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.

- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmConvertCurrency()

Cette fonction effectue la conversion entre deux devises à une date donnée.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **tmDate** : Ce paramètre contient la date de conversion. Elle permet de connaître le taux de conversion effectif à cette date.

- **strSrcName** : Ce paramètre contient la devise source de la conversion, c'est-à-dire celle que vous souhaitez convertir.
- **strDstName** : Ce paramètre contient la devise de destination de la conversion, c'est-à-dire celle dans laquelle sera exprimée la devise source.
- **dVal** : Ce paramètre contient le montant (exprimé dans l'unité monétaire de la devise source) à convertir.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Remarques

Note :

Les devises utilisées comme paramètres pour la fonction (**strSrcName** et **strDstName**) doivent impérativement être définies sous AssetCenter. De même un taux de change valide doit exister à la date à laquelle s'effectue la conversion (paramètre **tmDate**).

Exemple

L'exemple suivant effectue la conversion de 5000 FRF en dollars, à la date du 02/11/98.

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

AmConvertDateBasicToUnix()

Cette fonction convertit une date au format Basic (type "Date") en une date au format Unix (type "Long"). Cette fonction est inopérante à partir d'un outil externe car les deux types sont alors équivalents.

Syntaxe API

long AmConvertDateBasicToUnix(long hApiCnxBase, long tmTime);

Syntaxe Basic interne

Function AmConvertDateBasicToUnix(tmTime As Date) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **tmTime** : Ce paramètre contient la date à convertir.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.

- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmConvertDateIntlToUnix()

Cette fonction convertit une date au format international (type "Date") en une date au format Unix (type "Long").

Syntaxe API

long AmConvertDateIntlToUnix(long hApiCnxBase, char *strDate);

Syntaxe Basic interne

Function AmConvertDateIntlToUnix(strDate As String) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strDate** : Ce paramètre contient la date à convertir, au format international (yyyy-mm-dd hh:mm:ss).

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmConvertDateStringToUnix()

Convertit une date au format chaîne (telle qu'elle est affichée dans le "control panel" de Windows) en un "Long" Unix.

Syntaxe API

long AmConvertDateStringToUnix(long hApiCnxBase, char *strDate);

Syntaxe Basic interne

Function AmConvertDateStringToUnix(strDate As String) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **strDate** : Date au format chaîne à convertir.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmConvertDateUnixToBasic()

Cette fonction convertit une date au format Unix (type "Long") en une date au format Basic (type "Date"). Cette fonction est inopérante à partir d'un outil externe car les deux types sont alors équivalents.

Syntaxe API

long AmConvertDateUnixToBasic(long hApiCnxBase, long lTime);

Syntaxe Basic interne

Function AmConvertDateUnixToBasic(lTime As Long) As Date

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **lTime** : Ce paramètre contient la date à convertir.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmConvertDateUnixToIntl()

Cette fonction convertit une date au format Unix (type "Long") en une date au format international (yyyy-mm-dd hh:mm:ss).

Syntaxe API

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

Syntaxe Basic interne

Function AmConvertDateUnixToIntl(lUnixDate As Long) As String

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **lUnixDate** : Ce paramètre contient la date à convertir.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmConvertDateUnixToString()

Convertit une date au format "Long" Unix en une date au format chaîne (telle qu'elle est affichée dans le "control panel" de Windows).

Syntaxe API

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

Syntaxe Basic interne

Function AmConvertDateUnixToString(lUnixDate As Long) As String

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **lUnixDate** : Date au format "Long" Unix à convertir.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmConvertDoubleToString()

Cette fonction convertit un nombre en double précision en une chaîne de caractères. La chaîne est formatée conformément aux options régionales (de nombre) définies dans le panneau de contrôle de Windows.

Syntaxe API

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

Syntaxe Basic interne

Function AmConvertDoubleToString(dSrc As Double) As String

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **dSrc** : Ce paramètre contient le nombre en double précision à convertir.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmConvertMonetaryToString()

Cette fonction convertit une valeur monétaire en une chaîne de caractères. La chaîne est formatée conformément aux options régionales (monétaires) définies dans le panneau de contrôle de Windows.

Syntaxe API

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

Syntaxe Basic interne

Function AmConvertMonetaryToString(dSrc As Double) As String

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **dSrc** : Ce paramètre contient la valeur monétaire à convertir.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.

- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmConvertStringToDouble()

Cette fonction convertit une chaîne de caractères (dans un format conforme à celui défini dans le panneau de contrôle de Windows) en un nombre en double précision.

Syntaxe API

double AmConvertStringToDouble(char *strSrc);

Syntaxe Basic interne

Function AmConvertStringToDouble(strSrc As String) As Double

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strSrc** : Ce paramètre contient la chaîne de caractères à convertir.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmConvertStringToMonetary()

Cette fonction convertit une chaîne de caractères (dans un format conforme à celui défini dans le panneau de contrôle de Windows) en une valeur monétaire.

Syntaxe API

double AmConvertStringToMonetary(char *strSrc);

Syntaxe Basic interne

Function AmConvertStringToMonetary(strSrc As String) As Double

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **strSrc** : Ce paramètre contient la chaîne de caractères à convertir.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmCounter()

Cette fonction renvoie la valeur du compteur **strCounterName** incrémentée de 1. Des zéros sont ajoutés en préfixe si **iWidth** est supérieur au nombre de chiffres du compteur. Si le compteur comporte plus de chiffres que la valeur stockée dans **iWidth**, le résultat renvoyé par la fonction n'est en aucun cas tronqué.

Syntaxe Basic interne

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

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	

	Utilisable
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strCounterName** : Nom du compteur tel qu'il est défini sous AssetCenter (accès par le menu **Administration/ Compteurs**).
- **iWidth** : La valeur de ce paramètre force le formatage du résultat de la fonction, en l'exprimant sur n chiffres. Ce paramètre n'a de sens que dans le cas où la taille du compteur est inférieure à la valeur de ce paramètre.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Remarques

Note :

Si cette fonction est utilisée dans une action de type Script, il faut impérativement préciser un contexte pour l'action. Dans le cas contraire, une erreur est générée.

Exemple

L'exemple suivant renvoie la valeur du compteur "BonsLivraison" exprimée sur 5 chiffres :

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

Si par exemple le compteur "BonsLivraison" vaut "18", le résultat est le suivant :

```
00019
```

AmCreateAssetPort()

L'API AmCreateAssetPort crée un nouveau port pour un dispositif (lAssetId). Le nouveau port contient le nombre donné de broches (iPinCount) du type de connecteur de câble donné (lCabCnxTypeId). L'état des broches doit être "Disponible". Les broches qui seront ajoutées au port seront triées par numéro de séquence. Suivant la direction du port (bPinPortDir), les broches disponibles sont triées par ordre croissant (bPinPortDir=0) or décroissant (bPinPortDir=1). cette fonction assigne la fonction (lDutyId) donnée au nouveau port.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lAssetId** : ce paramètre est l'identifiant du dispositif.
- **lCabCnxTypeId** : ce paramètre est l'identifiant du type de connexion de câble.
- **lDutyId** : ce paramètre est l'identifiant du type de fonction du port.
- **iPinCount** : ce paramètre est le nombre de broches qui seront utilisées dans le nouveau port.
- **bPinPortDir** : ce paramètre précise la direction du port.
- **iConnStatus**
- **bConsecutivePins**
- **iPrevPinSeq**
- **bLogError**

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmCreateAssetsAwaitingDelivery()

Cette fonction permet de créer les biens qui sont en attente de réception

Syntaxe API

long AmCreateAssetsAwaitingDelivery(long hApiCnxBase, long lPOrdId);

Syntaxe Basic interne

Function AmCreateAssetsAwaitingDelivery(lPOrdId As Long) As Long

Champ d'application

Version : 3.61

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lPOrdId** : Ce paramètre contient l'identifiant de la commande concernée

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmCreateCable()

L'API AmCreateCable crée un nouveau câble. Le câble est créé suivant un modèle donné (lModelId), le rôle du câble (strCableRole), sa règle d'étiquetage (lLabelRuleId), sa localisation utilisateur (lUserId), et sa localisation hôte (lHostId). Si le projet (lProjectId) et l'intervention (lWorkOrderId) prennent

des valeurs, le nouveau câble est ajouté au projet et à l'intervention avec son commentaire (strComment). ce commentaire décrit l'action qui sera accomplie sur le câble. (i.e. "Mettre en place un nouveau câble").

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lModelId** : ce paramètre est l'identifiant du modèle de câble.
- **lUserId** : ce paramètre est l'identifiant de la localisation côté utilisateur.
- **lHostId** : ce paramètre est l'identifiant de la localisation côté hôte.
- **strCableRole** : ce paramètre définit le rôle du câble.
- **lProjectId** : ce paramètre est le projet associé à la mise en place du câble.

- **lWorkOrderId** : ce paramètre est l'identifiant de l'intervention associée à la mise en place du câble.
- **strComment** : ce paramètre est le commentaire qui sera joint à l'intervention (précisée par lWorkOrderId).
- **lLabelRuleId** : ce paramètre contient l'identifiant de la règle d'étiquetage qui sera appliquée lors de la création de l'étiquette pour le câble.
- **strLabel** : ce paramètre précise l'étiquette apposée sur le câble.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmCreateCableBundle()

L'API AmCreateCableBundle crée un nouveau faisceau pour un câble donné (lCableId). Le nouveau faisceau contient le nombre donné de paires de câble (iPairCount) du type donné de paire de câble (lPairType). L'état des paires doit être "Disponible". Cette fonction assigne la fonction donnée (lDutyId) au nouveau faisceau.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lCableId** : ce paramètre est l'identifiant du câble (cet identifiant doit exister dans la table des câbles).
- **lPairTypeId** : ce paramètre est l'identifiant du type de paire de câble.
- **lDutyId** : ce paramètre est l'identifiant de la fonction du faisceau.
- **iPairCount** : ce paramètre définit le nombre de paires du faisceau.
- **iStartPairSeq**
- **bLogError**

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction

AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmCreateCableLink()

L'API AmCreateCableLink crée une nouvelle liaison câble pour un câble (lCableId) et faisceau (lBundleId) donnés. La fonction de la liaison câble est établie grâce à la fonction donnée (lDutyId). La règle d'étiquetage de la liaison câble est établie grâce à la règle d'étiquetage donnée (lLabelRuleId).

Note :

L'étiquette n'est pas mise à jour grâce à la règle d'étiquetage donnée, AmRefreshLabel() doit être appelée séparément.

Si une liaison précédente (lPrevLinkId) est spécifiée, une liaison parente est créée entre les deux enregistrements pour lesquels la liaison précédente est la liaison fille.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓

	Utilisable
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lCableId** : ce paramètre est l'identifiant du câble pour la connexion.
- **lDutyId** : ce paramètre est l'identifiant de la fonction de connexion.
- **lBundleId** : ce paramètre est l'identifiant du faisceau de câble à connecter.
- **lPrevLinkId** : ce paramètre définit l'identifiant de la liaison câble utilisée par la connexion. Une valeur de 0 le rend facultatif.
- **iTraceDir** : ce paramètre définit la direction de la connexion.
  - 0=hôte vers utilisateur
  - 1=utilisateur vers hôte
- **lLabelRuleId** : ce paramètre est l'identifiant de la règle d'étiquetage utilisée.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmCreateDelivFromPO()

Cette fonction effectue la réception d'une commande à partir d'une commande et renvoie l'identifiant de la fiche de réception créée.

Syntaxe API

long AmCreateDelivFromPO(long hApiCnxBase, long lPOrdId);

Syntaxe Basic interne

Function AmCreateDelivFromPO(lPOrdId As Long) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lPOrdId** : Ce paramètre contient l'identifiant de la commande à réceptionner.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmCreateDevice()

L'API AmCreateDevice crée un nouveau dispositif. Le dispositif est créé grâce au modèle (lModelId) et localisation (lLocationId) donnés. La règle d'étiquetage dépend de la règle donnée (lLabelRuleId).

Note :

L'étiquette n'est pas mise à jour grâce à la règle d'étiquetage donnée, AmRefreshLabel() doit être appelée séparément.

Si le projet (lProjectId) et l'intervention (lWorkOrderId) prennent des valeurs, le nouveau bien est ajouté au projet et à l'intervention avec le commentaire contenu dans strComment. Ce commentaire décrit l'action effectuée sur le bien (i.e. "Installer un nouveau bien").

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **lModelId** : ce paramètre contient l'identifiant du modèle du nouveau dispositif.
- **lLocationId** : ce paramètre contient l'identifiant de la localisation du nouveau dispositif.
- **lProjectId** : ce paramètre contient l'identifiant du projet. Il peut prendre la valeur 0.
- **lWorkOrderId** : ce paramètre contient l'identifiant de l'intervention. Il peut prendre la valeur 0.
- **lLabelRuleId** : ce paramètre contient l'identifiant de la règle d'étiquetage qui sera utilisée par le bien.
- **strComment** : ce paramètre est le commentaire qui sera joint à l'intervention.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmCreateDeviceLink()

L'API AmCreateDeviceLink crée une liaison câble de type dispositif pour un dispositif (lAssetId) et un port (lPortId) donnés. La règle d'étiquetage de la liaison câble est établie grâce à la règle d'étiquetage donnée (lLabelRuleId).

Note :

L'étiquette n'est pas mise à jour grâce à la règle d'étiquetage donnée, AmRefreshLabel() doit être appelée séparément.

Si une liaison précédente (lPrevLinkId) est spécifiée, une liaison parente est créée entre les deux enregistrements. Si la direction de la chaîne de liaisons est utilisateur vers hôte (iTraceDir = 1), alors la liaison précédente est fille. Si la direction de la chaîne de liaisons est hôte vers utilisateur (iTraceDir = 0), alors la liaison précédente est parent.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lAssetId** : ce paramètre contient l'identifiant du bien qui sera connecté.

- **lPortId** : ce paramètre contient l'identifiant du port qui sera connecté.
- **lPrevLinkId** : ce paramètre contient l'identifiant de la liaison du dispositif permettant la connexion.
- **iTraceDir** : ce paramètre précise la direction de la connexion.
  - 0=hôte vers utilisateur
  - 1=utilisateur vers hôte
- **lLabelRuleId** : ce paramètre contient l'identifiant de la règle d'étiquetage utilisée pour la nouvelle connexion.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmCreateEstimFromReq()

Cette fonction effectue la création d'une devis à partir d'une demande d'achat et renvoie l'identifiant du devis créé.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lReqId** : Ce paramètre contient l'identifiant de la demande d'achat qui sert à la création du devis.
- **lSuppId** : Ce paramètre contient l'identifiant du fournisseur du devis qui sera créé à l'issue de l'exécution de la fonction.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmCreateEstimsFromAllReqLines()

Cette fonction crée un devis à partir d'une demande et renvoie l'identifiant du devis créé.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lReqId** : Ce paramètre contient l'identifiant de la demande à l'origine du devis.
- **bMergeLines** : Ce paramètre permet de préciser si les lignes de demande identiques doivent être combinées (**bMergeLines**=1) pour n'en créer qu'une seule. Les quantités décrites sur les lignes à combiner sont alors ajoutées et une seule ligne est créée.
- **lDefSuppId** : Ce paramètre contient l'identifiant du fournisseur par défaut pour le devis.

Sortie

- 0 : La fonction s'est exécutée normalement.

- Non nul : Code d'erreur.

AmCreateInvFromPO()

Cette fonction crée une facture fournisseur à partir d'une commande et renvoie l'identifiant de la facture fournisseur créée.

Syntaxe API

long AmCreateInvFromPO(long hApiCnxBase, long lPOrdId);

Syntaxe Basic interne

Function AmCreateInvFromPO(lPOrdId As Long) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lPOrdId** : Ce paramètre contient l'identifiant de la commande à l'origine de la facture.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmCreateLink()

Cette fonction modifie un lien d'un enregistrement et le fait pointer vers un nouvel enregistrement (**hApiRecDest**) dans la table de destination. Elle crée donc un lien entre deux enregistrements.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	

	Utilisable
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiRecord** : Ce paramètre contient le descripteur sur l'enregistrement contenant le lien à modifier.
- **strLinkName** : Ce paramètre contient le nom SQL du lien à modifier.
- **hApiRecDest** : Ce paramètre contient un descripteur sur l'enregistrement de destination du lien.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmCreateOrUpdateInvoiceFromReceipt()

Cette fonction permet de créer ou de mettre à jour une facture à partir d'une fiche de réception.

Syntaxe API

long AmCreateOrUpdateInvoiceFromReceipt(long hApiCnxBase, long lRecptId);

Syntaxe Basic interne

Function AmCreateOrUpdateInvoiceFromReceipt(lRecptId As Long) As Long

Champ d'application

Version : ?

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **lRecptId** : Ce paramètre contient l'identifiant de la facture concernée par l'opération.

Sortie

La fonction renvoie l'identifiant de la facture créée.

Remarques

Note :

La mise à jour n'est possible qu'en appelant cette fonction par un outil externe.

AmCreatePOFromEstim()

Cette fonction effectue la création d'une commande à partir d'un devis et renvoie l'identifiant de la commande créée.

Syntaxe API

long AmCreatePOFromEstim(long hApiCnxBase, long lEstimId);

Syntaxe Basic interne

Function AmCreatePOFromEstim(lEstimId As Long) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lEstimId** : Ce paramètre contient l'identifiant du devis qui sert à la création de la commande.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmCreatePOFromReq()

Cette fonction effectue la création d'une commande à partir d'une demande d'achat et renvoie l'identifiant de la commande créée.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lReqId** : Ce paramètre contient l'identifiant de la demande d'achat qui sert à la création de la commande.
- **lSuppId** : Ce paramètre contient l'identifiant du fournisseur de la commande qui sera créée à l'issue de l'exécution de la fonction.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmCreatePOrderFromRequest()

Cette fonction permet de créer une commande à partir d'une demande.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lRequestId** : Ce paramètre contient l'identifiant de la demande concernée.
- **lSupplierId** : Ce paramètre contient l'identifiant du fournisseur pour la commande.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmCreatePOrdersFromRequest()

Cette fonction permet de créer toutes les commandes necessaires à la satisfaction d'une demande donnée.

Syntaxe API

long AmCreatePOrdersFromRequest(long hApiCnxBase, long lRequestId);

Syntaxe Basic interne

Function AmCreatePOrdersFromRequest(lRequestId As Long) As Long

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **lRequestId** : Ce paramètre contient l'identifiant de la demande concernée

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmCreatePOsFromAllReqLines()

Cette fonction crée toutes les commandes à partir des lignes de demande d'une demande.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	

	Utilisable
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lReqId** : Ce paramètre contient l'identifiant de la demande à partir de laquelle les commandes vont être créées.
- **bMergeLines** : Ce paramètre permet de préciser si les lignes de demande identiques doivent être combinées (**bMergeLines**=1) pour n'en créer qu'une seule. Les quantités décrites sur les lignes à combiner sont alors ajoutées et une seule ligne est créée.
- **lDefSuppId** : Ce paramètre contient l'identifiant du fournisseur par défaut des éléments demandés. Ce paramètre est optionnel et possède "0" comme valeur par défaut..

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmCreateProjectCable()

L'API AmCreateProjectCable ajoute un câble (lCableId) à un projet (lProjectId) et une intervention (lWorkOrderId). Un commentaire (strComment) explique l'action effectuée (i.e. "Installer un nouveau câble").

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lProjectId** : ce paramètre contient l'identifiant du projet qui obtient le nouveau câble.
- **lWorkOrderId** : ce paramètre contient l'identifiant de l'intervention sur le câble.
- **lCableId** : ce paramètre contient l'identifiant du câble.
- **strComment** : ce paramètre est le commentaire qui sera joint à l'intervention.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmCreateProjectDevice()

L'API AmCreateProjectDevice ajoute un dispositif (lAssetId) à un projet (lProjectId) et une intervention (lWorkOrderId). Un commentaire (strComment) explique l'action effectuée (i.e. "Installer un nouveau dispositif").

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lProjectId** : ce paramètre contient l'identifiant du projet qui obtient le nouveau dispositif.
- **lWorkOrderId** : ce paramètre contient l'identifiant de l'intervention qui obtient le nouveau dispositif.

- **lAssetId** : ce paramètre contient l'identifiant du nouveau dispositif en tant que bien.
- **strComment** : ce paramètre est le commentaire qui sera joint à l'intervention.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmCreateProjectTrace()

L'API AmCreateProjectTrace ajoute une chaîne de liaisons (strTrace) à un projet (lProjectId) et une intervention (lWorkOrderId). La fonction de la chaîne de liaisons est établie suivant la fonction donnée (lDutyId). Le type de chaîne de liaisons (iTraceType) indique si la chaîne de liaisons est une connexion (lTraceType = 1) ou une déconnexion (lTraceType = 2). L'étiquette de la liaison utilisateur modifiée (strModLinkLabel) identifie quelle partie de la chaîne de liaisons est modifiée. Un commentaire (strComment) explique l'action effectuée (i.e. "Connecter ces dispositifs").

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lProjectId** : ce paramètre contient l'identifiant du projet qui obtient l'information de la chaîne de liaisons.
- **lWorkOrderId** : ce paramètre contient l'identifiant de l'intervention qui obtient l'information de la chaîne de liaisons.
- **iTraceType** : ce paramètre précise le type de chaîne de liaisons.
  - 1=connexion
  - 2=déconnexion
- **lDutyId** : ce paramètre contient l'identifiant de la fonction. Elle apparaît dans une intervention.
- **strModLinkLabel** : ce paramètre est un commentaire qui sera utilisé pour l'intervention.
- **strTrace** : ce paramètre est la chaîne de compte-rendu de la chaîne de liaisons qui sera utilisée pour l'intervention.
- **strComment** : ce paramètre est le commentaire qui sera joint à l'intervention.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmCreateReceiptFromPOrder()

Cette fonction permet de créer une reception à partir d'une commande.

Syntaxe API

long AmCreateReceiptFromPOrder(long hApiCnxBase, long lPOrderId);

Syntaxe Basic interne

Function AmCreateReceiptFromPOrder(lPOrderId As Long) As Long

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **lPOrderId** : Ce paramètre contient l'identifiant de la commande concernée.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmCreateRecord()

Cette fonction créée un enregistrement vide dans une table en tenant compte des valeurs par défaut. Ce nouvel enregistrement ne possède aucune existence dans la base de données tant qu'il n'a pas été inséré.

Syntaxe API

long AmCreateRecord(long hApiCnxBase, char *strTable);

Syntaxe Basic interne

Function AmCreateRecord(strTable As String) As Long

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **strTable** : Ce paramètre contient le nom SQL de la table dans laquelle vous souhaitez créer l'enregistrement.

Exemple

L'exemple suivant crée une personne dans la base de données :

```
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()

Cette fonction permet de créer tous les objets d'un cycle d'achat : Demande, Commande, Réception, Facture.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **dQty** : Ce paramètre contient la quantité (en unités de condionnement) à commander, réceptionner, puis facturer.
- **lCatRefId** : Ce paramètre contient l'identifiant de la référence catalogue.
- **dUnitPrice** : Ce paramètre contient le prix unitaire de la référence catalogue.
- **strCur** : Ce paramètre contient le code de la devise pour le prix de la référence catalogue.
- **lRequesterId** : Ce paramètre contient l'identifiant du demandeur.
- **lCostId** : Ce paramètre contient l'identifiant du centre de coût impacté.
- **lUserId** : Ce paramètre contient l'identifiant de l'utilisateur de l'élément commandé.
- **lStockId** : Ce paramètre contient l'identifiant du stock de livraison de l'élément.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Remarques

Equivalant à la séquence d'appels : amCreateRequestToReceipt, amCreateOrUpdateInvoiceFromReceipt.

AmCreateRequestToPOrder()

Cette fonction permet de créer les objets d'un cycle d'achat : Demande, Commande.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓

	Utilisable
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **dQty** : Ce paramètre contient la quantité (en unités de condionnement) à commander.
- **lCatRefId** : Ce paramètre contient l'identifiant de la référence catalogue.
- **dUnitPrice** : Ce paramètre contient le prix unitaire de la référence catalogue.
- **strCur** : Ce paramètre contient le code de la devise pour le prix de la référence catalogue.
- **lRequesterId** : Ce paramètre contient l'identifiant du demandeur.
- **lCostId** : Ce paramètre contient l'identifiant du centre de coût impacté.
- **lUserId** : Ce paramètre contient l'identifiant de l'utilisateur de l'élément commandé.
- **lStockId** : Ce paramètre contient l'identifiant du stock de livraison de l'élément.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmCreateRequestToReceipt()

Cette fonction permet de créer les objets d'un cycle d'achat : Demande, Commande, Reception.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **dQty** : Ce paramètre contient la quantité (en unités de condionnement) à commander, puis réceptionner.
- **lCatRefId** : Ce paramètre contient l'identifiant de la référence catalogue.
- **dUnitPrice** : Ce paramètre contient le prix unitaire de la référence catalogue.

- **strCur** : Ce paramètre contient le code de la devise pour le prix de la référence catalogue.
- **lRequesterId** : Ce paramètre contient l'identifiant du demandeur.
- **lCostId** : Ce paramètre contient l'identifiant du centre de coût impacté.
- **lUserId** : Ce paramètre contient l'identifiant de l'utilisateur de l'élément commandé.
- **lStockId** : Ce paramètre contient l'identifiant du stock de livraison de l'élément.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Remarques

Equivalant à la séquence d'appels : amCreateRequestToPOrder, amCreateReceiptFromPOrder.

AmCreateReturnFromReceipt()

Cette fonction permet de créer une fiche de retour à partir d'une fiche de réception.

Syntaxe API

long AmCreateReturnFromReceipt(long hApiCnxBase, long lRecptId);

Syntaxe Basic interne

Function AmCreateReturnFromReceipt(lRecptId As Long) As Long

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lRecptId** : Ce paramètre contient l'identifiant de la réception.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmCreateTraceHist()

L'API AmCreateTraceHist sert à créer l'historique de chaîne de liaisons et l'opération sur chaîne de liaisons à partir d'une connexion existante depuis un dispositif/câble source vers un dispositif/câble destination.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lSrcLinkId** : ce paramètre contient l'identifiant de la liaison attribué à la liaison source.
- **lDestLinkId** : ce paramètre contient l'identifiant de la liaison attribué à la liaison destination.
- **iTraceDir** : ce paramètre précise la direction de la connexion.
  - 0=hôte vers utilisateur
  - 1=utilisateur vers hôte
- **lCabTraceOutId** : ce paramètre contient l'identifiant de compte-rendu de chaîne de liaisons de câble.
- **strComment** : ce paramètre est le commentaire qui sera associé à l'opération de chaîne de liaisons.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmCreateTraceLink()

Cette fonction permet de créer une liaison entre des dispositifs de câblage.

Syntaxe Basic interne

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

Champ d'application

Version : 4.3.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **iLinkType** : Ce paramètre permet d'identifier le type d'élément pris en compte ("1" pour un dispositif de câblage, "0" pour un câble).
- **lAstCabId** : Ce paramètre contient l'identifiant du bien associé au dispositif de câblage.

- **lPrtBunId** : Ce paramètre contient l'identifiant de l'enregistrement concerné par l'opération. Cet identifiant est pris dans la table amCableBundle pour un câble ou dans la table amPort pour un dispositif de câblage.
- **lPrevLinkId** : Ce paramètre contient l'identifiant de l'élément servant de point de départ à la liaison.
- **iTraceDir** : Ce paramètre permet de préciser le sens de la liaison. Il peut prendre les valeurs "HOST_TO_USER" ou "USER_TO_HOST".
- **lDutyId** : Ce paramètre contient l'identifiant de la fonction de la liaison.
- **lLabelRuleId** : Ce paramètre contient l'identifiant de la règle d'étiquettage de la liaison (par défaut, cette valeur est nulle).

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmCryptPassword()

Cette fonction crypte le mot de passe d'un utilisateur, identifié par son login et son mot de passe.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strUser** : Ce paramètre contient le login de l'utilisateur dont vous souhaitez crypter le mot de passe.
- **strPasswd** : Ce paramètre contient, en clair, le mot de passe à crypter.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmCurrentDate()

Cette fonction renvoie la date courante sur le poste client.

Syntaxe API

long AmCurrentDate();

Syntaxe Basic interne

Function AmCurrentDate() As Date

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Remarques

Si la base est configurée pour utiliser les fuseaux horaires, le comportement de cette fonction est différent suivant qu'elle est appelée directement sous AssetCenter ou par le biais d'un programme externe. Sous AssetCenter, cette fonction a un comportement identique à la fonction Basic Now(). A partir d'un programme externe, la valeur retournée par cette fonction est rapportée au fuseau horaire GMT+0 sans décalage pour l'heure d'été.

AmCurrentIsoLang()

Cette fonction renvoie, conformément à la norme ISO, la langue utilisée dans AssetCenter ("fr" pour le français, "en" pour l'anglais,...).

Syntaxe API

long AmCurrentIsoLang(char *pstrLanguage, long lLanguage);

Syntaxe Basic interne

Function AmCurrentIsoLang() As String

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmCurrentLanguage()

Cette fonction renvoie la langue utilisée dans AssetCenter ("FR" pour le français, "US" pour l'anglais,...).

Syntaxe API

long AmCurrentLanguage(char *pstrLanguage, long lLanguage);

Syntaxe Basic interne

Function AmCurrentLanguage() As String

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmCurrentServerDate()

Cette fonction renvoie la date courante sur le serveur.

Syntaxe API

long AmCurrentServerDate(long hApiCnxBase);

Syntaxe Basic interne

Function AmCurrentServerDate() As Date

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmDaDepAddComputers()

Cette fonction permet de rajouter des cibles de déploiement à une instance de déploiement existante.

Syntaxe API

long AmDaDepAddComputers(long hApiCnxBase, long lInstanceId, char *strSelection);

Syntaxe Basic interne

Function AmDaDepAddComputers(lInstanceId As Long, strSelection As String) As Long

Champ d'application

Version : 4.2.0

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lInstanceId** : Ce paramètre contient l'identifiant de l'instance de déploiement concernée par l'opération.
- **strSelection** : Ce paramètre contient la liste des identifiants (séparés par des virgules) des ordinateurs ou groupes d'ordinateurs à rajouter à l'instance de déploiement.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
' This sample will create a deployment instance on all computers belog
ing to a given department
Dim hqComp As Long
Dim hr As Long
Dim lErr As Long
Dim lCompId As Long
Dim lDepInstanceId As Long

' Create the Deployment Instance with no associated computers, and in
suspended mode
lDepInstanceId = AmDaDepCreateInstance( "Description", _
                              lServerId, _
                              lWfSchemeId, _
                              "", _
                              {pgWorkflow.cbtStart.Value}, _
                              FALSE )

' Create a query on all computers belonging to employees for the strDe
partFullName department
hqComp = AmQueryCreate()
lErr = AmQueryExec( hqComp, "SELECT lComputerId FROM amComputer WHERE
Portfolio.User.FullName LIKE "  AmSqlTextConst( "/Company/Department1/
%") )

While lErr = 0
  lCompId = AmGetFieldLongValue( hqComp, 0 )
  ' Add the computer id to the instance
  lErr = AmDaDepAddComputers( lDepInstanceId, CStr(lCompId) )
  ' And loop until done
  lErr = AmQueryNext( hqComp )
Wend

' And now that all target computers have been added, start the instanc
e
hr = AmGetRecordFromMainId("amDaDepInstance", lDepInstanceId )
```

```
lErr = AmSetFieldLongValue(hr, "seRequest", 1)
lErr = AmUpdateRecord(hr)
```

AmDaDepCopyInstance()

Cette fonction permet de copier une instance de déploiement.

Syntaxe API

long AmDaDepCopyInstance(long hApiCnxBase, long lInstanceId, char *strStatusValues, long bStart);

Syntaxe Basic interne

Function AmDaDepCopyInstance(lInstanceId As Long, strStatusValues As String, bStart As Long) As Long

Champ d'application

Version : 4.2.0

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lInstanceId** : Ce paramètre contient l'identifiant de l'instance de déploiement à copier.
- **strStatusValues** : Ce paramètre contient la liste des états de déploiement pris en compte dans la copie de l'instance. Vous pouvez ainsi copier

uniquement les cibles pour lesquels l'état de déploiement est "Echec" ou "En attente" pour l'instance de déploiement à copier. Les états de déploiement sont identifiés par leur valeur associée dans l'énumération système (0: En attente; 1: En cours; 2: Succès; 3: Echec).

- **bStart** : Lorsque ce paramètre a pour valeur 1, l'instance de déploiement copiée est démarrée immédiatement.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

L'exemple suivant démarre une instance de déploiement sur des cibles en attente ou en échec, en réalisant une copie de l'instance de déploiement d'origine :

```
dim hr as Long

hr = 0

If [lDaDepInstanceId]
```

AmDaDepCreateInstance()

Cette fonction permet la création d'une instance de déploiement.

Syntaxe API

long AmDaDepCreateInstance(long hApiCnxBase, char *strDesc, long lServerId, long lWfSchemeId, char *strSelection, long tmStart, long bStart);

Syntaxe Basic interne

Function AmDaDepCreateInstance(strDesc As String, lServerId As Long, lWfSchemeId As Long, strSelection As String, tmStart As Date, bStart As Long) As Long

Champ d'application

Version : 4.2.0

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **strDesc** : Ce paramètre contient une description pour l'instance de déploiement.
- **lServerId** : Ce paramètre contient l'identifiant du serveur de déploiement à utiliser. Si vous laissez ce paramètre vide, la fonction utilisé, s'il existe, le serveur pour lequel l'option "Serveur par défaut" est sélectionnée.
- **lWfSchemeId** : Ce paramètre contient l'identifiant du workflow de déploiement utilisé pour créer l'instance.
- **strSelection** : Ce paramètre contient la liste des identifiants (séparés par des virgules) des ordinateurs ou groupes d'ordinateurs ciblés par le déploiement.
- **tmStart** : Ce paramètre contient la date de démarrage par le serveur de l'instance de déploiement.
- **bStart** : Lorsque ce paramètre a pour valeur 1, l'instance de déploiement est démarrée immédiatement.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
' This sample will create a deployment instance on all computers belog
ing to a given department
Dim hqComp As Long
Dim hr As Long
Dim lErr As Long
Dim lCompId As Long
Dim lDepInstanceId As Long

' Create the Deployment Instance with no associated computers, and in
suspended mode
lDepInstanceId = AmDaDepCreateInstance( "Description", _
                              lServerId, _
                              lWfSchemeId, _
                              "", _
                              {pgWorkflow.cbtStart.Value}, _
                              FALSE )

' Create a query on all computers belonging to employees for the strDe
partFullName department
hqComp = AmQueryCreate()
lErr = AmQueryExec( hqComp, "SELECT lComputerId FROM amComputer WHERE
Portfolio.User.FullName LIKE "  AmSqlTextConst( "/Company/Department1/
%") )

While lErr = 0
  lCompId = AmGetFieldLongValue( hqComp, 0 )
  ' Add the computer id to the instance
  lErr = AmDaDepAddComputers( lDepInstanceId, CStr(lCompId) )
  ' And loop until done
  lErr = AmQueryNext( hqComp )
Wend

' And now that all target computers have been added, start the instanc
e
hr = AmGetRecordFromMainId("amDaDepInstance", lDepInstanceId )
```

```
lErr = AmSetFieldLongValue(hr, "seRequest", 1)
lErr = AmUpdateRecord(hr)
```

AmDateAdd()

Cette fonction calcule une nouvelle date en fonction d'une date de départ à laquelle est ajoutée une durée réelle.

Syntaxe API

long AmDateAdd(long tmStart, long tsDuration);

Syntaxe Basic interne

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

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **tmStart** : Ce paramètre contient la date à laquelle sera ajoutée une durée.
- **tsDuration** : Ce paramètre contient la durée, exprimée en secondes, à ajouter à la date **tmStart**.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

L'exemple ci-dessous, illustre la différence entre la fonction amDateAdd() et la fonction amDateAddLogical(). Une durée de 30 jours sera ajoutée à la date 1/1/1999 (1er janvier 1999) au moyen de chacune des fonctions.

AmDateAdd ajoute une durée réelle, soit 30 jours dans le cas présent :

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

La fonction renvoie la valeur :

```
1999/01/31
```

AmDateAddLogical ajoute une durée logique, soit 30 jours (=1 mois) dans le cas présent :

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

La fonction renvoie la valeur :

```
1999/02/01
```

AmDateAddLogical()

Cette fonction calcule une nouvelle date en fonction d'une date de départ à laquelle est ajoutée une durée logique (un mois comporte 30 jours).

Syntaxe API

long AmDateAddLogical(long tmStart, long tsDuration);

Syntaxe Basic interne

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

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **tmStart** : Ce paramètre contient la date à laquelle sera ajoutée une durée.
- **tsDuration** : Ce paramètre contient la durée, exprimée en secondes, à ajouter à la date **tmStart**.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

L'exemple ci-dessous, illustre la différence entre la fonction amDateAdd() et la fonction amDateAddLogical(). Une durée de 30 jours sera ajoutée à la date 1/1/1999 (1er janvier 1999) au moyen de chacune des fonctions.

AmDateAdd ajoute une durée réelle, soit 30 jours dans le cas présent :

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

La fonction renvoie la valeur :

```
1999/01/31
```

AmDateAddLogical ajoute une durée logique, soit 30 jours (=1 mois) dans le cas présent :

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

La fonction renvoie la valeur :

```
1999/02/01
```

AmDateDiff()

Cette fonction calcule en secondes la durée écoulée entre deux dates.

Syntaxe API

long AmDateDiff(long tmEnd, long tmStart);

Syntaxe Basic interne

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

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓

	Utilisable
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **tmEnd** : Ce paramètre contient la date de fin de la période sur laquelle est effectué le calcul.
- **tmStart** : Ce paramètre contient la date de début de la période sur laquelle est effectué le calcul.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

L'exemple suivant calcule la durée écoulée entre le 01/01/98 et le 01/01/99.

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

AmDbExecAql()

Cette fonction permet d'exécuter une requête AQL sur la base de données.

Syntaxe API

long AmDbExecAql(long hApiCnxBase, char *strAqlStatement);

Syntaxe Basic interne

Function AmDbExecAql(strAqlStatement As String) As Long

Champ d'application

Version : 4.1.0

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **strAqlStatement** : Ce paramètre contient la requête AQL à exécuter.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmDbGetDate()

Cette fonction renvoie le résultat au format date, de l'exécution d'une requête AQL sans curseur. Si la requête ne renvoie aucun résultat, la valeur 0 est retournée sans déclencher d'erreur.

Syntaxe API

long AmDbGetDate(long hApiCnxBase, char *strQuery);

Syntaxe Basic interne

Function AmDbGetDate(strQuery As String) As Date

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strQuery** : Ce paramètre contient l'intégralité de la requête AQL dont on veut récupérer le résultat.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmDbGetDouble()

Cette fonction renvoie le résultat (sous la forme d'un nombre en double précision), de l'exécution d'une requête AQL. Si la requête ne renvoie aucun résultat, la valeur 0 est retournée sans déclencher d'erreur.

Syntaxe API

double AmDbGetDouble(long hApiCnxBase, char *strQuery);

Syntaxe Basic interne

Function AmDbGetDouble(strQuery As String) As Double

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strQuery** : Ce paramètre contient l'intégralité de la requête AQL dont on veut récupérer le résultat.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmDbGetList()

Cette fonction renvoie, sous la forme d'une liste, le résultat de l'exécution d'une requête AQL. Le nombre d'éléments sélectionnés par la requête AQL est limité à 99.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **strQuery** : Ce paramètre contient la requête AQL que vous souhaitez exécuter.
- **strColSep** : Ce paramètre contient le caractère utilisé comme séparateur de colonnes dans le résultat donné par la fonction.
- **strLineSep** : Ce paramètre contient le caractère utilisé comme séparateur de lignes dans le résultat donné par la fonction.
- **strIdSep** : Ce paramètre contient le caractère utilisé comme séparateur d'identifiant dans le résultat donné par la fonction.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmDbGetListEx()

Cette fonction renvoie, sous la forme d'une liste, le résultat de l'exécution d'une requête AQL. A la différence de la fonction **AmDbGetList**, cette fonction n'est pas limitée dans le nombre d'éléments sélectionnés par la requête AQL.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strQuery** : Ce paramètre contient la requête AQL que vous souhaitez exécuter.
- **strColSep** : Ce paramètre contient le caractère utilisé comme séparateur de colonnes dans le résultat donné par la fonction.
- **strLineSep** : Ce paramètre contient le caractère utilisé comme séparateur de lignes dans le résultat donné par la fonction.
- **strIdSep** : Ce paramètre contient le caractère utilisé comme séparateur d'identifiant dans le résultat donné par la fonction.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction

AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmDbGetLong()

Cette fonction renvoie le résultat de l'exécution d'une requête AQL. Si la requête ne renvoie aucun résultat, la valeur 0 est retournée sans déclencher d'erreur.

Syntaxe API

long AmDbGetLong(long hApiCnxBase, char *strQuery);

Syntaxe Basic interne

Function AmDbGetLong(strQuery As String) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strQuery** : Ce paramètre contient l'intégralité de la requête AQL dont on veut récupérer le résultat.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

L'exemple suivant renvoie le numéro d'identifiant d'un fournisseur de produit :

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

AmDbGetPk()

Cette fonction renvoie la clé primaire d'une table en fonction de la clause WHERE d'une requête AQL. Si la requête ne renvoie aucun résultat, la valeur 0 est retournée sans déclencher d'erreur.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strTableName** : Nom SQL de la table dont on veut récupérer la clé primaire.
- **strWhere** : Clause WHERE d'une requête AQL.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmDbGetString()

Cette fonction renvoie, sous la forme d'une chaîne formatée, le résultat de l'exécution d'une requête AQL. Le nombre d'éléments sélectionnés par la requête AQL est limité à 99.

Avertissement :

N'utilisez pas cette fonction pour récupérer la valeur d'un simple champ de type chaîne. Cette fonction est à rapprocher de la fonction AmDbGetList ou de la fonction AmDbGetListEx.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strQuery** : Ce paramètre contient la requête AQL que vous souhaitez exécuter.
- **strColSep** : Ce paramètre contient le caractère utilisé comme séparateur de colonnes dans la chaîne finale.
- **strLineSep** : Ce paramètre contient le caractère utilisé comme séparateur de lignes dans la chaîne finale.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.

- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Remarques

Dans la syntaxe API, le paramètre lResult doit contenir la taille attendue du résultat de l'exécution de la fonction.

Exemple

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

retournera la chaine :

```
Carpenter|/Taltek/I.S. Department/Carpenter\, Jerome\, DEMO-M016/=2345
9,Chavez|/Taltek/I.S. Department/Chavez\, Philip\, DEMO-M014/=23460,Ch
ouraqui|/Taltek/Sales/Los Angeles Agency/Chouraqui\, Thomas\, DEMO-M01
7/=23491,Cipriani|/Taltek/Sales/Los Angeles Agency/Cipriani\, Fred\, D
EMO-M018/=23492,Clech|/Taltek/Sales/Burbank Agency/Clech\, Richard\, D
EMO-M021/=23482,Colombo|/Taltek/Finance/Colombo\, Gerald\, DEMO-M022/=
23441
```

On remarque le caractère d'échappement \ devant les virgules.

La même requête avec **amDbGetString()** n'ajoutera pas ces caractères d'échappement, ce qui la rend impropre pour remplir une liste. Par exemple :

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

affichera :

```
/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()

Cette fonction renvoie, sous la forme d'une chaîne formatée, le résultat de l'exécution d'une requête AQL. A la différence de la fonction **AmDbGetString**, cette fonction n'est pas limitée dans le nombre d'éléments sélectionnés par la requête AQL.

Avertissement :

N'utilisez pas cette fonction pour récupérer la valeur d'un simple champ de type chaîne. Cette fonction est à rapprocher de la fonction AmDbGetList ou de la fonction AmDbGetListEx.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **strQuery** : Ce paramètre contient la requête AQL que vous souhaitez exécuter.
- **strColSep** : Ce paramètre contient le caractère utilisé comme séparateur de colonnes dans la chaîne finale.
- **strLineSep** : Ce paramètre contient le caractère utilisé comme séparateur de lignes dans la chaîne finale.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmDeadLine()

Cette fonction calcule une date d'échéance en fonction d'un calendrier, d'une date de début et d'un nombre de secondes ouvrées écoulées.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strCalendarSqlName** : Ce paramètre contient le nom SQL du calendrier des périodes ouvrées utilisé comme base pour le calcul de la date d'échéance.
- **tmStart** : Ce paramètre contient la date de début de la période.
- **tsDuration** : Ce paramètre contient le nombre de secondes ouvrées écoulées à partir de la date de début de période.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

L'exemple suivant calcule la date d'échéance en fonction du calendrier de nom SQL "Calendar_Paris", d'une date de début de période fixée au 01/09/1998 à 8h00 et d'un nombre de secondes écoulées de 450000.

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

Cet exemple renvoie la date d'échéance, à savoir le 22/09/1998 à 18h00.

AmDecrementLogLevel()

Cette fonction permet de remonter d'un cran dans l'arborescence d'une fenêtre de journal de la page terminale d'un assistant.

Syntaxe Basic interne

Function AmDecrementLogLevel() As Long

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.

- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmDefAssignee()

Cette fonction recherche le numéro d'identifiant du chargé de groupe par défaut pour un groupe de support donné.

Syntaxe API

long AmDefAssignee(long hApiCnxBase, long lGroupId);

Syntaxe Basic interne

Function AmDefAssignee(lGroupId As Long) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **lGroupId** : Ce paramètre contient le numéro d'identifiant d'un groupe de support.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

L'exemple générique suivant renvoie l'identifiant du chargé par défaut d'un groupe de support :

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

Vous pouvez directement saisir la valeur numérique de l'identifiant, comme dans cet exemple :

```
AmDefAssignee(24)
```

AmDefaultCurrency()

Cette fonction renvoie la devise par défaut utilisée sous AssetCenter.

Syntaxe API

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

Syntaxe Basic interne

Function AmDefaultCurrency() As String

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmDefEscalationScheme()

Cette fonction recherche la procédure d'escalade par défaut en fonction de la localisation et de la gravité du dossier de support.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strLocFullName** : Ce paramètre contient le nom complet de la localisation.
- **lSeverityLvl** : Ce paramètre contient la valeur de la gravité.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

L'exemple générique suivant renvoie l'identifiant de la procédure d'escalade par défaut en fonction de la localisation et de la gravité :

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

Vous pouvez directement saisir les valeurs des paramètres, comme dans cet exemple :

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

AmDefGroup()

Cette fonction renvoie le numéro d'identifiant du groupe de support par défaut en fonction d'un type de problème, d'une localisation et d'un contrat de maintenance.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **lProblemClassId** : Ce paramètre contient le numéro d'identifiant d'un type de problème.
- **strLocFullName** : Ce paramètre contient le nom complet d'une localisation.
- **lAssetMainCntId** : Ce paramètre contient le numéro d'identifiant d'un contrat de maintenance.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Remarques

La méthode utilisée pour définir le groupe de support par défaut est la suivante :

1. La fonction recherche les groupes de support associés au type de problème du dossier.
2. Parmi les groupes ainsi retenus, la fonction recherche les groupes de support associés à la localisation la plus "proche" de celle du bien : localisation directe, sinon localisation parente, et ainsi de suite jusqu'à la localisation racine.
3. Si aucun groupe ne peut être retenu par la méthode précédente, et si le moteur supporte les doubles jointures externes, la fonction recherche les groupes qui ne sont associés à aucune localisation.

   Pour connaître la liste des SGBD qui supportent les doubles jointures externes, consultez le manuel **Helpdesk**, chapitre **Références (Helpdesk)**, section **SGBD qui supportent les doubles jointures externes**.
4. Si le moteur supporte les doubles-jointures externes, la fonction sélectionne, parmi les groupes précédement retenus, le groupe de support dans le cadre de contrats desquels les groupes de support interviennent et des contrats de maintenance couvrant le bien.
5. Si aucun groupe n'est trouvé, la fonction reprend les étapes 1, 2, 3 et 4 en partant du type de problème d'un niveau supérieur dans l'arborescence des types de problèmes, et ceci jusqu'au type de problème racine.

Exemple

L'exemple générique suivant calcule le numéro d'identifiant du groupe de support par défaut en fonction des trois paramètres : type de problème, localisation et contrat de maintenance.

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

Vous pouvez directement saisir la valeur numérique des paramètres utilisant des numéros d'identifiant, comme dans cet exemple :

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

AmDeleteLink()

Cette fonction détruit un lien d'un enregistrement.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	

	Utilisable
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiRecord**: Ce paramètre contient le descripteur sur l'enregistrement contenant le lien à détruire.
- **strLinkName**: Ce paramètre contient le nom SQL du lien à détruire.
- **hApiRecDest**: Ce paramètre contient un descripteur sur l'enregistrement de destination du lien qui doit être détruit.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmDeleteRecord()

Cette fonction détruit un enregistrement dans la base de données.

Syntaxe API

long AmDeleteRecord(long hApiRecord);

Syntaxe Basic interne

Function AmDeleteRecord(hApiRecord As Long) As Long

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiRecord** : Ce paramètre contient un descripteur sur l'enregistrement que vous souhaitez détruire.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmDisconnectTrace()

L'API AmDisconnectTrace déconnecte la chaîne de liaisons entre un noeud utilisateur (lEndId) et un noeud hôte (lStartId) dans la table des liaisons de câbles. Si tout noeud se trouve à la fin d'une chaîne de liaisons, il sera supprimé de la table des liaisons de câbles. Il crée aussi des entrées d'historique de chaîne de liaisons et des entrées d'opérations sur chaîne de liaisons suite à la déconnexion.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lStartId** : ce paramètre contient l'identifiant de la connexion hôte lequel va être déconnecté.
- **lEndId** : ce paramètre contient l'identifiant de la connexion utilisateur lequel va être déconnecté.
- **strComment** : ce paramètre est la chaîne de l'opération sur la chaîne de liaisons montrant les nouvelles connexions et déconnexions.
- **lCabTraceOutId** : ce paramètre contient l'identifiant de compte-rendu de chaîne de liaisons de câble.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmDuplicateRecord()

Cette fonction permet de dupliquer un enregistrement.

Syntaxe API

long AmDuplicateRecord(long hApiRecord, long bInsert);

Syntaxe Basic interne

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

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiRecord** : Ce paramètre contient le descripteur de l'enregistrement à dupliquer.
- **bInsert** : Ce paramètre permet de préciser si vous souhaitez insérer l'enregistrement dupliqué immédiatement (=1) ou non (=0).

Sortie

- 0 : La fonction s'est exécutée normalement.

- Non nul : Code d'erreur.

AmEndOfNthBusinessDay()

Donne la dernière heure ouvrée du nième jour (identifié par l'entier **lDayCount**) à compter d'une date donnée et en respectant un calendrier.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strCalendarSqlName** : Nom du calendrier utilisé pour le calcul.
- **tmStart** : Date de début du calcul.

- **lDayCount** : Nombre de jours ouvrés entiers à ajouter à dStart pour le calcul.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmEnumValList()

Cette fonction renvoie une chaîne contenant toutes les valeurs d'une énumération système. Les différentes valeurs sont triées alphabétiquement et sont délimitées par le séparateur indiqué dans le paramètre **strLineSep**.

Dans le cas où une valeur de l'énumération contient le caractère utilisé comme séparateur ou un "\", le préfixe "\" est utilisé.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strEnumName** : Ce paramètre contient le nom SQL de l'énumération système dont vous souhaitez récupérer les valeurs.
- **bNoCase** : Ce paramètre permet de préciser si le tri alphabétique tient compte de la casse (=1) ou non (=0).
- **strLineSep** : Ce paramètre contient le caractère utilisé pour délimiter les valeurs de l'énumération.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmEvalScript()

Cette fonction permet d'évaluer un script par son nom à partir du contexte courant. Cette fonction possède deux utilisations :

- Evaluer un script système (Valeur par défaut, Obligatoire...)
- Appeler une fonction d'une bibliothèque de scripts.

Syntaxe Basic interne

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

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strScriptName** : Ce paramètre contient le nom du script à évaluer. Dans le premier cas d'utilisation, il s'agit du nom du script système (DefVal, ...). Dans le deuxième cas d'utilisation, il s'agit du nom d'une bibliothèque de scripts.
- **strObject** : Ce paramètre contient l'objet auquel se rapporte le script. Il peut s'agir du nom SQL d'un champ ou d'une fonction faisant partie d'une bibliothèque.
- **strPath** : Ce paramètre optionnel permet de préciser un chemin (lien.lien.lien...) qui permet de décaler le contexte d'évaluation du script. Ce paramètre est inopérant dans le deuxième cas d'utilisation de la fonction.
- **...** : Pour l'appel à une fonction d'une bibliothèque de scripts, permet de passer des paramètres à la fonction appelée.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Remarques

Voici la liste des noms des scripts système utilisables :

- Sur une table : IsValid, IsRelevant
- Sur un champ : DefVal, Mandatory, Historized, ReadOnly, Irrelevant
- Sur un lien : Historized, Filter, Irrelevant
- Sur une caractéristique : DefVal, Mandatory, Available, Historized

AmExecTransition()

Cette fonction déclenche une transition valide à partir de la page en cours.

Syntaxe Basic interne

Function AmExecTransition(strTransName As String) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **strTransName** : Ce paramètre contient le nom de la transition tel qu'il est défini dans le script de l'assistant. Une erreur est renvoyée si la transition n'existe pas. La fonction est inopérante (et ne renvoie pas d'erreur) si la transition n'est pas valide.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmExecuteActionById()

Cette fonction exécute une action identifiée par son identifiant.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓

	Utilisable
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lActionId** : Ce paramètre contient l'identifiant de l'action à exécuter.
- **strTableName** : Dans le cas d'une action contextuelle, ce paramètre contient le nom SQL de la table sur laquelle l'action est exécutée. Si ce paramètre est omis, dans le cas d'une action contextuelle, la fonction échouera. Pour une action non contextuelle, ce paramètre n'est pas interprété et donc facultatif.
- **lRecordId** : Ce paramètre contient l'identifiant de l'enregistrement sur lequel porte éventuellement l'action. Pour une action non contextuelle, ce paramètre n'est pas interprété et donc facultatif.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmExecuteActionByName()

Cette fonction exécute une action identifiée par son nom SQL.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **strSqlName** : Ce paramètre contient le nom SQL de l'action à exécuter.
- **strTableName** : Dans le cas d'une action contextuelle, ce paramètre contient le nom SQL de la table sur laquelle l'action est exécutée. Si ce paramètre est omis, dans le cas d'une action contextuelle, la fonction échouera. Pour une action non contextuelle, ce paramètre n'est pas interprété et donc facultatif.
- **lRecordId** : Ce paramètre contient l'identifiant de l'enregistrement sur lequel porte éventuellement l'action. Pour une action non contextuelle, ce paramètre n'est pas interprété et donc facultatif.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmExportDocument()

Cette fonction permet d'exporter un document attaché à un enregistrement.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **lDocId** : Ce paramètre contient l'identifiant du document à exporter.
- **strFileName** : Ce paramètre contient le nom du document à exporter, tel qu'il est stocké dans le champ FileName de la table des documents.

Sortie

- 0 : La fonction s'est exécutée normalement.

- Non nul : Code d'erreur.

AmExportReport()

Cette fonction permet d'exporter dans un fichier un rapport Crystal de la base de données.

Syntaxe Basic interne

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

Champ d'application

Version : 4.3.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **lReportId** : Ce paramètre contient l'identifiant de l'enregistrement du rapport Crystal à exporter.
- **strFileName** : Ce paramètre contient le chemin complet du fichier dans lequel est réalisé l'export.

Sortie

- 0 : La fonction s'est exécutée normalement.

- Non nul : Code d'erreur.

AmFindCable()

L'API AmFindCable trouve le câble disponible suivant tiré entre la localisation d'un utilisateur (lUserId) et d'un hôte (lHostId) donnés. Le câble doit être du type (strCabType) et du rôle (strCableRole) spécifiés. Aussi, l'état du câble doit prendre la valeur "Disponible". Les câbles sont triés par ordre croissant d'identifiant et seuls les câbles plus grands que l'identifiant (lPrevCabId) du câble précédent sont sélectionnés.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **lPrevCableId** : ce paramètre contient l'identifiant du câble précédent.
- **strCabType** : ce paramètre définit le type du câble à chercher.
- **lUserId** : ce paramètre contient l'identifiant de la localisation utilisateur.
- **lHostId** : ce paramètre contient l'identifiant de la localisation hôte.
- **strCableRole** : ce paramètre est le rôle du câble à localiser.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmFindDevice()

L'API AmFindDevice trouve un dispositif d'un type (strDevType) donné dans une localisation (lLocationId) donnée. Les dispositifs sont triés par ordre croissant d'identifiant et seuls ceux plus grands que le dispositif (lPrevDeviceId) précédent sont sélectionnés.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **lPrevDeviceId** : ce paramètre contient l'identifiant du précédent dispositif recherché. La valeur 0 est utilisée pour démarrer une recherche.
- **strDeviceType** : ce paramètre définit le type du dispositif à localiser.
- **lLocationId** : ce paramètre contient l'identifiant de la localisation à chercher.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmFindRootLink()

Cette fonction permet de récupérer la liaison racine d'une chaîne de liaison.

Syntaxe API

long AmFindRootLink(long hApiCnxBase, long lLinkId);

Syntaxe Basic interne

Function AmFindRootLink(lLinkId As Long) As Long

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **lLinkId** : Ce paramètre contient l'identifiant du lien concerné par l'opération.

Sortie

La fonction renvoie l'identifiant de la liaison racine.

AmFindTermDevice()

L'API AmFindTermDevice trouve le dispositif disponible suivant dans un répartiteur donné (lTermField) pour un rôle de câble donné (strCableRole). Les dipositifs sont triés par ordre croissant de numéro de séquence et seuls les biens plus grands que le numéro de séquence précédent (strPrevTermSeq) sont séléctionnés. Aussi, pour les dispositifs à base de broches (bPinBased=1), on compare le nombre total de broches requises (iPinPortCount) au nombre total de broches restant sur le dispositif. Pour les dispositifs à base de ports (bPinBased=0) on s'assure qu'il y a au moins un port restant sur le dispositif et que le côté hôte ou utilisateur de ce port est disponible grâce au drapeau (bCheckAvail = 0 - user device, bCheckAvail = 1 - host device).

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **iPrevTermSeq** : ce paramètre est la séquence précédente du répartiteur recherché. La valeur 0 est utilisée pour démarrer une recherche.
- **lTermFieldId** : ce paramètre contient l'identifiant du répartiteur.
- **strCableRole** : ce paramètre est le rôle du câble à localiser.
- **bPinBased** : ce paramètre précise si le dispositif est à base de broches ou de ports.
- **iPinPortCount** : pour les dispositifs à base de broches, ce paramètre est le nombre total de broches requis pour créer un port virtuel. Pour les dispositifs à base de ports, ce paramètre est 1 puisque cette API est appelée pour chaque port requis.
- **bCheckAvail** : ce paramètre sert à déterminer quel côté du port doit être disponible.
  - 0=dispositif utilisateur, vérifie l'hôte disponible
  - 1=dispositif hôte, vérifie l'utilisateur disponible

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmFindTermField()

L'API AmFindTermField trouve un répartiteur qui fournit la fonction (lDutyId) donnée depuis la localisation (lLocationId) donnée. Elle continuera à trouver des répartiteurs additionnels dans une localisation donnée et pour une fonction donnée si la valeur de lPrevTermFieldId est plus grande que 0.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **lDutyId** : ce paramètre définit la fonction à localiser.
- **lLocationId** : ce paramètre contient l'identifiant de la localisation à chercher.
- **lPrevTermFieldId** : ce paramètre contient l'identifiant du répartiteur.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction

AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmFlushTransaction()

Cette fonction purge la liste des tâches des agents (comme après une opération de Commit à la base de données).

Syntaxe API

long AmFlushTransaction(long hApiCnxBase);

Syntaxe Basic interne

Function AmFlushTransaction() As Long

Champ d'application

Version : 4.3.0

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmFormatCurrency()

Cette fonction affiche une valeur monétaire dans une devise donnée. Le symbole standard de la devise est également affiché.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 4.3.0

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **dAmount** : Ce paramètre contient la valeur monétaire à afficher.
- **strCurrency** : Ce paramètre contient la devise utilisée pour l'opération.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

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

Cet exemple affiche :

```
US$500,00
```

AmFormatLong()

Cette fonction remplace un token dans une chaîne de caractères par la valeur contenue dans une variable de type Long.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 4.3.0

	Utilisable
AssetCenter API	✓

	Utilisable
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **lNumber** : Ce paramètre contient le Long à insérer dans la chaîne de caractères contenue dans le paramètre **strFormat**.
- **strFormat** : Ce paramètre contient la chaîne de caractères à traiter. Tous les tokens de type "%d" sont remplacés par la valeur contenue dans le paramètre **lNumber**.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGeneratePlanningData()

Cette fonction permet de générer une visualisation graphique des plannings.

Syntaxe Basic interne

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

Champ d'application

Version : 4.3.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strTableSqlName** : Ce paramètre contient le nom SQL de la table contenant les données à partir desquelles le planning est généré.
- **strProperties** : Ce paramètre contient les propriétés du planning créé.

Note :

Pour plus d'informations sur la syntaxe de ces propriétés, reportez-vous au manuel d'Administration, section Référence : syntaxe du paramétrage des pages de visualisation graphique des plannings.

- **strIds** : Ce paramètre contient la liste des identifiants (séparés par une virgule) des enregistrements dont les données sont prises en compte dans la création du planning.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGenSqlName()

Cette fonction génère un nom SQL valide à partir d'une chaîne texte classique. Les espaces sont remplacés par des underscores ("_"). Cette fonction est particulièrement utile pour définir la valeur par défaut du nom SQL d'une caractéristique à partir de son nom.

Syntaxe API

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

Syntaxe Basic interne

Function AmGenSqlName(strText As String) As String

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strText** : Chaîne de caractères à partir de laquelle vous souhaitez générer un nom SQL.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

L'exemple suivant définit la valeur par défaut du nom SQL d'un objet de nom "Label" de la base de données AssetCenter :

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

AmGetCatRef()

Cette fonction recherche pour un modèle donné une référence hors catalogue valide (les dates de validité sont respectées) pour laquelle les règles suivantes sont respectées :

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

La fonction renvoie en priorité une référence qui n'a pas été créée au vol. Si aucune référence n'est trouvée est que le paramètre **bCreate** a pour valeur "1", une nouvelle référence hors catalogue ainsi qu'un produit sont créés (lequel pointe sur le modèle).

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 4.1.0

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lModelId** : Ce paramètre contient l'identifiant du modèle concerné par l'opération.
- **bCreate** : Ce paramètre permet de préciser si une référence hors catalogue est créée, dans le cas où la recherche ne renvoie aucun résultat.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Remarques

Note :

La fonction ne requiert pas de préciser un fournisseur, puisque la recherche s'effectue sur des références hors catalogue, quelque soit le fournisseur.

AmGetCatRefFromCatProduct()

Cette fonction est identique à la fonction **amGetCatRef**, au détail près que la recherche s'effectue pour un produit particulier.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 4.1.0

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lCatProductId** : Ce paramètre contient l'identfiant du produit concerné par l'opération.
- **bCreate** : Ce paramètre permet de préciser si une référence hors catalogue est créée, dans le cas où la recherche ne renvoie aucun résultat.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetComputeString()

Cette fonction renvoie la chaîne de description d'un enregistrement donné en fonction d'un modèle.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	

	Utilisable
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strTableName** : Ce paramètre contient le nom SQL de la table de l'enregistrement dont on souhaite récupérer la chaîne de description.
- **lRecordId** : Ce paramètre contient l'identifiant de l'enregistrement au sein de la table.
- **strTemplate** : Ce paramètre contient, sous forme de chaîne de caractères, le modèle utilisé pour la chaîne de description.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

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

AmGetCurrentNTDomain()

Cette fonction retourne le nom du domaine NT du login courant.

Syntaxe API

long AmGetCurrentNTDomain(char *pstrDomain, long lDomain);

Syntaxe Basic interne

Function AmGetCurrentNTDomain() As String

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
RetVal = amGetCurrentNTDomain()
```

AmGetCurrentNTUser()

Cette fonction permet de récupérer le login de l'utilisateur connecté à Windows (NT ou 2000).

Syntaxe API

long AmGetCurrentNTUser(char *pstrUser, long lUser);

Syntaxe Basic interne

Function AmGetCurrentNTUser() As String

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetFeat()

Cette fonction créée un objet caractéristique à partir du descripteur d'une table et retourne le descripteur de l'objet caractéristique créé.

Syntaxe API

long AmGetFeat(long hApiTable, long lPos);

Syntaxe Basic interne

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

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiTable** : Ce paramètre contient un descripteur sur une table.
- **lPos** : Ce paramètre contient la position de la caractéristique à l'intérieur de la table.

AmGetFeatCount()

Cette fonction renvoie le nombre de caractéristiques sur la table précisée dans le paramètre **hApiTable**.

Syntaxe API

long AmGetFeatCount(long hApiTable);

Syntaxe Basic interne

Function AmGetFeatCount(hApiTable As Long) As Long

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiTable** : Ce paramètre contient un descripteur sur une table.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetField()

Cette fonction créée un objet champ à partir du descripteur d'une requête, d'un enregistrement ou d'une table et retourne le descripteur de l'objet champ créé.

Syntaxe API

long AmGetField(long hApiObject, long lPos);

Syntaxe Basic interne

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

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiObject** : Ce paramètre contient un descripteur sur une requête, un enregistrement ou une table.
- **lPos** : Ce paramètre contient la position du champ (son index) à l'intérieur de l'objet.

AmGetFieldCount()

Cette fonction renvoie le nombre de champs contenus dans l'objet courant.

Syntaxe API

long AmGetFieldCount(long hApiObject);

Syntaxe Basic interne

Function AmGetFieldCount(hApiObject As Long) As Long

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiObject** : Ce paramètre contient un descripteur sur un enregistrement, une requête ou une table valides.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetFieldDateOnlyValue()

Cette fonction renvoie la valeur d'un champ contenu dans l'objet courant. Cette valeur est renvoyée au format "Date" (à partir d'un outil externe, il s'agit

d'un Long). A l'inverse de la fonction **AmGetFieldDateValue**, seule la partie Date est renvoyée, la partie heure est omise.

Syntaxe API

long AmGetFieldDateOnlyValue(long hApiObject, long lFieldPos);

Syntaxe Basic interne

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

Champ d'application

Version : 4.3.0

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiObject** : Ce paramètre contient un descripteur sur une requête ou un enregistrement.
- **lFieldPos** : Ce paramètre contient le numéro du champ à l'intérieur de l'objet courant.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetFieldDateValue()

Cette fonction renvoie la valeur d'un champ contenu dans l'objet courant. Cette valeur est renvoyée au format "Date" (à partir d'un outil externe, il s'agit d'un Long).

Syntaxe API

long AmGetFieldDateValue(long hApiObject, long lFieldPos);

Syntaxe Basic interne

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

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiObject** : Ce paramètre contient un descripteur sur une requête ou un enregistrement.
- **lFieldPos** : Ce paramètre contient le numéro du champ à l'intérieur de l'objet courant.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetFieldDescription()

Cette fonction renvoie, sous la forme d'une chaîne de caractères (format "String"), la description d'un champ identifié par un descripteur.

Syntaxe API

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

Syntaxe Basic interne

Function AmGetFieldDescription(hApiField As Long) As String

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiField** : Ce paramètre contient un descripteur valide sur le champ dont on souhaite connaître la description longue.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetFieldDoubleValue()

Cette fonction renvoie la valeur d'un champ contenu dans l'objet courant. Cette valeur est renvoyée au format "Double".

Syntaxe API

double AmGetFieldDoubleValue(long hApiObject, long lFieldPos);

Syntaxe Basic interne

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

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiObject** : Ce paramètre contient un descripteur sur une requête ou un enregistrement.
- **lFieldPos** : Ce paramètre contient le numéro du champ à l'intérieur de l'objet courant.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetFieldFormat()

Cette fonction est utile quand le "UserType" (cf. fichier "database.txt") du champ concerné a pour valeur :

- Enumération Système
- Enumération
- Durée
- Nom de table ou de champ

La fonction renvoie alors le format du "UserType", à savoir :

UserType	Format renvoyé par la fonction
Enumération Système	Liste des entrées de l'énumération système.
Enumération	Nom de l'énumération associée au champ.
Durée	Format d'affichage.
Nom de table ou de champ	Nom SQL du champ qui stocke le nom SQL de la table contenant le champ que précise le champ décrit.

Syntaxe API

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

Syntaxe Basic interne

Function AmGetFieldFormat(hApiField As Long) As String

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	

	Utilisable
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiField** : Ce paramètre contient un descripteur valide sur le champ dont on souhaite connaître le "UserType".

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetFieldFormatFromName()

Cette fonction renvoie le format du "UserType" d'un champ, à partir de son nom.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strTableName** : Ce paramètre contient le nom SQL de la table contenant le champ concerné par l'opération.
- **strFieldName** : Ce paramètre contient le nom SQL du champ.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetFieldFromName()

Cette fonction créée un objet champ à partir de son nom et retourne le descripteur de l'objet champ créé.

Syntaxe API

long AmGetFieldFromName(long hApiObject, char *strName);

Syntaxe Basic interne

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

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiObject** : Ce paramètre contient un descripteur sur une requête, un enregistrement ou une table.
- **strName** : Ce paramètre contient le nom du champ.

AmGetFieldLabel()

Cette fonction renvoie, sous la forme d'une chaîne de caractères (format "String"), le libellé d'un champ identifié par un descripteur.

Syntaxe API

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

Syntaxe Basic interne

Function AmGetFieldLabel(hApiField As Long) As String

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiField** : Ce paramètre contient un descripteur valide sur le champ dont on souhaite connaître le libellé.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetFieldLabelFromName()

Cette fonction renvoie le label d'un champ à partir de son nom SQL.

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strTableName** : Ce paramètre contient le nom SQL de la table contenant le champ concerné par l'opération.
- **strFieldName** : Ce paramètre contient le nom SQL du champ.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction

AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetFieldLongValue()

Cette fonction renvoie la valeur d'un champ contenu dans l'objet courant.

Syntaxe API

long AmGetFieldLongValue(long hApiObject, long lFieldPos);

Syntaxe Basic interne

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

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiObject** : Ce paramètre contient un descripteur sur une requête ou un enregistrement.
- **lFieldPos** : Ce paramètre contient le numéro du champ à l'intérieur de l'objet courant.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Remarques

Note :

Si vous utilisez cette fonction pour récupérer la valeur d'un champ de type date, heure ou date+heure, l'entier long renvoyé par la fonction représente le nombre de secondes écoulées depuis le 01/01/1970 à 00:00:00.

AmGetFieldName()

Cette fonction renvoie le nom d'un champ contenu dans l'objet courant.

Syntaxe API

long AmGetFieldName(long hApiObject, long lFieldPos, char *pstrBuffer, long lBuffer);

Syntaxe Basic interne

Function AmGetFieldName(hApiObject As Long, lFieldPos As Long) As String

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiObject** : Ce paramètre contient un descripteur sur une requête, un enregistrement ou une table.
- **lFieldPos** : Ce paramètre contient le numéro du champ à l'intérieur de l'objet courant. Par exemple, la valeur "0" désigne le premier champ.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetFieldRights()

Cette fonction renvoie les droits utilisateurs d'un champ de l'objet courant. Ces droits sont renvoyés sous la forme d'une chaîne composée de trois caractères précisant les droits en lecture/ insertion/ mise à jour :

- "r" : désigne l'autorisation en lecture.
- "i" : désigne l'autorisation en insertion.
- "u" : désigne l'autorisation en mise à jour.

Par exemple, pour un champ en lecture seule, la fonction renverra la valeur "r ".

Syntaxe API

long AmGetFieldRights(long hApiObject, long lFieldPos, char *pstrBuffer, long lBuffer);

Syntaxe Basic interne

Function AmGetFieldRights(hApiObject As Long, lFieldPos As Long) As String

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiObject** : Ce paramètre contient un descripteur sur une requête, un enregistrement ou une table.
- **lFieldPos** : Ce paramètre contient le numéro du champ à l'intérieur de l'objet courant.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction

AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetFieldSize()

Cette fonction renvoie la taille d'un champ.

Syntaxe API

long AmGetFieldSize(long hApiField);

Syntaxe Basic interne

Function AmGetFieldSize(hApiField As Long) As Long

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiField** : Ce paramètre contient un descripteur sur le champ dont on veut connaître la taille.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetFieldSqlName()

Cette fonction renvoie, sous la forme d'une chaîne de caractères (format "String"), le nom SQL d'un champ identifié par un descripteur.

Syntaxe API

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

Syntaxe Basic interne

Function AmGetFieldSqlName(hApiField As Long) As String

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiField** : Ce paramètre contient un descripteur valide sur le champ dont on souhaite connaître le nom SQL.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetFieldStrValue()

Cette fonction renvoie la valeur d'un champ contenu dans l'objet courant. Cette valeur est renvoyée au format chaîne.

Attention : Quand cette fonction est utilisée au travers des APIs AssetCenter, elle attend deux paramètres supplémentaires **strBuffer** et **lBuffer**, qui définissent respectivement une chaîne de caractères utilisée comme buffer pour le stockage de la chaîne récupérée et la taille de ce buffer. La chaîne **strBuffer** doit être formatée (remplie de caractères) et posséder la taille définie par **lBuffer**. La portion de code suivante est incorrecte, la chaîne utilisée comme buffer n'étant pas dimensionnée :

```
Dim strBuffer as String
Dim lRec as Long
Dim lBuffer as Long
lBuffer=20
lRec=AmGetFieldStrValue(1, 0, strBuffer, lBuffer)
```

Voici la portion de code corrigée :

```
Dim strBuffer as String
Dim lRec as Long
```

```
Dim lBuffer as Long
strBuffer=String(21, " ") ' Le buffer est dimensionné à 21 caractères
(" ")
lBuffer=20
lRec=AmGetFieldStrValue(1, 0, strBuffer, lBuffer)
```

Lorsque vous formattez la chaîne du buffer au moyen de la fonction "String", n'utilisez jamais "0" comme caractère de remplissage. Dimensionnez le buffer avant chaque appel de la fonction **AmGetFieldStrValue**, en particulier si cette fonction se trouve dans une boucle et utilise toujours la même chaîne comme buffer.

Syntaxe API

long AmGetFieldStrValue(long hApiObject, long lFieldPos, char *pstrBuffer, long lBuffer);

Syntaxe Basic interne

Function AmGetFieldStrValue(hApiObject As Long, lFieldPos As Long) As String

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiObject** : Ce paramètre contient un descripteur sur une requête ou un enregistrement.

- **lFieldPos** : Ce paramètre contient le numéro du champ à l'intérieur de l'objet courant.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetFieldType()

Cette fonction renvoie le type d'un champ.

Syntaxe API

long AmGetFieldType(long hApiField);

Syntaxe Basic interne

Function AmGetFieldType(hApiField As Long) As Long

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	

	Utilisable
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiField** : Ce paramètre contient un descripteur sur le champ dont on veut connaître le type.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Remarques

Note :

Le tableau ci-dessous donne la correspondance entre les valeurs retournées par la fonction **AmGetFieldType** et les types de champs :

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()

Cette fonction renvoie le "UserType" (cf. fichier **database.txt**) d'un champ identifié par un descripteur, sous la forme d'un entier long. Pour un champ, les valeurs valides renvoyées sont récapitulées dans le tableau ci-dessous :

Valeur stockée	Valeur en clair
0	Default
1	Number
2	Yes/ No
3	Money
4	Date
5	Date+Time
7	System itemized list
8	Custom itemized list
10	Percentage

Valeur stockée	Valeur en clair
11	Time span
12	Table or field SQL name

Pour un lien, les valeurs valides sont les suivantes :

Valeur stockée	Valeur en clair
0	Normal
1	Comment
2	Image
3	History
4	Feature value

Jusqu'à la version 4.0.0, la fonction renvoie toujours 0 pour un lien. A partir de la version 4.1.0 d'AssetCenter, la fonction renvoie une des valeurs suivantes pour un lien :

- 0 : Normal
- 1 : Commentaire
- 2 : Image
- 3 : Historique
- 5 : Script

Syntaxe API

long AmGetFieldUserType(long hApiField);

Syntaxe Basic interne

Function AmGetFieldUserType(hApiField As Long) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓

	Utilisable
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiField** : Ce paramètre contient un descripteur valide sur le champ dont on souhaite connaître le "UserType".

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetForeignKey()

Récupère le descripteur sur la clé externe d'un lien, lui même identifié par son descripteur.

Syntaxe API

long AmGetForeignKey(long hApiField);

Syntaxe Basic interne

Function AmGetForeignKey(hApiField As Long) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiField** : Descripteur sur le lien concerné par l'opération.

AmGetIndex()

Cette fonction créée un objet index à partir du descripteur d'une requête, d'un enregistrement ou d'une table et retourne le descripteur de l'objet index créé.

Syntaxe API

long AmGetIndex(long hApiTable, long lPos);

Syntaxe Basic interne

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

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiTable** : Ce paramètre contient un descripteur sur une table.
- **lPos** : Ce paramètre contient la position de l'index à l'intérieur de la table.

AmGetIndexCount()

Cette fonction renvoie le nombre d'index contenus dans la table précisée dans le paramètre **hApiTable**.

Syntaxe API

long AmGetIndexCount(long hApiTable);

Syntaxe Basic interne

Function AmGetIndexCount(hApiTable As Long) As Long

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓

	Utilisable
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiTable** : Ce paramètre contient un descripteur sur une table.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetIndexField()

Cette fonction renvoie un descripteur sur un champ identifié par sa position au sein de l'index (le lpos ème champ de l'index).

Syntaxe API

long AmGetIndexField(long hApiIndex, long lPos);

Syntaxe Basic interne

Function AmGetIndexField(hApiIndex As Long, lPos As Long) As Long

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiIndex** : Ce paramètre contient un descripteur valide sur l'index concerné par l'opération.
- **lPos** : Ce paramètre contient la position du champ au sein de l'index.

AmGetIndexFieldCount()

Cette fonction renvoie le nombre de champs qui composent un index.

Syntaxe API

long AmGetIndexFieldCount(long hApiIndex);

Syntaxe Basic interne

Function AmGetIndexFieldCount(hApiIndex As Long) As Long

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓

	Utilisable
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiIndex** : Ce paramètre contient un descripteur valide sur l'index concerné par l'opération.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetIndexFlags()

Cette fonction renvoie les paramètres d'un index.

Syntaxe API

long AmGetIndexFlags(long hApiIndex);

Syntaxe Basic interne

Function AmGetIndexFlags(hApiIndex As Long) As Long

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiIndex** : Ce paramètre contient un descripteur valide sur l'index concerné par l'opération.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Remarques

La valeur retournée par la fonction est la résultante d'une combinaison logique (OR) des valeurs suivantes :

- 1 : l'index autorise les doublons,
- 2 : l'index autorise la valeur nulle,
- 4 : l'index ne respecte pas la casse.

Ainsi, si la fonction renvoie la valeur 3, vous pouvez en déduire que l'index accepte les doublons et la valeur nulle (1 OR 2 = 3).

AmGetIndexName()

Cette fonction renvoie le nom d'un index.

Syntaxe API

long AmGetIndexName(long hApiIndex, char *pstrBuffer, long lBuffer);

Syntaxe Basic interne

Function AmGetIndexName(hApiIndex As Long) As String

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiIndex** : Ce paramètre contient un descripteur valide sur l'index dont on souhaite connaître le nom.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.

- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetLink()

Cette fonction créée un objet lien à partir du descripteur d'une table et retourne le descripteur de l'objet lien créé.

Syntaxe API

long AmGetLink(long hApiTable, long lPos);

Syntaxe Basic interne

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

Champ d'application

Version : 3.02

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiTable** : Ce paramètre contient un descripteur sur une table.

- **lPos** : Ce paramètre contient la position du lien (son index) à l'intérieur de l'objet.

AmGetLinkCardinality()

Cette fonction renvoie la cardinalité d'un lien.

Syntaxe API

long AmGetLinkCardinality(long hApiField);

Syntaxe Basic interne

Function AmGetLinkCardinality(hApiField As Long) As Long

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiField** : Ce paramètre contient un descripteur sur le lien dont vous souhaitez connaître la cardinalité.

Sortie

- 1 : Le lien est de cardinalité 1-1.
- 2 : Le lien est de cardinalité 1-n.

AmGetLinkCount()

Cette fonction renvoie le nombre de liens contenus dans la table courante.

Syntaxe API

long AmGetLinkCount(long hApiTable);

Syntaxe Basic interne

Function AmGetLinkCount(hApiTable As Long) As Long

Champ d'application

Version : 3.02

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiTable** : Ce paramètre contient un descripteur sur une table valide.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetLinkDstField()

Cette fonction renvoie le champ (clé étrangère) sur lequel pointe le lien définit par le paramètre **hApiField**.

Syntaxe API

long AmGetLinkDstField(long hApiField);

Syntaxe Basic interne

Function AmGetLinkDstField(hApiField As Long) As Long

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiField** : Ce paramètre contient un descripteur sur le lien concerné par l'opération.

AmGetLinkFeatureValue()

Renvoie la valeur d'une caractéristique de type "Lien".

Syntaxe API

long AmGetLinkFeatureValue(long hApiObject, long lFieldPos, long lRecordId);

Syntaxe Basic interne

Function AmGetLinkFeatureValue(hApiObject As Long, lFieldPos As Long, lRecordId As Long) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiObject** : Ce paramètre contient un descripteur sur une requête ou un enregistrement.
- **lFieldPos** : Ce paramètre contient le numéro du champ à l'intérieur de l'objet courant.
- **lRecordId** : Ce paramètre contient le numéro d'identifitant de l'enregistrement dont on veut récupérer la valeur pour la caractéristique.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
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()

Cette fonction créée un objet lien à partir de son nom et retourne le descripteur de l'objet lien créé.

Syntaxe API

long AmGetLinkFromName(long hApiTable, char *strName);

Syntaxe Basic interne

Function AmGetLinkFromName(hApiTable As Long, strName As String) As Long

Champ d'application

Version : 3.02

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiTable** : Ce paramètre contient un descripteur sur une table.
- **strName** : Ce paramètre contient le nom SQL du lien.

AmGetLinkType()

Cette fonction renvoie le type d'un lien.

Syntaxe API

long AmGetLinkType(long hApiField);

Syntaxe Basic interne

Function AmGetLinkType(hApiField As Long) As Long

Champ d'application

Version : 3.02

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiField** : Ce paramètre contient un descripteur sur le lien dont on veut connaître le type.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetMainField()

Cette fonction créée un objet champ, correspondant au champ principal d'une table donnée. Elle renvoie un descripteur sur le champ ainsi créé.

Syntaxe API

long AmGetMainField(long hApiTable);

Syntaxe Basic interne

Function AmGetMainField(hApiTable As Long) As Long

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiTable** : Ce paramètre contient un descripteur sur la table dont on cherche le champ principal.

AmGetMemoField()

Cette fonction créée un objet champ, correspondant au champ de type Memo d'une table donnée. Elle renvoie un descripteur sur le champ ainsi créé.

Syntaxe API

long AmGetMemoField(long hApiTable);

Syntaxe Basic interne

Function AmGetMemoField(hApiTable As Long) As Long

Champ d'application

Version : 4.1.0

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiTable** : Ce paramètre contient un descripteur sur la table dont on cherche le champ Memo.

AmGetNextAssetPin()

L'API AmGetNextAssetPin trouve la broche disponible suivante sur un dispositif (lAssetId). Son numéro de séquence trie les broches. Suivant la direction du port (bPinPortDir), les broches disponibles sont triées par ordre croissant (bPinPortDir = 0) ou décroissant (bPinPortDir = 1).

Syntaxe API

long AmGetNextAssetPin(long hApiCnxBase, long lAssetId, long bPinPortDir, long iPrevPinSeq);

Syntaxe Basic interne

Function AmGetNextAssetPin(lAssetId As Long, bPinPortDir As Long, iPrevPinSeq As Long) As Long

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **lAssetId** : ce paramètre contient l'identifiant du dispositif.
- **bPinPortDir** : ce paramètre est la direction suivant laquelle chercher.
  - 0=croissant
  - 1=décroissant
- **iPrevPinSeq**

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.

- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetNextAssetPort()

L'API AmGetNextAssetPort trouve le port disponible suivant sur un dispositif (lAssetId) fournissant une fonction donnée (lDutyId) ou pas de fonction du tout. L'état du port doit être "Disponible". Des drapeaux booléens précisent si le côté utilisateur (bCheckUser) et/ou le côté hôte (bCheckHost) du port doivent être vérifiés. L'API compare la valeur utilisateur (bUserAvail) et/ou les valeurs hôte (bHostAvail) si le drapeau booléen prend la valeur vraie. Les ports sont triés selon leur numéro de séquence. Suivant la direction du port (bPinPortDir), les ports disponibles sont triés par ordre croissant (bPinPortDir = 0) ou décroissant (bPinPortDir = 1).

Syntaxe API

long AmGetNextAssetPort(long hApiCnxBase, long lAssetId, long lCabCnxTypeId, long lDutyId, long bCheckUser, long bCheckHost, long bUserAvail, long bHostAvail, long bPinPortDir, long iPrevPortSeq);

Syntaxe Basic interne

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

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓

	Utilisable
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **lAssetId** : ce paramètre contient l'identifiant du dispositif à chercher.
- **lCabCnxTypeId** : ce paramètre contient l'identifiant du type de connexion de câble pour le port.
- **lDutyId** : ce paramètre contient l'identifiant de la fonction du port.
- **bCheckUser** : ce paramètre est un drapeau vérifiant le côté utilisateur.
- **bCheckHost** : ce paramètre est un drapeau vérifiant le côté hôte.
- **bUserAvail** : ce paramètre définit l'état de disponibilité du côté utilisateur à vérifier.
- **bHostAvail** : ce paramètre définit l'état de disponibilité du côté hôte à vérifier.
- **bPinPortDir** : ce paramètre définit la direction de broche à vérifier.
  - 0=croissant
  - 1=décroissant
- **iPrevPortSeq**

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetNextCableBundle()

L'API AmGetNextCableBundle trouve le faisceau disponible suivant sur un câble (lCableId) fournissant une fonction donnée (lDutyId) ou pas de fonction du tout. L'état du faisceau doit être "Disponible". Des drapeaux booléens précisent si le côté utilisateur (bCheckUser) et/ou le côté hôte (bCheckHost) du faisceau doivent être vérifiés. L'API compare la valeur utilisateur (bUserAvail) et/ou les valeurs hôte (bHostAvail) si le drapeau booléen prend la valeur vraie.

Syntaxe API

long AmGetNextCableBundle(long hApiCnxBase, long lCableId, long lDutyId, long bCheckUser, long bCheckHost, long bUserAvail, long bHostAvail);

Syntaxe Basic interne

Function AmGetNextCableBundle(lCableId As Long, lDutyId As Long, bCheckUser As Long, bCheckHost As Long, bUserAvail As Long, bHostAvail As Long) As Long

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **lCableId** : ce paramètre contient l'identifiant du câble à vérifier.
- **lDutyId** : ce paramètre contient l'identifiant de la fonction à localiser.
- **bCheckUser** : ce paramètre établit de vérifier la connexion du faisceau du côté utilisateur .
- **bCheckHost** : ce paramètre établit de vérifier la connexion du faisceau du côté hôte.
- **bUserAvail** : ce paramètre définit l'état de connexion du côté utilisateur à localiser.
- **bHostAvail** : ce paramètre définit l'état de connexion du côté hôte à localiser.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetNextCablePair()

L'API AmGetNextCablePair trouve la paire de câble disponible suivante dans un câble (lCableId) d'un type donné (lPairTypeId). Les paires sont triées par identifiant de paire de câble.

Syntaxe API

long AmGetNextCablePair(long hApiCnxBase, long lCableId, long lPairTypeId, long iStartPairSeq);

Syntaxe Basic interne

Function AmGetNextCablePair(lCableId As Long, lPairTypeId As Long, iStartPairSeq As Long) As Long

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **lCableId** : ce paramètre contient l'identifiant du câble à chercher.
- **lPairTypeId** : ce paramètre définit le type de paire du câble à localiser.
- **iStartPairSeq**

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetNTDomains()

Cette fonction permet de récupérer le domaine de l'utilisateur connecté à la base de données.

Syntaxe API

long AmGetNTDomains(char *pstrDomains, long lDomains);

Syntaxe Basic interne

Function AmGetNTDomains() As String

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetNTMachinesInDomain()

Cette fonction permet de récupérer la liste des machines d'un domaine en une colonne (noms des machines séparés par des virgules). Si le domaine est vide, la fonction retourne ERR_CANCEL(2), mais l'exécution n'est pas interrompue.

Syntaxe API

long AmGetNTMachinesInDomain(char *strDomain, char *pstrMachines, long lMachines, long bUseDC);

Syntaxe Basic interne

Function AmGetNTMachinesInDomain(strDomain As String, bUseDC As Long) As String

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strDomain** : Ce paramètre contient le nom du domaine à explorer.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetNTUsersInDomain()

Cette fonction permet de récupérer la liste des utilisateurs sur un domaine. La liste est retournée en deux colonnes (login,fullname). '|' est utilisé comme séparateur de colonnes, ',' comme séparateur de lignes.

Syntaxe API

long AmGetNTUsersInDomain(char *strDomain, char *pstrUsers, long lUsers);

Syntaxe Basic interne

Function AmGetNTUsersInDomain(strDomain As String) As String

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **strDomain** : Ce paramètre contient le nom du domaine à explorer.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetPOLinePrice()

Cette fonction permet de calculer le prix d'une ligne de commande.

Syntaxe API

double AmGetPOLinePrice(long hApiCnxBase, long lPOrdLineId);

Syntaxe Basic interne

Function AmGetPOLinePrice(lPOrdLineId As Long) As Double

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓

	Utilisable
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lPOrdLineId** : Ce paramètre contient l'identifiant de la ligne de commande.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetPOLinePriceCur()

Cette fonction permet de retrouver le code devise applicable à une ligne de commande

Syntaxe API

long AmGetPOLinePriceCur(long hApiCnxBase, long lPOrdLineId, char *pstrPrice, long lPrice);

Syntaxe Basic interne

Function AmGetPOLinePriceCur(lPOrdLineId As Long) As String

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lPOrdLineId** : Ce paramètre contient l'identifiant de la ligne de commande.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetPOLineReference()

Cette fonction permet de récupérer le libellé de la référence catalogue correspondant à la ligne de commande.

Syntaxe API

long AmGetPOLineReference(long hApiCnxBase, long lPOrdLineId, char *pstrRef, long lRef);

Syntaxe Basic interne

Function AmGetPOLineReference(lPOrdLineId As Long) As String

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lPOrdLineId** : Ce paramètre contient l'identifiant de la ligne de commande.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetRecordFromMainId()

Cette fonction renvoie le numéro d'identifiant d'un enregistrement identifié par une valeur de la clé primaire de la table contenant cet enregistrement.

Syntaxe API

long AmGetRecordFromMainId(long hApiCnxBase, char *strTable, long lId);

Syntaxe Basic interne

Function AmGetRecordFromMainId(strTable As String, lId As Long) As Long

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **strTable** : Ce paramètre contient le nom SQL de la table contenant l'enregistrement concerné.
- **lId** : Ce paramètre contient la valeur de la clé primaire de la table pour cet enregistrement.

Remarques

Cette fonction renvoie systématiquement un descripteur d'enregistrement, excepté lorsque la table n'existe pas. S'il n'existe aucun enregistrement dans la table spécifiée, une erreur surviendra à chaque nouvelle exécution de fonction utilisant le descripteur renvoyé par cette fonction.

AmGetRecordHandle()

Cette fonction renvoie le descripteur d'un enregistrement qui est le résultat courant d'une requête identifiée par son descripteur. Cet enregistrement pourra être utilisé pour écrire dans la base de données. Cette fonction n'est opérante que si la requête contient la clé primaire de l'enregistrement.

Syntaxe API

long AmGetRecordHandle(long hApiQuery);

Syntaxe Basic interne

Function AmGetRecordHandle(hApiQuery As Long) As Long

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiQuery** : Ce paramètre contient un descripteur valide sur un objet requête.

AmGetRecordId()

Cette fonction renvoie le numéro d'identifiant d'un enregistrement identifié par son descripteur. Dans le cas d'un enregistrement en cours d'insertion, cette valeur sera 0.

Syntaxe API

long AmGetRecordId(long hApiRecord);

Syntaxe Basic interne

Function AmGetRecordId(hApiRecord As Long) As Long

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiRecord** : Ce paramètre contient un descripteur valide sur l'enregistrement dont on souhaite connaître le numéro d'identifiant.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetRelDstField()

Cette fonction renvoie un descripteur sur le champ destination d'un lien.

Syntaxe API

long AmGetRelDstField(long hApiField);

Syntaxe Basic interne

Function AmGetRelDstField(hApiField As Long) As Long

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiField** : Ce paramètre contient un descripteur valide sur le lien concerné par l'opération.

AmGetRelSrcField()

Cette fonction renvoie un descripteur sur le champ source d'un lien.

Syntaxe API

long AmGetRelSrcField(long hApiField);

Syntaxe Basic interne

Function AmGetRelSrcField(hApiField As Long) As Long

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiField** : Ce paramètre contient un descripteur valide sur le lien concerné par l'opération.

AmGetRelTable()

Cette fonction renvoie un descripteur sur la table de relation d'un lien de cardinalité N-N.

Syntaxe API

long AmGetRelTable(long hApiField);

Syntaxe Basic interne

Function AmGetRelTable(hApiField As Long) As Long

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiField** : Ce paramètre contient un descripteur valide sur le lien concerné par l'opération.

Sortie

En cas d'erreur, cette fonction retourne un descripteur non valide (de valeur nulle).

AmGetReverseLink()

Cette fonction renvoie le descripteur du lien inverse du lien spécifié par le descripteur contenu dans le paramètre **hApiField**.

Syntaxe API

long AmGetReverseLink(long hApiField);

Syntaxe Basic interne

Function AmGetReverseLink(hApiField As Long) As Long

Champ d'application

Version : 3.02

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓

	Utilisable
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiField** : Ce paramètre contient un descripteur sur le lien dont on veut connaître le lien inverse.

AmGetSelfFromMainId()

Renvoie la chaîne de description pour un enregistrement d'une table donnée.

Syntaxe API

long AmGetSelfFromMainId(long hApiCnxBase, char *strTableName, long lId, char *pstrRecordDesc, long lRecordDesc);

Syntaxe Basic interne

Function AmGetSelfFromMainId(strTableName As String, lId As Long) As String

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **strTableName** : Ce paramètre contient le nom SQL de la table contenant l'enregistrement concerné par l'opération.
- **lId** : Ce paramètre contient le numéro d'identifiant de l'enregistrement concerné par l'opération.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetSourceTable()

Renvoie le descripteur de la table source du lien indiqué dans le paramètre **hApiField**.

Syntaxe API

long AmGetSourceTable(long hApiField);

Syntaxe Basic interne

Function AmGetSourceTable(hApiField As Long) As Long

Champ d'application

Version : 3.02

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiField** : Ce paramètre contient un descripteur valide sur le lien dont on veut connaître la table source.

Sortie

En cas d'erreur, cette fonction retourne un descripteur non valide (de valeur nulle).

AmGetTable()

Cette fonction renvoie le descripteur d'une table identifiée par sa position (son numéro) dans la connexion courante.

Syntaxe API

long AmGetTable(long hApiCnxBase, long lPos);

Syntaxe Basic interne

Function AmGetTable(lPos As Long) As Long

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **lPos** : Ce paramètre contient la position de la table dans la connexion courante. Ses valeurs sont comprises entre "0" et **AmGetTableCount**.

Sortie

En cas d'erreur, cette fonction retourne un descripteur non valide (de valeur nulle).

AmGetTableCount()

Cette fonction renvoie le nombre de tables de la base de données sur laquelle porte la connexion courante.

Syntaxe API

long AmGetTableCount(long hApiCnxBase);

Syntaxe Basic interne

Function AmGetTableCount() As Long

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetTableDescription()

Cette fonction renvoie, sous la forme d'une chaîne de caractères (format "String"), la description longue d'une table identifiée par un descripteur.

Syntaxe API

long AmGetTableDescription(long hApiTable, char *pstrDesc, long lDesc);

Syntaxe Basic interne

Function AmGetTableDescription(hApiTable As Long) As String

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiTable** : Ce paramètre contient un descripteur valide sur la table dont on souhaite connaître la description longue.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetTableFromName()

Cette fonction renvoie le descripteur d'une table identifiée par son nom SQL dans la connexion courante.

Syntaxe API

long AmGetTableFromName(long hApiCnxBase, char *strName);

Syntaxe Basic interne

Function AmGetTableFromName(strName As String) As Long

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strName** : Ce paramètre contient la position le nom SQL de la table dont on veut récupérer le descripteur.

Sortie

En cas d'erreur, cette fonction retourne un descripteur non valide (de valeur nulle).

AmGetTableLabel()

Cette fonction renvoie, sous la forme d'une chaîne de caractères (format "String"), le libellé d'une table identifiée par un descripteur.

Syntaxe API

long AmGetTableLabel(long hApiTable, char *pstrLabel, long lLabel);

Syntaxe Basic interne

Function AmGetTableLabel(hApiTable As Long) As String

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiTable** : Ce paramètre contient un descripteur valide sur la table dont on souhaite connaître le libellé.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetTableName()

Renvoie le nom SQL d'une table sous la forme d'une chaîne de caractères.

Syntaxe API

long AmGetTableName(long hApiTable, char *pstrBuffer, long lBuffer);

Syntaxe Basic interne

Function AmGetTableName(hApiTable As Long) As String

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiTable** : Descripteur valide sur la table dont vous souhaitez récupérer le nom.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetTableRights()

Cette fonction renvoie, sous la forme d'une chaîne de caractères (format "String"), les droits utilisateur sur une table identifiée par un descripteur. La chaîne renvoyée est composée au maximum de deux caractères qui indiquent l'état des droits en création et en destruction :

- "c" indique que l'utilisateur possède les droits en création sur la table.
- "d" indique que l'utilisateur possède les droits en destruction sur la table.

Ainsi, par exemple :

- " c" indique que l'utilisateur possède uniquement un droit en création sur la table.
- "cd" indique que l'utilisateur possède les droits en création et en destruction sur la table.

Syntaxe API

long AmGetTableRights(long hApiTable, char *pstrBuffer, long lBuffer);

Syntaxe Basic interne

Function AmGetTableRights(hApiTable As Long) As String

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiTable** : Ce paramètre contient un descripteur valide sur la table pour laquelle on souhaite connaître les droits utilisateurs.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetTableSqlName()

Cette fonction renvoie, sous la forme d'une chaîne de caractères (format "String"), le nom SQL d'une table identifiée par un descripteur.

Syntaxe API

long AmGetTableSqlName(long hApiTable, char *pstrBuffer, long lBuffer);

Syntaxe Basic interne

Function AmGetTableSqlName(hApiTable As Long) As String

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiTable** : Ce paramètre contient un descripteur valide sur la table dont on souhaite connaître le nom SQL.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetTargetTable()

Retourne le nom SQL de la table de destination d'un lien.

Syntaxe API

long AmGetTargetTable(long hApiField);

Syntaxe Basic interne

Function AmGetTargetTable(hApiField As Long) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiField** : Descripteur sur le lien concerné par l'opération.

Sortie

En cas d'erreur, cette fonction retourne un descripteur non valide (de valeur nulle).

AmGetTrace()

L'API AmGetTrace détecte la chaîne de liaisons entre deux noeuds (lUserId, lHostId) dans la table des liaisons de câbles. La direction de la chaîne de liaisons (iTraceDir) précise si la chaîne de liaisons doit être utilisateur vers hôte (iTraceDir = 1) ou hôte vers utilisateur (iTraceDir = 0). Le type de chaîne de liaisons (iTraceType) indique si la chaîne de liaisons est une connexion (iTraceType = 1) ou une déconnexion (iTraceType = 2). L'indicateur de chaîne de liaisons complète (bFullTrace) précise si la chaîne de liaisons inclut seulement des noeuds modifiés (bFullTrace=0) ou la chaîne de liaisons complète (bFullTrace=1)

Syntaxe API

long AmGetTrace(long hApiCnxBase, long lUserId, long lHostId, long iTraceDir, long iTraceType, long bFullTrace, char *pstrTrace, long lTrace);

Syntaxe Basic interne

Function AmGetTrace(lUserId As Long, lHostId As Long, iTraceDir As Long, iTraceType As Long, bFullTrace As Long) As String

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **lUserId** : ce paramètre contient l'identifiant de la liaison de connexion de départ.
- **lHostId** : ce paramètre contient l'identifiant de la liaison de connexion d'arrivée.
- **iTraceDir** : ce paramètre précise la direction de la connexion.
  - 0=hôte vers utilisateur
  - 1=utilisateur vers hôte
- **iTraceType** : ce paramètre précise le type de connexion.
  - 1=connexion
  - 2=déconnexion

- **bFullTrace** : ce paramètre spécifie d'ignorer la chaîne de liaisons partielle et de retourner la chaîne de la chaîne de liaisons entière.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetTraceFromHist()

L'API AmGetTraceFromHist sert à calculer une chaîne de caractères à partir de l'historique de chaîne de liaisons et à l'aide des opérations sur chaîne de liaisons afin de distinguer la connectivité nouvelle de celle existante.

Syntaxe API

long AmGetTraceFromHist(long hApiCnxBase, long lProjTraceOutId, long iTraceDir, char *strDelimiter, char *pstrTraceint, long lTraceint, long bUpdateFlag);

Syntaxe Basic interne

Function AmGetTraceFromHist(lProjTraceOutId As Long, iTraceDir As Long, strDelimiter As String, bUpdateFlag As Long) As String

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lProjTraceOutId** : ce paramètre contient l'identifiant de la chaîne de liaisons du projet.
- **iTraceDir** : ce paramètre précise la direction de la connexion.
  - 0=hôte vers utilisateur
  - 1=utilisateur vers hôte
- **strDelimiter** : ce paramètre est le délimiteur de chaînes qui montre les connexions et déconnexions existantes.
- **bUpdateFlag** : ce paramètre optionnel met à jour le champ amCabTraceOut.TraceString.
  - 0=faux
  - 1=vrai

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmGetTypedLinkField()

Renvoie un descripteur sur le champ dont la valeur est le nom SQL de la table de destination du lien typé indiqué dans le paramètre **hApiField**.

Syntaxe API

long AmGetTypedLinkField(long hApiField);

Syntaxe Basic interne

Function AmGetTypedLinkField(hApiField As Long) As Long

Champ d'application

Version : 3.02

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiField** : Ce paramètre contient un descripteur valide sur le lien typé à l'origine de l'opération.

AmGetVersion()

Cette fonction renvoie le numéro de compilation de la version d'AssetCenter sous la forme d'une chaîne de caractères.

Syntaxe API

long AmGetVersion(char *pstrBuf, long lBuf);

Syntaxe Basic interne

Function AmGetVersion() As String

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmHasAdminPrivilege()

Cette fonction renvoie la valeur "TRUE" (valeur différente de 0) si l'utilisateur connecté possède les droits administratifs.

Syntaxe API

long AmHasAdminPrivilege(long hApiCnxBase);

Syntaxe Basic interne

Function AmHasAdminPrivilege() As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmHasRelTable()

Cette fonction permet de tester si un lien possède ou non une table de relation.

Syntaxe API

long AmHasRelTable(long hApiField);

Syntaxe Basic interne

Function AmHasRelTable(hApiField As Long) As Long

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiField** : Ce paramètre contient un descripteur valide sur le lien concerné par l'opération.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmHasRightsForCreation()

Cette fonction permet de déterminer si l'utilisateur connecté possède les droits en création sur une table donnée.

Syntaxe Basic interne

Function AmHasRightsForCreation(strTable As String) As Long

Champ d'application

Version : 4.3.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strTable** : Ce paramètre contient le nom SQL de la table concernée par l'opération.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
RetVal=amHasRightsForCreation("amEmplDept")
```

AmHasRightsForDeletion()

Cette fonction permet de déterminer si l'utilisateur connecté possède les droits en destruction sur une table donnée.

Syntaxe Basic interne

Function AmHasRightsForDeletion(strTable As String) As Long

Champ d'application

Version : 4.3.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strTable** : Ce paramètre contient le nom SQL de la table concernée par l'opération.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.

- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
RetVal=amHasRightsForDeletion("amEmplDept")
```

AmHasRightsForFieldUpdate()

Cette fonction permet de déterminer si l'utilisateur connecté possède les droits en mise à jour sur un champ donné.

Syntaxe Basic interne

Function AmHasRightsForFieldUpdate(strTable As String, strField As String) As Long

Champ d'application

Version : 4.3.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **strTable** : Ce paramètre contient le nom SQL de la table concernée par l'opération.
- **strField** : Ce paramètre contient le nom SQL du champ (de la table précisée dans le paramètre **strTable**) concerné par l'opération.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
RetVal=amHasRightsForFieldUpdate("amEmplDept", "Location")
```

AmHelpdeskCanCloseFile()

Cette fonction permet de déterminer si l'utilisateur connecté peut ou non clore un dossier de support.

Syntaxe Basic interne

Function AmHelpdeskCanCloseFile() As Long

Champ d'application

Version : 4.3.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Remarques

Note :

Si l'utilisateur connecté peut clore un dossier de support, la fonction renvoie la valeur "1".

AmHelpdeskCanProceed()

Cette fonction permet de déterminer si l'utilisateur connecté peut ou non poursuivre la résolution d'un dossier de support.

Syntaxe Basic interne

Function AmHelpdeskCanProceed() As Long

Champ d'application

Version : 4.3.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Remarques

Note :

Si l'utilisateur connecté peut poursuivre la résolution d'un dossier de support, la fonction renvoie la valeur "1".

AmHelpdeskCanSaveCall()

Cette fonction permet de déterminer si l'utilisateur connecté peut ou non enregistrer un dossier de support.

Syntaxe Basic interne

Function AmHelpdeskCanSaveCall() As Long

Champ d'application

Version : 4.3.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Remarques

Note :

Si l'utilisateur connecté peut enregistrer un dossier de support, la fonction renvoie la valeur "1".

AmImportDocument()

Cette fonction crée et importe un document depuis un fichier.

Syntaxe API

long AmImportDocument(long hApiCnxBase, long lDocObjId, char *strTableName, char *strFileName, char *strCategory, char *strDesignation);

Syntaxe Basic interne

Function AmImportDocument(lDocObjId As Long, strTableName As String, strFileName As String, strCategory As String, strDesignation As String) As Long

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **lDocObjId** : Ce paramètre contient la valeur qui sera stockée dans le champ lDocObjId de la table amDocument.
- **strTableName** : Ce paramètre contient la valeur qui sera stockée dans le champ DocObjTable de la table amDocument. En pratique il s'agit du nom SQL de la table contenant l'enregistrement auquel le document est attaché.
- **strFileName** : Ce paramètre contient le nom du fichier à importer.
- **strCategory** : Ce paramètre contient la catégorie du document, telle qu'elle apparaît sous AssetCenter.
- **strDesignation** : Ce paramètre contient la désignation du document, telle qu'elle apparaît dans AssetCenter.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmImportReport()

Cette fonction permet d'importer un rapport Crystal à partir d'un fichier. L'import s'effectue dans un enregistrement existant de la table **amReport** de la base de données.

Syntaxe Basic interne

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

Champ d'application

Version : 4.3.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lReportId** : Ce paramètre contient l'identifiant de l'enregistrement de la table **amReport** dans lequel le rapport importé sera stocké.
- **strFileName** : Ce paramètre contient le nom complet du fichier contenant le rapport à importer.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmIncrementLogLevel()

Cette fonction affiche le message **strMsg** dans une fenêtre d'historique et créée un noeud dans la page finale d'un assistant.

Tous les prochains messages apparaîtront sous ce noeud.

Syntaxe Basic interne

Function AmIncrementLogLevel(strMsg As String, iType As Long) As Long

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **strMsg** : Ce paramètre contient le texte du message à afficher.
- **iType** : Ce paramètre définit l'icône associée au message. Les valeurs possibles sont "1" pour une erreur, "2" pour un avertissement et "4" pour une information.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmInsertRecord()

Cette fonction insère un enregistrement précédemment créé dans la base de données. Seuls les enregistrements créés au moyen de la fonction **AmCreateRecord** peuvent être insérés dans la base de données. Les enregistrements accédés au moyen d'une requête ne peuvent être insérés.

Syntaxe API

long AmInsertRecord(long hApiRecord);

Syntaxe Basic interne

Function AmInsertRecord(hApiRecord As Long) As Long

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiRecord** : Ce paramètre contient un descripteur sur l'enregistrement que vous souhaitez insérer dans la base de données.

Sortie

- 0 : La fonction s'est exécutée normalement.

- Non nul : Code d'erreur.

AmInstantiateReqLine()

Cette fonction permet d'instancier directement une ligne de demande donnée.

Syntaxe API

long AmInstantiateReqLine(long hApiCnxBase, long lRequestLineId, long bFinal, long lPOrderLineId, double dQty);

Syntaxe Basic interne

Function AmInstantiateReqLine(lRequestLineId As Long, bFinal As Long, lPOrderLineId As Long, dQty As Double) As Long

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lRequestLineId** : Ce paramètre contient l'identifiant de la ligne de demande.
- **bFinal** : Ce paramètre permet de préciser si oui ou non vous souhaitez finaliser l'affectation.

- **lPOrderLineId** : Ce paramètre contient l'identifiant de la ligne de commande.
- **dQty** : Ce paramètre contient la quantité à instancier.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

Remarques

La fonction permet de créer les éléments demandés sans passer par le cycle d'achat. Si bFinal = FALSE, alors l'élément sera créé avec l'état d'affectation En attente de réception.

AmInstantiateRequest()

Cette fonction permet d'instancier directement le contenu complet d'une demande donnée.

Syntaxe API

long AmInstantiateRequest(long hApiCnxBase, long lRequestId, long lMulFactor);

Syntaxe Basic interne

Function AmInstantiateRequest(lRequestId As Long, lMulFactor As Long) As Long

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lRequestId** : Ce paramètre contient l'identifiant de demande.
- **lMulFactor** : Ce paramètre permet de préciser le nombre d'instanciations à effectuer.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmIsConnected()

Cette fonction teste si la connexion courante est valide.

Syntaxe API

long AmIsConnected(long hApiCnxBase);

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	

	Utilisable
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmIsFieldForeignKey()

Cette fonction permet de déterminer si un champ est une clé étrangère de la base de données.

Syntaxe API

long AmIsFieldForeignKey(long hApiField);

Syntaxe Basic interne

Function AmIsFieldForeignKey(hApiField As Long) As Long

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓

	Utilisable
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiField** : Ce paramètre contient un descripteur sur le champ qui doit être identifié.

Sortie

- 1 : Le champ est une clé étrangère.
- 0 : Le champ n'est pas une clé étrangère.

AmIsFieldIndexed()

Cette fonction permet de déterminer si un champ est indexé ou non.

Syntaxe API

long AmIsFieldIndexed(long hApiField);

Syntaxe Basic interne

Function AmIsFieldIndexed(hApiField As Long) As Long

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiField** : Ce paramètre contient un descripteur sur le champ qui doit être identifié.

Sortie

- 1 : Le champ est indexé.
- 0 : Le champ n'est pas indexé.

AmIsFieldPrimaryKey()

Cette fonction permet de déterminer si un champ est une clé primaire de la base de données.

Syntaxe API

long AmIsFieldPrimaryKey(long hApiField);

Syntaxe Basic interne

Function AmIsFieldPrimaryKey(hApiField As Long) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiField** : Ce paramètre contient un descripteur sur le champ qui doit être identifié.

Sortie

- 1 : Le champ est une clé primaire.
- 0 : Le champ n'est pas une clé primaire.

AmIsHelpdeskAdmin()

Cette fonction permet de déterminer si l'utilisateur connecté est ou non administrateur de support.

Syntaxe Basic interne

Function AmIsHelpdeskAdmin() As Long

Champ d'application

Version : 4.3.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓

	Utilisable
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Remarques

Note :

Si l'utilisateur connecté est administrateur de support, la fonction renvoie la valeur "1".

AmIsHelpdeskMember()

Cette fonction permet de déterminer si l'utilisateur connecté fait ou non partie d'un groupe de support.

Syntaxe Basic interne

Function AmIsHelpdeskMember() As Long

Champ d'application

Version : 4.3.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Remarques

Note :

Si l'utilisateur connecté fait partie d'un groupe de support, la fonction renvoie la valeur "1".

AmIsHelpdeskSuper()

Cette fonction permet de déterminer si l'utilisateur connecté est ou non responsable d'un groupe de support.

Syntaxe Basic interne

Function AmIsHelpdeskSuper() As Long

Champ d'application

Version : 4.3.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Remarques

Note :

Si l'utilisateur connecté est responsable d'un groupe de support, la fonction renvoie la valeur "1".

AmIsLink()

Détermine si l'objet identifié par son descripteur est un lien ou un champ.

Syntaxe API

long AmIsLink(long hApiField);

Syntaxe Basic interne

Function AmIsLink(hApiField As Long) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiField** : Descripteur sur l'objet concerné par l'opération.

Sortie

- 1 : L'objet est un lien.
- 0 : L'objet est un champ.

AmIsModuleAuthorized()

Cette fonction permet de déterminer si l'utilisateur connecté a ou non accès à module donné de l'application.

Syntaxe Basic interne

Function AmIsModuleAuthorized(strModuleName As String) As Long

Champ d'application

Version : 4.3.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strModuleName** : Ce paramètre contient le nom du module concerné par l'opération. La liste des modules possibles est la suivante :
  - ITAM : module de gestion du parc
  - Reconc : module de réconciliation
  - Contract : module de gestion des contrats
  - Leasing : module de gestion du leasing
  - Procurement : module de gestion des achats
  - Finance : module de gestion des coûts
  - Helpdesk : module de gestion du Helpdesk
  - Cable : module de gestion du cablage
  - Barcode : module code à barres
  - Admin : module d'administration
  - Visio : module d'intégration Visio
  - API : bibliothèque dynamique de fonctions
  - Wizard : module de gestion des assistants
  - Workflow : module de gestion des workflows
  - AutoCAD : module d'intégration AutoCAD

- Knowlix : module d'intégration Knowlix
- DA_Automation : module d'automatisation
- DA_RemoteControl : module de prise de contrôle à distance

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Remarques


Note :

Tous les modules ne sont pas accessibles ou activables à partir de l'application. La disponibilité de certains modules est conditionnée par le type de licence acquis auprès de Peregrine Systems, Inc.

AmIsTypedLink()

Détermine si l'objet identifié par son descripteur est un lien typé ou non.

Syntaxe API

long AmIsTypedLink(long hApiField);

Syntaxe Basic interne

Function AmIsTypedLink(hApiField As Long) As Long

Champ d'application

Version : 3.02

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiField** : Descripteur sur l'objet concerné par l'opération.

Sortie

- 1 : L'objet est un lien typé.
- 0 : L'objet n'est pas un lien typé.

AmLastError()

Cette fonction renvoie le dernier code d'erreur généré par la dernière fonction exécutée dans le contexte de la connexion correspondante.

Syntaxe API

long AmLastError(long hApiCnxBase);

Syntaxe Basic interne

Function AmLastError() As Long

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmLastErrorMsg()

Cette fonction renvoie le dernier message d'erreur survenu lors de la connexion courante.

Syntaxe API

long AmLastErrorMsg(long hApiCnxBase, char *pstrBuffer, long lBuffer);

Syntaxe Basic interne

Function AmLastErrorMsg() As String

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmListToString()

Cette fonction convertit le résultat d'une chaîne de caractères obtenue au moyen de la fonction **AmDbGetList** en une chaîne de caractères affichable telle qu'elle apparaîtrait avec la fonction **AmDbGetString**.

Syntaxe API

long AmListToString(char *return, long lreturn, char *strSource, char *strColSep, char *strLineSep, char *strIdSep);

Syntaxe Basic interne

Function AmListToString(strSource As String, strColSep As String, strLineSep As String, strIdSep As String) As String

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strSource** : Ce paramètre contient la chaîne de caractères à convertir.
- **strColSep** : Ce paramètre contient le caractère utilisé comme séparateur de colonnes dans la chaîne à convertir.
- **strLineSep** : Ce paramètre contient le caractère utilisé comme séparateur de lignes dans la chaîne à convertir.
- **strIdSep** : Ce paramètre contient le caractère utilisé comme séparateur d'identifiant dans la chaîne à convertir.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmLog()

Cette fonction affiche le message **strMessage** dans une fenêtre d'historique.

Syntaxe Basic interne

Function AmLog(strMessage As String, iLogType As Long) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strMessage** : Ce paramètre contient le texte du message à afficher.
- **iLogType** : Ce paramètre définit l'icône associée au message. Les valeurs possibles sont "1" pour une erreur, "2" pour un avertissement et "4" pour une information.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

Exemple

```
AmLog("Ceci est un message")
```

AmLoginId()

Cette fonction renvoie l'identifiant de l'utilisateur connecté.

Syntaxe API

long AmLoginId(long hApiCnxBase);

Syntaxe Basic interne

Function AmLoginId() As Long

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.

- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

L'exemple suivant définit l'identifiant de l'utilisateur connecté comme valeur par défaut pour un champ de la base de données :

```
RetVal=AmLoginId()
```

AmLoginName()

Cette fonction renvoie le nom de login de l'utilisateur connecté.

Syntaxe API

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

Syntaxe Basic interne

Function AmLoginName() As String

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

L'exemple suivant définit le nom de login de l'utilisateur connecté comme valeur par défaut pour un champ de la base de données :

```
RetVal=AmLoginName()
```

AmMapSubReqLineAgent()

Cette fonction permet d'établir les liens possibles entre les sous-lignes d'une ligne de demande et celles d'une ligne de commande.

Syntaxe API

long AmMapSubReqLineAgent(long hApiCnxBase, long lRequestLineId, long lPorderLineId);

Syntaxe Basic interne

Function AmMapSubReqLineAgent(lRequestLineId As Long, lPorderLineId As Long) As Long

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lRequestLineId** : Ce paramètre contient l'identifiant de la ligne de demande.
- **lPorderLineId** : Ce paramètre contient l'identifiant de la ligne de demande.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmMoveCable()

L'API AmMoveCable transfère un câble (lCableId) de sa localisation actuelle à une localisation de destination donnée (lToLocId). Si le projet (lProjectId) et l'intervention (lWorkOrderId) prennent des valeurs, le câble est ajouté au projet et à l'intervention avec le commentaire donné (strComment). Ce commentaire décrit l'action qui sera accomplie sur le câble (i.e. "Transférer le câble d'ici jusqu'à là").

Syntaxe API

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

Syntaxe Basic interne

Function AmMoveCable(lCableId As Long, lToLocId As Long, lProjectId As Long, lWorkOrderId As Long, strComment As String) As Long

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lCableId** : ce paramètre contient l'identifiant du câble à transférer.
- **lToLocId** : ce paramètre contient l'identifiant de la nouvelle localisation du câble.
- **lProjectId** : ce paramètre contient l'identifiant du projet.
- **lWorkOrderId** : ce paramètre contient l'identifiant de l'intervention.
- **strComment** : ce paramètre est le commentaire qui sera joint à l'intervention.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmMoveDevice()

L'API AmMoveDevice transfère un dispositif (lDeviceId) de sa localisation actuelle jusqu'à une localisation de destination donnée (lToLocationId). Si le projet (lProjectId) et l'intervention (lWorkOrderId) prennent des valeurs, le dispositif est ajouté au projet et à l'intervention avec le commentaire donné (strComment). Ce commentaire décrit l'action qui sera accomplie sur le dispositif (i.e. "Transférer le dispositif d'ici jusqu'à là").

Syntaxe API

long AmMoveDevice(long hApiCnxBase, long lDeviceId, long lToLocationId, long lProjectId, long lWorkOrderId, char *strComment);

Syntaxe Basic interne

Function AmMoveDevice(lDeviceId As Long, lToLocationId As Long, lProjectId As Long, lWorkOrderId As Long, strComment As String) As Long

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lDeviceId** : ce paramètre contient l'identifiant du dispositif qui sera transféré.

- **lToLocationId** : ce paramètre contient l'identifiant de la nouvelle localisation du dispositif.
- **lProjectId** : ce paramètre contient l'identifiant du projet.
- **lWorkOrderId** : ce paramètre contient l'identifiant de l'intervention.
- **strComment** : ce paramètre est le commentaire qui sera joint à l'intervention.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmMsgBox()

Cette fonction affiche une boîte de dialogue contenant un message.

Syntaxe Basic interne

Function AmMsgBox(strMessage As String, lMode As Long) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **strMessage** : Ce paramètre contient le message affiché dans la boîte de dialogue.
- **lMode** : Ce paramètre contient le type de boîte de dialogue affiché (O pour une boîte simple avec un bouton OK, 1 pour une boîte avec les boutons OK et Annuler, 2 pour une boîte avec le seul bouton Annuler).

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

Exemple

```
AmMsgBox("Déménagement effectué")
```

AmOpenConnection()

Crée une connection sur une base de données AssetCenter. **strDataSource** doit être une source de données valide (les sources de données apparaissent dans la boîte de connexion d'AssetCenter).

Vous pouvez ouvrir plusieurs connexions sur une même base ou sur des bases de données différentes.

Syntaxe API

long AmOpenConnection(char *strDataSource, char *strUser, char *strPwd);

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strDataSource** : Nom de la source de données pour la connexion.
- **strUser** : Nom de l'utilisateur pour le connexion.
- **strPwd** : Mot de passe de l'utilisateur sur la base de données.

AmOpenScreen()

Cette fonction permet d'ouvrir un écran sous AssetCenter.

Syntaxe Basic interne

Function AmOpenScreen(strScreenId As String, strContext As String, strFilter As String, iMode As Long, strBindField As String, bStayReadOnly As Long) As Long

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	

	Utilisable
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strScreenId** : Ce paramètre contient le nom SQL de la vue de l'écran système ou utilisateur que vous souhaitez ouvrir (dans cet ordre de priorité).
- **strContext** : Ce paramètre optionnel contient la liste des identifiants des enregistrements sélectionnés dans la liste à l'ouverture de l'écran.
- **strFilter** : Ce paramètre contient un filtre AQL appliqué sur la liste à l'ouverture de l'écran.
- **iMode** : Ce paramètre contient le mode d'ouverture de l'écran : consultation, édition, etc. Les valeurs possibles sont : 0 (Pas d'action en cours), 1 (Pas d'action en cours), 2 (Modification en cours), 3 (Création en cours), 4 (Duplication en cours), 5 (Ajout en cours), 6 (Choix en cours).
- **strBindField** : Ce paramètre permet d'ouvrir un écran avec un filtre et un mode comme pour l'ouverture d'une fenêtre liée. Il prend le nom SQL du champ source ou bien la valeur CurrentSrcChoice pour utiliser le contexte en cours.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmOverflowTables()

Cette fonction renvoie les noms SQL des tables de débordement d'une table donnée.

Syntaxe API

long AmOverflowTables(long hApiCnxBase, char *strBasisTable, char *strOverflowTables, long lOverflowTables);

Syntaxe Basic interne

Function AmOverflowTables(strBasisTable As String) As String

Champ d'application

Version : 4.3.0

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strBasisTable** : Ce paramètre contient le nom SQL de la table concernée par l'opération.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Remarques

Note :

La virgule est utilisée comme séparateur dans la liste renvoyée par la fonction. Si aucune table de débordement n'existe pour une table donnée, la fonction renvoie une chaîne vide.

Exemple

L'exemple suivant renvoie les tables de débordement de la table des éléments de parc (amPortfolio) :

```
RetVal = AmOverflowTables("amPortfolio")
```

Le résultat de cet exemple est :

```
amComputer,amSoftInstall,amPhone
```

AmPagePath()

Cette fonction renvoie, sous la forme d'une chaîne, le chemin de l'assistant, c'est-à-dire la liste des pages parcourues sans tenir compte des retours en arrière.

Syntaxe Basic interne

Function AmPagePath() As String

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	

	Utilisable
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmProgress()

Cette fonction affiche, dans la page finale d'un assistant, une barre de progression représentant un pourcentage.

Syntaxe Basic interne

Function AmProgress(iProgress As Long) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **iProgress** : Ce paramètre contient le pourcentage (entre 0 et 100) de complétion qui détermine la taille de la barre de progression.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

Exemple

```
AmProgress(85)
```

Cette fonction affiche une barre de progression représentant un pourcentage de 85%.

AmPurgeRecord()

Cette fonction détruit un enregistrement.

Syntaxe API

long AmPurgeRecord(long hApiRecord);

Syntaxe Basic interne

Function AmPurgeRecord(hApiRecord As Long) As Long

Champ d'application

Version : 4.3.0

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiRecord** : Ce paramètre contient le descripteur sur l'enregistrement concerné par l'opération.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

Remarques

Note :

Le traitement des enregistrements liés dépend du type de lien. Dans le cas d'un lien de type OWN, les enregistrements liés sont traités à l'identique. Dans le cas d'un lien DEFINE ou NORMAL, les clés étrangères des enregistrements liés sont remises à 0 et les champs d'archivage sont renseignés avec l'identifiant de l'enregistrement archivé et sa chaîne de description.

IMPORTANT :

Cette fonction est disponible pour un enregistrement provenant d'une table d'archivage ou d'une table standard.

AmQueryCreate()

Cette fonction créée un objet requête dans la connexion courante. Cet objet peut ensuite être utilisé pour envoyer des commandes AQL au serveur de base de données.

Syntaxe API

long AmQueryCreate(long hApiCnxBase);

Syntaxe Basic interne

Function AmQueryCreate() As Long

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

AmQueryExec()

Cette fonction exécute une requête AQL. Elle renvoie le premier résultat de la requête. Le résultat suivant peut être obtenu au moyen de la fonction **AmQueryNext**.

Lorsque la requête transmise par cette fonction renvoie un champ de type "Memo" (enregistrement de la table de nom SQL amComment), la taille de ce dernier est tronquée à 255 caractères.

Syntaxe API

long AmQueryExec(long hApiQuery, char *strQueryCommand);

Syntaxe Basic interne

Function AmQueryExec(hApiQuery As Long, strQueryCommand As String) As Long

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiQuery** : Ce paramètre contient un descripteur valide sur l'objet requête auquel sont transmises les commandes AQL.
- **strQueryCommand** : Ce paramètre contient le corps de la requête AQL sous forme de chaîne.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmQueryGet()

Cette fonction exécute une requête AQL sans curseur (un seul résultat). Elle ne renvoie qu'une seule ligne de résultats.

Syntaxe API

long AmQueryGet(long hApiQuery, char *strQueryCommand);

Syntaxe Basic interne

Function AmQueryGet(hApiQuery As Long, strQueryCommand As String) As Long

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiQuery** : Ce paramètre contient un descripteur valide sur l'objet requête auquel sont transmises les commandes AQL.
- **strQueryCommand** : Ce paramètre contient le corps de la requête AQL sous forme de chaîne.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmQueryNext()

Cette fonction renvoie le résultat suivant d'une requête préalablement exécutée au moyen de la fonction **AmQueryExec**.

Syntaxe API

long AmQueryNext(long hApiQuery);

Syntaxe Basic interne

Function AmQueryNext(hApiQuery As Long) As Long

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiQuery** : Ce paramètre contient un descripteur valide sur l'objet requête auquel sont transmises les commandes AQL.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmQuerySetAddMainField()

Cette fonction permet de passer une requête dans un mode où le champ principal de la table est automatiquement ajouté à la liste des champs à retourner. Une telle requête ne retournera jamais l'enregistrement d'identifiant nul.

Syntaxe API

long AmQuerySetAddMainField(long hApiQuery, long bAddMainField);

Syntaxe Basic interne

Function AmQuerySetAddMainField(hApiQuery As Long, bAddMainField As Long) As Long

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiQuery** : Ce paramètre contient un descripteur valide sur un objet requête.
- **bAddMainField** : Ce paramètre peut avoir deux valeurs :
  - True : Le champ principal de la table est ajouté,
  - False : Le champ principal de la table n'est pas ajouté.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmQuerySetFullMemo()

Par défaut, lors de l'exécution de la fonction **AmQueryExec**, la requête tronque les champs de type Memo à 254 caractères. Cette fonction passe la requête dans un mode où elle ramenera les valeurs des champs Memo dans leur integralité.

Syntaxe API

long AmQuerySetFullMemo(long hApiQuery, long bFullMemo);

Syntaxe Basic interne

Function AmQuerySetFullMemo(hApiQuery As Long, bFullMemo As Long) As Long

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiQuery** : Ce paramètre contient un descripteur valide sur un objet requête.
- **bFullMemo** : Ce paramètre peut avoir deux valeurs :
  - True : La requête renvoie l'intégralité du champ Memo,
  - False : La requête tronque les champs Memo à 254 caractères.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmQueryStartTable()

Cette fonction renvoie un descripteur sur la table sur laquelle porte une requête identifiée par son descripteur.

Syntaxe API

long AmQueryStartTable(long hApiQuery);

Syntaxe Basic interne

Function AmQueryStartTable(hApiQuery As Long) As Long

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiQuery** : Ce paramètre contient un descripteur valide sur un objet requête.

Sortie

En cas d'erreur, cette fonction retourne un descripteur non valide (de valeur nulle).

AmQueryStop()

Cette fonction interrompt l'exécution d'une requête identifiée par son descripteur. Cette requête doit avoir été préalablement lancée au moyen de la fonction **AmQueryExec**.

Syntaxe API

long AmQueryStop(long hApiQuery);

Syntaxe Basic interne

Function AmQueryStop(hApiQuery As Long) As Long

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiQuery** : Ce paramètre contient un descripteur valide sur un objet requête.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmReceiveAllPOLines()

Cette fonction effectue la réception de tous les éléments sur une ligne de commande.

 Note :

Attention : les lignes de réception sont créées par un agent au moment du "commit" de la transaction. Vous ne pouvez pas y accéder avant.

Syntaxe API

long AmReceiveAllPOLines(long hApiCnxBase, long lPOrdId, long lDelivId);

Syntaxe Basic interne

Function AmReceiveAllPOLines(lPOrdId As Long, lDelivId As Long) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lPOrdId** : Ce paramètre contient l'identifiant de la ligne de commande contenant les éléments à réceptionner.
- **lDelivId** : Ce paramètre contient l'identifiant de la fiche de réception qui recevra tous les éléments présents sur la ligne de commande.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmReceivePOLine()

Cette fonction effectue la réception d'une certaine quantité d'éléments sur une ligne de commande et renvoie le numéro d'identifiant de la ligne de réception.

Note :

Attention : les lignes de réception sont créées par un agent au moment du "commit" de la transaction. Vous ne pouvez pas y accéder avant.

Syntaxe API

long AmReceivePOLine(long hApiCnxBase, long lPOrdLineId, long lDelivId, double dQty);

Syntaxe Basic interne

Function AmReceivePOLine(lPOrdLineId As Long, lDelivId As Long, dQty As Double) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lPOrdLineId** : Ce paramètre contient l'identifiant de la ligne de commande contenant les éléments à réceptionner.
- **lDelivId** : Ce paramètre contient l'identifiant de la fiche de réception qui recevra une certaine quantité des éléments présents sur la ligne de commande.

- **dQty** : Ce paramètre contient la quantité d'éléments sur la ligne de commande à réceptionner dans la fiche de réception.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmRefreshAllCaches()

Cette fonction rafraîchit l'ensemble des caches utilisés sous AssetCenter.

Syntaxe API

long AmRefreshAllCaches(long hApiCnxBase);

Syntaxe Basic interne

Function AmRefreshAllCaches() As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	

	Utilisable
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmRefreshLabel()

L'API AmRefreshLabel rafraîchit la chaîne de l'étiquette d'un enregistrement donné (lMainId) dans une table donnée (strTableName).

Syntaxe API

long AmRefreshLabel(long hApiCnxBase, long lMainId, char *strTableName, char *pstrLabel, long lLabel);

Syntaxe Basic interne

Function AmRefreshLabel(lMainId As Long, strTableName As String) As String

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **lMainId** : ce paramètre contient l'identifiant qui sera rafraîchi.
- **strTableName** : ce paramètre précise le nom de la table associé à lMainId.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmRefreshProperty()

Réévalue la valeur d'une propriété identifiée par le paramètre **strVarName**. Si cette propriété utilise un script, celui-ci est à nouveau exécuté.

L'arbre de dépendance est remis à jour, le cas échéant.

Syntaxe Basic interne

Function AmRefreshProperty(strVarName As String) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	

	Utilisable
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strVarName** : Nom de la propriété (de l'assistant) que vous souhaitez réévaluer.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmRefreshTraceHist()

L'API AmRefreshTraceHist rafraîchit un historique complet de chaîne de liaisons de projet. Elle possède aussi un paramètre optionnel qui rafraîchit les entrées "individuelles" d'historique de chaîne de liaisons. Si ce paramètre n'est pas présent, l'historique complet de chaîne de liaisons sera rafraîchi.

Syntaxe API

long AmRefreshTraceHist(long hApiCnxBase, long lCabTraceOutId, long lTraceHistId);

Syntaxe Basic interne

Function AmRefreshTraceHist(lCabTraceOutId As Long, lTraceHistId As Long) As Long

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lCabTraceOutId** : ce paramètre contient l'identifiant de compte-rendu de chaîne de liaisons de câble.
- **lTraceHistId** : ce paramètre optionnel rafraîchit les entrées "individuelles" d'historique de chaîne de liaisons.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmReleaseHandle()

Cette fonction libère le descripteur et tous les sous-descripteurs d'un objet.

Syntaxe API

long AmReleaseHandle(long hApiObject);

Syntaxe Basic interne

Function AmReleaseHandle(hApiObject As Long) As Long

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiObject** : Ce paramètre contient un descripteur sur l'objet concerné.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmRemoveCable()

L'API AmRemoveCable enlève un câble (lCableId) de sa localisation actuelle. L'état du câble est changé en "Indisponible". Si le projet (lProjectId) et l'intervention (lWorkOrderId) prennent des valeurs, le câble est ajouté au projet et à l'intervention avec le commentaire donné (strComment). Ce commentaire décrit l'action qui sera accomplie sur le câble (i.e. "Enlever un câble de sa localisation actuelle").

Syntaxe API

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

Syntaxe Basic interne

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

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lCableId** : ce paramètre contient l'identifiant du câble à enlever.
- **lProjectId** : ce paramètre contient l'identifiant du projet.
- **lWorkOrderId** : ce paramètre contient l'identifiant de l'intervention.
- **strComment** : ce paramètre est le commentaire qui sera joint à l'intervention.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmRemoveDevice()

L'API AmRemoveDevice enlève un dispositif (lDeviceId) de sa localisation actuelle. L'état du câble est changé en "Indisponible". Si le projet (lProjectId) et l'intervention (lWorkOrderId) prennent des valeurs, le câble est ajouté au

projet et à l'intervention avec le commentaire donné (strComment). Ce commentaire décrit l'action qui sera accomplie sur le dispositif (i.e. "Enlever un dispositif de sa localisation actuelle").

Syntaxe API

long AmRemoveDevice(long hApiCnxBase, long lDeviceId, long lProjectId, long lWorkOrderId, char *strComment);

Syntaxe Basic interne

Function AmRemoveDevice(lDeviceId As Long, lProjectId As Long, lWorkOrderId As Long, strComment As String) As Long

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lDeviceId** : ce paramètre contient l'identifiant du dispositif à enlever.
- **lProjectId** : ce paramètre contient l'identifiant du projet.
- **lWorkOrderId** : ce paramètre contient l'identifiant de l'intervention.
- **strComment** : ce paramètre est le commentaire qui sera joint à l'intervention.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmRestoreRecord()

Cette fonction procède à la restauration d'un enregistrement archivé.

Syntaxe API

long AmRestoreRecord(long hApiRecord);

Syntaxe Basic interne

Function AmRestoreRecord(hApiRecord As Long) As Long

Champ d'application

Version : 4.3.0

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiRecord** : Ce paramètre contient le descripteur sur l'enregistrement concerné par l'opération.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

Remarques

 Note :

Le traitement des enregistrements liés dépend du type de lien. Dans le cas d'un lien de type OWN, les enregistrements liés sont traités à l'identique. Dans le cas d'un lien DEFINE ou NORMAL, les clés étrangères des enregistrements liés sont remises à 0 et les champs d'archivage sont renseignés avec l'identifiant de l'enregistrement archivé et sa chaîne de description.

 IMPORTANT :

Cette fonction n'est disponible que pour un enregistrement provenant d'une table d'archivage.

AmReturnAsset()

Cette fonction permet de retourner un bien.

Syntaxe API

long AmReturnAsset(long hApiCnxBase, long lAstId, long lReturnId, long bCanMerge);

Syntaxe Basic interne

Function AmReturnAsset(lAstId As Long, lReturnId As Long, bCanMerge As Long) As Long

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lAstId** : Ce paramètre contient l'identifiant du bien à retourner.
- **lReturnId** : Ce paramètre contient l'identifiant de la fiche de retour.
- **bCanMerge** : Ce paramètre permet de préciser si le retour peut être fusionné avec une ligne déjà existante dans la fiche de retour.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmReturnContract()

Cette fonction permet de retourner un contrat.

Syntaxe API

long AmReturnContract(long hApiCnxBase, long lCntrId, long lReturnId, long bCanMerge);

Syntaxe Basic interne

Function AmReturnContract(lCntrId As Long, lReturnId As Long, bCanMerge As Long) As Long

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lCntrId** : Ce paramètre contient l'identifiant du contrat à retourner.
- **lReturnId** : Ce paramètre contient l'identifiant de la fiche de retour.
- **bCanMerge** : Ce paramètre permet de préciser si le retour peut être fusionné avec une ligne déjà existante dans la fiche de retour.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction

AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmReturnPortfolioItem()

Cette fonction permet de retourner un élément de parc.

Syntaxe API

long AmReturnPortfolioItem(long hApiCnxBase, long lPfId, double dQty, long lFromRecptLineId, long lReturnId, long bCanMerge);

Syntaxe Basic interne

Function AmReturnPortfolioItem(lPfId As Long, dQty As Double, lFromRecptLineId As Long, lReturnId As Long, bCanMerge As Long) As Long

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lPfId** : Ce paramètre contient l'identifiant de l'élément de parc à retourner.

- **dQty** : Ce paramètre contient la quantité (dans l'unité du modèle) à retourner.
- **lFromRecptLineId** : Ce paramètre contient l'identifiant de la ligne de réception source.
- **lReturnId** : Ce paramètre contient l'identifiant de la fiche de retour.
- **bCanMerge** : Ce paramètre permet de préciser si le retour peut être fusionné avec une ligne déjà existante dans la fiche de retour.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmReturnTraining()

Cette fonction permet de retourner une formation.

Syntaxe API

long AmReturnTraining(long hApiCnxBase, long lTrainingId, long lReturnId, long bCanMerge);

Syntaxe Basic interne

Function AmReturnTraining(lTrainingId As Long, lReturnId As Long, bCanMerge As Long) As Long

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lTrainingId** : Ce paramètre contient l'identifiant de la formation à retourner.
- **lReturnId** : Ce paramètre contient l'identifiant de la fiche de retour.
- **bCanMerge** : Ce paramètre permet de préciser si le retour peut être fusionné avec une ligne déjà existante dans la fiche de retour.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmReturnWorkOrder()

Cette fonction permet de retourner une intervention.

Syntaxe API

long AmReturnWorkOrder(long hApiCnxBase, long lWOId, long lReturnId, long bCanMerge);

Syntaxe Basic interne

Function AmReturnWorkOrder(lWOId As Long, lReturnId As Long, bCanMerge As Long) As Long

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **lWOId** : Ce paramètre contient l'identifiant de l'intervention à retourner.
- **lReturnId** : Ce paramètre contient l'identifiant de la fiche de retour.
- **bCanMerge** : Ce paramètre permet de préciser si le retour peut être fusionné avec une ligne déjà existante dans la fiche de retour.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmRevCryptPassword()

Cette fonction encrypte un mot de passe de façon réversible. La fonction permettant de décrypter un mot de passe encrypté grâce à cette fonction n'est pas exposée.

Syntaxe API

long AmRevCryptPassword(long hApiCnxBase, char *return, long lreturn, char *strPassword);

Syntaxe Basic interne

Function AmRevCryptPassword(strPassword As String) As String

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strPassword** : Ce paramètre contient le mot de passe à encrypter.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmRgbColor()

Cette fonction donne la valeur RGB de la couleur correspondant au paramètre **strText**.

Syntaxe API

long AmRgbColor(char *strText);

Syntaxe Basic interne

Function AmRgbColor(strText As String) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **strText**: Ce paramètre contient le nom d'une couleur :
  - White
  - ltGray
  - Gray
  - Dkgray
  - Black
  - Red
  - Green
  - Blue
  - Yellow
  - Cyan
  - Magenta
  - Dkyellow
  - Dkgreen
  - Dkcyan
  - Dkblue
  - Dkmagenta
  - Dkred

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmRollback()

Cette fonction annule toutes les modifications effectuées avant la déclaration de début de transaction (effectuée via la fonction **AmStartTransaction**).

Syntaxe API

long AmRollback(long hApiCnxBase);

Syntaxe Basic interne

Function AmRollback() As Long

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmSetFieldDateOnlyValue()

Cette fonction modifie un champ d'un enregistrement. Aucune mise à jour de la base de données n'est effectuée par cette fonction. La modification sera

effectuée lors de la mise à jour ou l'insertion de l'enregistrement, ou encore lors du commit de la transaction.

Syntaxe API

long AmSetFieldDateOnlyValue(long hApiRecord, char *strFieldName, long dtptmValue);

Syntaxe Basic interne

Function AmSetFieldDateOnlyValue(hApiRecord As Long, strFieldName As String, dtptmValue As Date) As Long

Champ d'application

Version : 4.3.0

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiRecord** : Ce paramètre contient le descripteur sur l'enregistrement contenant le champ à modifier.
- **strFieldName** : Ce paramètre contient le nom SQL du champ à modifier.
- **dtptmValue** : Ce paramètre contient la nouvelle valeur du champ au format "Date" uniquement. A l'inverse de la fonction **AmSetFieldDateValue**, seule la partie Date est traitée, la partie heure est omise.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmSetFieldDateValue()

Cette fonction modifie un champ d'un enregistrement. Aucune mise à jour de la base de données n'est effectuée par cette fonction. La modification sera effectuée lors de la mise à jour ou l'insertion de l'enregistrement, ou encore lors du commit de la transaction.

Syntaxe API

long AmSetFieldDateValue(long hApiRecord, char *strFieldName, long tmValue);

Syntaxe Basic interne

Function AmSetFieldDateValue(hApiRecord As Long, strFieldName As String, tmValue As Date) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiRecord** : Ce paramètre contient le descripteur sur l'enregistrement contenant le champ à modifier.
- **strFieldName** : Ce paramètre contient le nom SQL du champ à modifier.
- **tmValue** : Ce paramètre contient la nouvelle valeur du champ au format "Date".

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmSetFieldDoubleValue()

Cette fonction modifie en mémoire un champ d'un enregistrement. Aucune mise à jour de la base de données n'est effectuée par cette fonction.

Syntaxe API

long AmSetFieldDoubleValue(long hApiRecord, char *strFieldName, double dValue);

Syntaxe Basic interne

Function AmSetFieldDoubleValue(hApiRecord As Long, strFieldName As String, dValue As Double) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiRecord** : Ce paramètre contient le descripteur sur l'enregistrement contenant le champ à modifier.
- **strFieldName** : Ce paramètre contient le nom SQL du champ à modifier.
- **dValue** : Ce paramètre contient la nouvelle valeur du champ au format "Double".

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmSetFieldLongValue()

Cette fonction modifie un champ d'un enregistrement. Aucune mise à jour de la base de données n'est effectuée par cette fonction. Pour modifier la valeur d'un champ date, heure ou date+heure, vous devez donner comme nouvelle valeur le nombre de secondes écoulées depuis le 01/01/1970 à 00:00:00.

Syntaxe API

long AmSetFieldLongValue(long hApiRecord, char *strFieldName, long lValue);

Syntaxe Basic interne

Function AmSetFieldLongValue(hApiRecord As Long, strFieldName As String, lValue As Long) As Long

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiRecord** : Ce paramètre contient le descripteur sur l'enregistrement contenant le champ à modifier.
- **strFieldName** : Ce paramètre contient le nom SQL du champ à modifier. Vous pouvez également donner le nom SQL d'une caractéristique, d'un champ de type "Commentaire" ou encore d'un champ de script.
- **lValue** : Ce paramètre contient la nouvelle valeur du champ.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmSetFieldStrValue()

Cette fonction modifie un champ d'un enregistrement. Aucune mise à jour de la base de données n'est effectuée par cette fonction.

Syntaxe API

long AmSetFieldStrValue(long hApiRecord, char *strFieldName, char *strValue);

Syntaxe Basic interne

Function AmSetFieldStrValue(hApiRecord As Long, strFieldName As String, strValue As String) As Long

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiRecord** : Ce paramètre contient le descripteur sur l'enregistrement contenant le champ à modifier.
- **strFieldName** : Ce paramètre contient le nom SQL du champ à modifier.
- **strValue** : Ce paramètre contient la nouvelle valeur du champ au format "String" (chaîne de caractères).

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmSetLinkFeatureValue()

Cette fonction fixe la valeur d'une caractéristique de type lien pour un enregistrement donné.

Syntaxe API

long AmSetLinkFeatureValue(long hApiRecord, char *strFeatSqlName, char *strDstSelfValue, long lDstId);

Syntaxe Basic interne

Function AmSetLinkFeatureValue(hApiRecord As Long, strFeatSqlName As String, strDstSelfValue As String, lDstId As Long) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiRecord** : Ce paramètre contient l'identifiant de l'enregistrement auquel est associée la caractéristique de type lien.
- **strFeatSqlName** : Ce paramètre contient le nom SQL de la caractéristique de type lien dont on souhaite fixer la valeur. Ce nom SQL est toujours préfixé par "fv_".

- **strDstSelfValue** : Ce paramètre contient la valeur de la caractéristique telle qu'elle sera affichée pour l'enregistrement. Il s'agit de la valeur "Self" de l'enregistrement d'identifiant **lDstId**. Si vous renseignez ce paramètre avec une valeur non valide ou non existante, l'intégrité de la base de données risque d'être corrompue.
- **lDstId** : Ce paramètre contient l'identifiant de l'enregistrement sur lequel pointe la caractéristique de type lien.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmSetProperty()

Cette fonction fixe la valeur d'une propriété identifiée par son nom. Elle met également à jour l'arbre de dépendances de cette propriété.

Syntaxe Basic interne

Function AmSetProperty(strVarName As String, vValue As Variant) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **strVarName** : Ce paramètre contient le nom de la propriété dont on souhaite fixer la valeur.
- **vValue** : Ce paramètre contient la nouvelle valeur pour la propriété.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmShowCableCrossConnect()

Cette fonction affiche l'écran des interconnexions d'un câble.

Syntaxe Basic interne

Function AmShowCableCrossConnect(lCableId As Long) As Long

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **lCableId** : Ce paramètre contient l'identifiant du câble concerné par l'opération.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmShowDeviceCrossConnect()

Cette fonction affiche l'écran des interconnexions d'un dispositif de câblage.

Syntaxe Basic interne

Function AmShowDeviceCrossConnect(lDeviceId As Long) As Long

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **lDeviceId** : Ce paramètre contient l'identifiant du dispositif de câblage concerné par l'opération.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmSqlTextConst()

Cette fonction transforme une chaîne de caractères en vue de son utilisation dans une requête. Les opérations suivantes sont effectuées sur la chaîne :

- Tous les simples guillemets (') sont doublés,
- Des guillemets simples sont ajoutés en début et en fin de châine.

Syntaxe API

long AmSqlTextConst(char *return, long lreturn, char *str);

Syntaxe Basic interne

Function AmSqlTextConst(str As String) As String

Champ d'application

Version : 4.00

	Utilisable
AssetCenter API	✓

	Utilisable
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **str** : Ce paramètre contient la chaîne de caractères à traîter.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim strReq as String
strReq="SELECT lEmplDeptId FROM amEmplDept WHERE Name=" & amSqlTextCon
st(strName)
```

Cette requête est valide, même si la variable strName contient des simples guillemets.

AmStandIn()

Cette fonction renvoie l'identifiant de la personne remplaçant la personne d'identifiant **lEmployeeId** à la date **tmDate**.

Syntaxe API

long AmStandIn(long hApiCnxBase, long lEmployeeId, long tmDate);

Syntaxe Basic interne

Function AmStandIn(lEmployeeId As Long, tmDate As Date) As Long

Champ d'application

Version : 4.3.0

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **lEmployeeId** : Ce paramètre contient l'identifiant de la personne dont on veut connaître le remplaçant.
- **tmDate** : Ce paramètre contient la date à laquelle la fonction opère la recherche.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Remarques

Note :

Si à la date **tmDate** précisée, la personne d'identifiant **lEmployeeId** est présente, la fonction renvoie son identifiant.

Si la personne est absente et qu'aucun remplaçant n'est désigné, la fonction renvoie 0.

Exemple

```
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()

Cette fonction renvoie l'identifiant du groupe de personnes remplaçant la personne d'identifiant **lEmployeeId** à la date **tmDate**.

Syntaxe API

long AmStandInGroup(long hApiCnxBase, long lEmployeeId, long tmDate);

Syntaxe Basic interne

Function AmStandInGroup(lEmployeeId As Long, tmDate As Date) As Long

Champ d'application

Version : 4.3.0

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **lEmployeeId** : Ce paramètre contient l'identifiant de la personne dont on veut connaître le groupe remplaçant.
- **tmDate** : Ce paramètre contient la date à laquelle la fonction opère la recherche.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Remarques

Note :

Si à la date **tmDate** précisée, la personne d'identifiant **lEmployeeId** est présente, la fonction renvoie 0.

Si la personne est absente et qu'aucun groupe remplaçant n'est désigné, la fonction renvoie également 0.

AmStartTransaction()

Cette fonction démarre une nouvelle transaction avec la base de données associée à la connexion. La prochaine commande de "Commit" ou de "Rollback" validera ou annulera toutes les modifications apportées à la base de données.

Syntaxe API

long AmStartTransaction(long hApiCnxBase);

Syntaxe Basic interne

Function AmStartTransaction() As Long

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmStartup()

Cette fonction doit être appelée avant tout autre fonction. Elle initialise l'appel aux librairies AssetCenter.

Syntaxe API

void AmStartup();

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	

AmTableDesc()

Cette fonction génère une chaîne de format "<Description de la table> (<Nom SQL de la table>)" à partir du nom SQL de la table.

Syntaxe API

long AmTableDesc(long hApiCnxBase, char *return, long lreturn, char *strSqlName);

Syntaxe Basic interne

Function AmTableDesc(strSqlName As String) As String

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strSqlName** : Nom SQL de la table pour laquelle on veut générer une chaîne de description. Si ce paramètre contient un nom SQL non valide, la fonction renvoie un point d'interrogation ("?").

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

L'exemple suivant génère une chaîne de description pour la table des biens (Nom SQL : amAsset) :

```
AmTableDesc("amAsset")
```

Le résultat est le suivant :

```
Biens (amAsset)
```

AmTaxRate()

Cette fonction calcule un taux de taxe en fonction d'un type de taxe, d'une juridiction fiscale et d'une date.

Syntaxe API

double AmTaxRate(long hApiCnxBase, char *strTaxRateName, long lTaxLocId, long tmDate, double dValue);

Syntaxe Basic interne

Function AmTaxRate(strTaxRateName As String, lTaxLocId As Long, tmDate As Date, dValue As Double) As Double

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strTaxRateName** : Ce paramètre contient le nom SQL du type de taxe utilisé pour calculer le taux de taxe.
- **lTaxLocId** : Ce paramètre contient le numéro d'identifiant de la juridiction fiscale concernée par le type de taxe.
- **tmDate** : Ce paramètre contient la date à laquelle vous souhaitez évaluer le taux de taxe.

- **dValue** : Paramètre obsolète, présent pour des raisons de compatibilité. Renseignez le de manière arbitraire.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

AmUpdateDetail()

Cette fonction est utilisée dans les assistants d'aide à la saisie. La définition du contexte (la table pour laquelle un enregistrement est mis à jour ou renseigné à l'aide de l'assistant) n'est donc pas nécessaire. La fonction met à jour un ou renseigne des champs ou des liens du contexte en fonction d'une valeur. Cette fonction est interdite dans les assistants non modaux.

Syntaxe Basic interne

Function AmUpdateDetail(strFieldName As String, varValue As Variant) As Long

Champ d'application

Version : 3.00

Utilisable
AssetCenter API
Script de configuration d'un champ ou d'un lien
Action de type "Script"
Workflow de déploiement

	Utilisable
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **strFieldName** : Ce paramètre contient le nom SQL du champ à mettre à jour.
- **varValue** : Ce paramètre contient la nouvelle valeur du champ.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmUpdateLossLines()

Cette fonction permet de mettre à jour toutes les valeurs de pertes pour les contrats dont la règle de valeur de perte est celle d'identifiant **lLossValId**.

Syntaxe Basic interne

Function AmUpdateLossLines(lLossValId As Long) As Long

Champ d'application

Version : 4.3.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **lLossValId** : Ce paramètre contient l'identifiant de la règle de valeur de perte.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmUpdateRecord()

Cette fonction permet de mettre à jour un enregistrement.

Syntaxe API

long AmUpdateRecord(long hApiRecord);

Syntaxe Basic interne

Function AmUpdateRecord(hApiRecord As Long) As Long

Champ d'application

Version : 2.52

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	✓
Workflow de déploiement	

	Utilisable
Script d'un assistant	
Script FINISH.DO d'un assistant	✓

Entrée

- **hApiRecord** : Ce paramètre contient un descripteur sur l'enregistrement que vous souhaitez mettre à jour.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmUpdateUser()

Cette fonction met à jour les informations (domaine, nom d'utilisateur NT, description) d'une personne dans la base de données.

Syntaxe API

long AmUpdateUser(long hApiCnxBase, long lId, char *strNTUserName, char *strNTDomain, char *strNTUserDesc);

Syntaxe Basic interne

Function AmUpdateUser(lId As Long, strNTUserName As String, strNTDomain As String, strNTUserDesc As String) As Long

Champ d'application

Version : 4.3.0.0

	Utilisable
AssetCenter API	✓

	Utilisable
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **lId** : Ce paramètre contient l'identifiant de l'utilisateur concerné dans la base de données.
- **strNTUserName** : Ce paramètre contient le nom d'utilisateur NT de la personne.
- **strNTDomain** : Ce paramètre contient le nom de domaine NT de la personne.
- **strNTUserDesc** : Ce paramètre contient la description associée à l'utilisateur NT.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmValueOf()

Utilisée au sein d'un assistant, cette fonction renvoie la valeur de la propriété identifiée par le paramètre **strVarName**.

Syntaxe Basic interne

Function AmValueOf(strVarName As String) As Variant

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strVarName** : Ce paramètre contient le nom de la propriété dont on veut connaître la valeur.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

L'exemple suivant renvoie la valeur de la propriété "Page1.Label" :

```
AmValueOf("Page1.Label")
```

Utilisez cette fonction avec prudence car elle brise la chaîne de dépendances de la propriété qu'elle traite.

AmWizChain()

Cette fonction exécute un assistant B au sein d'un assistant A. En fin d'exécution de l'assistant B, la fonction rend la main à l'assistant A.

Syntaxe Basic interne

Function AmWizChain(strWizSqlName As String) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strWizSqlName** : Nom SQL de l'assistant à exécuter.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

AmWorkTimeSpanBetween()

Cette fonction renvoie la durée ouvrée entre deux dates. Cette durée est exprimée en secondes et respecte les informations d'un calendrier des périodes ouvrées.

Syntaxe API

long AmWorkTimeSpanBetween(long hApiCnxBase, char *strCalendarSqlName, long tmEnd, long tmStart);

Syntaxe Basic interne

Function AmWorkTimeSpanBetween(strCalendarSqlName As String, tmEnd As Date, tmStart As Date) As Date

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strCalendarSqlName** : Ce paramètre contient le nom SQL du calendrier des périodes ouvrées utilisé comme base pour le calcul de la durée ouvrée écoulée entre les deux dates. Si ce paramètre est omis, la durée calculée ne tient compte d'aucune période ouvrée.

- **tmEnd** : Ce paramètre contient la date de fin de la période sur laquelle on effectue le calcul de la durée ouvrée.
- **tmStart** : Ce paramètre contient la date de début de la période sur laquelle on effectue le calcul de la durée ouvrée.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

L'exemple suivant calcule la durée ouvrée entre le 01/09/1998 à 8h00 et le 24/09/1998 à 19h00. Le calendrier utilisé, de nom SQL "Calendar_Paris" définit les périodes ouvrées suivantes :

- Du lundi au jeudi de 8h00 à 12h00, puis de 14h00 à 18h00.
- Le vendredi de 8h00 à 12h00, puis de 14h00 à 17h00.

```
AmWorkTimeSpanBetween("Calendar_Paris", "1998/09/24 19:00:00", "1998/0
9/01 08:00:00")
```

Cet exemple renvoie la valeur 507600 qui représente le nombre de secondes ouvrées écoulées entre les deux dates.

AppendOperand()

Concatène une chaîne en fonction des paramètres passés à la fonction. Le résultat a la forme suivante :

```
strExpr strOperator strOperand
```

Syntaxe Basic interne

Function AppendOperand(strExpr As String, strOperator As String, strOperand As String) As String

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strExpr** : Expression à concaténer.
- **strOperator** : Opérateur à concaténer.
- **strOperand** : Opérande à concaténer.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Remarques

Note :

Si l'un des paramètres **strExpr** ou **strOperand** est omis, **strOperator** n'est pas utilisé dans la concaténation.

ApplyNewVals()

Affecte des valeurs identiques pour les cellules identifiées d'un contrôle "ListBox".

Syntaxe Basic interne

Function ApplyNewVals(strValues As String, strNewVals As String, strRows As String, strRowFormat As String) As String

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strValues** : Chaîne source contenant les valeurs d'un contrôle "ListBox" à traiter.
- **strNewVals** : Nouvelle valeur à affecter aux cellules concernées.

- **strRows** : Identifiants des lignes à traiter. Les identifiants sont séparés par une virgule.
- **strRowFormat** : Instructions de formatage de la sous-liste. Les instructions sont séparées par le caractère "|". Chaque instruction représente le numéro de la colonne qui contiendra **strNewVals**.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Asc()

Renvoie le code ASCII du premier caractère d'une chaîne.

Syntaxe Basic interne

Function Asc(strAsc As String) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **strAsc** : Chaîne de caractères sur laquelle opère la fonction.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim iCount as Integer
Dim strString as String
  For iCount=Asc("A") To Asc("Z")
    strString = strString & Str(iCount)
  Next iCount
  RetVal=strString
```

Atn()

Renvoie l'arc tangente d'un nombre, exprimé en radians

Syntaxe Basic interne

Function Atn(dValue As Double) As Double

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **dValue** : Nombre dont vous souhaitez connaître l'arc tangente.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim dPi as Double
Dim strString as String
  dPi=4*Atn(1)
  strString = Str(dPi)
  RetVal=strString
```

BasicToLocalDate()

Cette fonction convertit une date au format Basic en une date au format chaîne (telle qu'elle est affichée dans le "control panel" de Windows).

Syntaxe Basic interne

Function BasicToLocalDate(strDateBasic As String) As String

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strDateBasic** : Date au format Basic à convertir.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

BasicToLocalTime()

Cette fonction convertit une heure au format Basic en une heure au format chaîne (telle qu'elle est affichée dans le "control panel" de Windows).

Syntaxe Basic interne

Function BasicToLocalTime(strTimeBasic As String) As String

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strTimeBasic** : Heure au format Basic à convertir.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

BasicToLocalTimeStamp()

Cette fonction convertit un ensemble Date+Heure au format Basic en un ensemble Date+Heure au format chaîne (telle qu'elle est affichée dans le "control panel" de Windows).

Syntaxe Basic interne

Function BasicToLocalTimeStamp(strTSBasic As String) As String

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strTSBasic** : Date+Heure au format Basic à convertir.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Beep()

Emet un son (un beep) sur la machine.

Syntaxe Basic interne

Function Beep()

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

CDbl()

Convertit une expression en un double ("Double").

Syntaxe Basic interne

Function CDbl(dValue As Double) As Double

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **dValue** : Expression à convertir.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim dNumber As Double
Dim iInteger as Integer
  iInteger = 25
  dNumber=CDbl(iInteger)
  RetVal=dNumber
```

ChDir()

Change le répertoire courant.

Syntaxe Basic interne

Function ChDir(strDirectory As String)

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strDirectory** : Nouveau répertoire courant.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

ChDrive()

Change le lecteur courant.

Syntaxe Basic interne

Function ChDrive(strDrive As String)

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strDrive** : Nouveau lecteur.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Chr()

Renvoie la chaîne correspondant au code ASCII passé par le paramètre **iChr**.

Syntaxe Basic interne

Function Chr(lChr As Long) As String

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **lChr** : Code ASCII du caractère.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
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()

Convertit une expression en un entier ("Integer").

Syntaxe Basic interne

Function CInt(iValue As Long) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **iValue** : Expression à convertir.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim iNumber As Integer
Dim dDouble as Double
  dDouble = 25.24589
  iNumber=CInt(dDouble)
  RetVal=iNumber
```

CLng()

Convertit une expression en un long ("Long").

Syntaxe Basic interne

Function CLng(lValue As Long) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **lValue** : Expression à convertir.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim lNumber As Long
Dim iInteger as Integer
  iInteger = 25
  lNumber=CLng(iInteger)
  RetVal=lNumber
```

Cos()

Renvoie le cosinus d'un nombre, exprimé en radians.

Syntaxe Basic interne

Function Cos(dValue As Double) As Double

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **dValue** : Nombre dont vous souhaitez connaître le cosinus.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim dCalc as Double
  dCalc=Cos(150)
  RetVal=dCalc
```

CountOccurences()

Compte le nombre d'occurrences d'une chaîne de caractères à l'intérieur d'une autre chaîne.

Syntaxe Basic interne

Function CountOccurences(strSearched As String, strPattern As String, strEscChar As String) As Long

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strSearched** : Chaîne de caractères à l'intérieur de laquelle s'effectue la recherche.
- **strPattern** : Chaîne de caractères à rechercher à l'intérieur de **strSearched**.
- **strEscChar** : Caractère d'échappement. Si la fonction rencontre ce caractère à l'intérieur de la chaîne **strSearched**, la recherche s'arrête.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim MyStr
  MyStr=CountOccurences("toi|moi|toi,moi|toi", "toi", ",") :'Renvoie l
a valeur "2"
  MyStr=CountOccurences("toi|moi|toi,moi|toi", "toi", "|") :'Renvoie l
a valeur "1"
```

CountValues()

Compte le nombre d'éléments dans une chaîne de caractères en tenant compte d'un séparateur et d'un caractère d'échappement.

Syntaxe Basic interne

Function CountValues(strSearched As String, strSeparator As String, strEscChar As String) As Long

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strSearched** : Chaîne de caractères à traiter.
- **strSeparator** : Séparateur utilisé pour délimiter les éléments.
- **strEscChar** : Caractère d'échappement. Si ce caractère préfixe un séparateur, ce dernier sera ignoré.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim MyStr
  MyStr=CountValues("toi|moi|toi\|moi|toi", "|", "\") :'Renvoie la val
eur 4
  MyStr=CountValues("toi|moi|toi\|moi|toi", "|", "") :'Renvoie la vale
ur 5
```

CSng()

Convertit une expression en un nombre à virgule flottante ("Float").

Syntaxe Basic interne

Function CSng(fValue As Single) As Single

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **fValue** : Expression à convertir.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim dNumber As Double
Dim iInteger as Integer
  iInteger = 25
  dNumber=CSng(iInteger)
  RetVal=dNumber
```

CStr()

Convertit une expression en une chaîne de caractères ("String").

Syntaxe Basic interne

Function CStr(strValue As String) As String

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strValue** : Expression à convertir.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim dNumber As Double
Dim strMessage as String
  dNumber = 2,452873
  strMessage=CStr(dNumber)
  RetVal=strMessage
```

CurDir()

Renvoie le chemin courant.

Syntaxe Basic interne

Function CurDir() As String

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

CVar()

Convertit une expression en un variant ("Variant").

Syntaxe Basic interne

Function CVar(vValue As Variant) As Variant

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **vValue** : Expression à convertir.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

DaContext()

Cette fonction revoie la valeur d'un contexte identifié par son nom.

Syntaxe Basic interne

Function DaContext(strField As String) As Variant

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strField** : ce paramètre contient le nom du contexte dont vous souhaitez retrouver la valeur. La liste des contextes disponibles est la suivante :
  - Instance.Id
  - Workflow.Id
  - Status.Id
  - Computer.Id
  - Computer.Name
  - Computer.OperatingSystem
  - Computer.OSServiceLevel
  - Computer.OSBuildNumber
  - Computer.TcpIpHostName
  - Computer.TcpIpAddress
  - Computer.IpxSpxAddress
  - Computer.PhysicalAddress
  - Computer.Workgroup
  - Computer.ComputerType
  - Agent.Id.0
  - Agent.Name.0
  - Agent.seVersion.0
  - Agent.seType.0
  - Agent.sPortNumber.0

- Agent.TcpIpHostName.0
- Agent.NetBIOSId.0
- Agent.IpxNode.0
- Agent.IpxSocket.0
- Agent.IpxNetwork.0
- Agent.Password.0

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

Cet exemple vérifie que les variables d'environnement peuvent être récupérées.

```
' Check that environment variables can be retrieved - this also retrie
ves all
' variables in one shot, improving performances
lRc = DaFirstEnv()

' Get useful environment variables and store them in context
if lRc <> 0 then
 ' Get environment variables
 strProcArch      = DaGetEnv( "PROCESSOR_ARCHITECTURE" )
 strProcId        = DaGetEnv( "PROCESSOR_IDENTIFIER" )
 strProcLevel     = DaGetEnv( "PROCESSOR_LEVEL" )
 strProcRevision  = DaGetEnv( "PROCESSOR_REVISION" )
 strProcCount     = DaGetEnv( "NUMBER_OF_PROCESSORS" )
 strOs            = DaGetEnv( "OS" )
 strComputer      = DaGetEnv( "COMPUTERNAME" )

 ' Check that computer name is the right one
 if strComputer <> DaContext( "Computer.Name" ) then
   print "Warning: computer name does not look like the right one !"
 end if

 ' Store them in context
 DaSetContext "EnvInfo.ProcArch", strProcArch
```

```
DaSetContext "EnvInfo.ProcId", strProcId
DaSetContext "EnvInfo.ProcLevel", strProcLevel
DaSetContext "EnvInfo.ProcRevision", strProcRevision
DaSetContext "EnvInfo.ProcCount", strProcCount
DaSetContext "EnvInfo.Os", strOs
DaSetContext "EnvInfo.Computer", strComputer

' Dump
print DaDumpContext()
end if
```

DaCopy()

Cette fonction copie un fichier, un ensemble de fichiers ou un répertoire, localement. Cette fonction possède deux modes opératoires :

- copie locale au serveur de déploiement (la source et la destination de la copie se trouvent toutes deux sur le serveur de déploiement)
- copie locale à la cible de déploiement (la source et la destination de la copie se trouvent toutes deux sur la cible de déploiement)

Le mode opératoire est déterminé par la valeur de l'option **file.on_server**, fixée au moyen de la fonction **DaSetOption**() :

- si cette option a pour valeur 1, la copie porte sur le serveur de déploiement
- si cette option a pour valeur 0, la copie porte sur la cible de déploiement.

Note :

Pour plus d'informations, nous vous invitons à consulter le descriptif de la fonction **DaSetOption**().

Note :

Cette fonction effectue les mêmes opérations que l'activité **Copie de fichiers**.

Syntaxe Basic interne

Function DaCopy(strSource As String, strDest As String, strNameFilter As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strSource** : ce paramètre contient le chemin source des éléments à copier. Suivant le mode opératoire de la fonction il s'agit soit d'un chemin relatif au chemin du dépôt de fichier (dans le cas d'une copie locale au serveur de déploiement), soit d'un chemin absolu (dans le cas d'une copie locale à la cible de déploiement).
- **strDest** : ce paramètre contient le chemin destination des éléments à copier. Suivant le mode opératoire de la fonction il s'agit soit d'un chemin relatif au chemin du dépôt de fichier (dans le cas d'une copie locale au serveur de déploiement), soit d'un chemin absolu (dans le cas d'une copie locale à la cible de déploiement).
- **strNameFilter** : caractères joker de type DOS (* et ?) utilisés pour filtrer les noms de fichiers.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
' copy user spreadsheet documents to a backup folder
dim l as long
l = DaSetOption( "file.recursive", 1 )
l = DaCopy( "c:\%USERPROFILE%", "c:\backup", "\*.xls" )
```

DaDbDeleteList()

Cette fonction efface un ou plusieurs enregistrements dans la base de données AssetCenter.

Syntaxe Basic interne

Function DaDbDeleteList(strTable As String, strList As String, strLineSep As String, strIdSep As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strTable** : ce paramètre contient le nom SQL de la table dans laquelle les enregistrements sont insérés ou mis à jour.
- **strList** : ce paramètre contient la chaîne de caractères décrivant les enregistrement à mettre à jour ou à insérer.

Note :

Ce paramètre peut être renseigné par une liste obtenue grâce à la fonction **DbGetList** ou si le paramètre **strIdStep** est vide par une liste d'identifiants d'enregistrements.

- **strLineSep** : ce paramètre contient le caractère utilisé comme séparateur de lignes dans la chaîne de caractères contenue dans le paramètre **strList**.
- **strIdSep** : ce paramètre contient l'identifiant de l'enregistrement à mettre à jour ou à insérer. Si ce paramètre possède une valeur nulle, la fonction considère qu'il s'agit d'une insertion. Si la valeur est non nulle, la fonction considère qu'il s'agit d'une mise à jour de l'enregistrement possédant cette valeur comme identifiant.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

DaDbGetList()

Cette fonction renvoie, sous la forme d'une liste, le résultat de l'exécution d'une requête AQL.

Note :

Pour plus d'informations sur le langage de requêtes AQL, veuillez consulter le chapitre **AQL** du manuel intitulé **Utilisation avancée**.

Syntaxe Basic interne

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

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strQuery** : ce paramètre contient la requête AQL que vous souhaitez exécuter.
- **strColSep** : ce paramètre contient le caractère utilisé comme séparateur de colonnes dans le résultat donné par la fonction.

  Une colonne équivaut à un champ d'un enregistrement renvoyé par la requête.
- **strLineSep** : ce paramètre contient le caractère utilisé comme séparateur de lignes dans le résultat donné par la fonction.

  Une ligne équivaut à un enregistrement renvoyé par la requête.
- **strIdSep** : ce paramètre contient le caractère utilisé comme séparateur d'identifiant d'enregistrement dans le résultat donné par la fonction.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.

- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim strList As String
Dim lWorkgroup, lErr As Long

' Get list of computer ids in workgroup 'ToChange'
strList = DaDbGetList( "SELECT Workgroup FROM amComputer WHERE Workgro
up='ToChange'", "|", "/", "@" )

' Replace all 'ToChange' workgroups by 'Changed'. InStr will return 0
if not found
lWorkgroup = InStr( 0, strList, "ToChange" )
Do While lWorkgroup > 0
  strList = Left$( strList, lWorkgroup - 1 ) + "Changed" + Mid$( strLi
st, lWorkgroup + Len("ToChange"), Len(strList) )

  lWorkgroup = InStr( 0, strList, "ToChange" )
Loop

' Update computers
lErr = DaDbSetList( "amComputer", "Workgroup", strList, "|", "/", "@"
)
```

DaDbSetList()

Cette fonction crée ou met à jour des enregistrements dans une table de la base de données AssetCenter.

Note :

Cette fonction est la fonction inverse de la fonction **DaDbGetList**.

Syntaxe Basic interne

Function DaDbSetList(strTable As String, strFields As String, strList As String, strColSep As String, strLineSep As String, strIdSep As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strTable** : nom SQL de la table sur laquelle porte l'opération.
- **strFields** : noms SQL des champs à modifier (dans le cas de la modification d'un enregistrement existant) ou à peupler (dans le case de la création d'un nouvel enregistrement. Le caractère défini dans le paramètre **strColSep** est utilisé comme séparateur pour les noms SQL des champs.
- **strList** : liste des enregistrements concernés par l'opération. Chaque enregistrement est séparé par le caractère **strLineSep**. Dans le cas d'une modification (mise à jour d'un enregistrement) il convient de préciser le numéro d'identifiant de l'enregistrement à modifier en ajoutant le caractère défini dans le paramètre **strIdSep** suivi du numéro d'identifiant.
- **strColSep** : caractère utilisé comme séparateur de colonnes (une colonne équivaut à un champ).
- **strLineSep** : caractère utilisé comme séparateur pour les enregistrements (séparateur de lignes).
- **strIdSep** : caractère utilisé pour préciser le numéro d'identifiant de l'enregistrement à modifier.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.

- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
' Update name and workgroup of the computer '42' and insert a new comp
uter named 'NewComputer' in computers table
' We could also use 'NewName|NewWorkgroup@0' to insert the new compute
r.

lErr = DaDbSetList( "amComputer", "Name|Workgroup", "UpdatedName|Updat
edWorkgroup@42/NewName|NewWorkgroup","|", "/", "@" )
```

DaDelete()

Cette fonction efface un fichier, un ensemble de fichiers, ou un répertoire. Cette fonction possède deux modes opératoires :

- effacement sur le serveur de déploiement (dans le dépôt de fichiers)
- effacement sur la cible de déploiement

Le mode opératoire est déterminé par la valeur de l'option **file.on_server**, fixée au moyen de la fonction **DaSetOption**() :

- si cette option a pour valeur 1, l'effacement porte sur le serveur de déploiement
- si cette option a pour valeur 0, l'effacement porte sur la cible de déploiement.

Note :

Pour plus d'informations, nous vous invitons à consulter le descriptif de la fonction **DaSetOption**().

Note :

Cette fonction effectue les mêmes opérations que l'activité **Effacement de fichiers**.

Syntaxe Basic interne

Function DaDelete(strToDelete As String, strNameFilter As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strToDelete** : ce paramètre contient le chemin des éléments à effacer. Si les éléments sont à effacer sur le serveur de déploiement, il s'agit d'un chemin relatif par rapport au chemin absolu du dépôt de fichiers. Si les éléments sont à effacer sur la cible de déploiement, il s'agit d'un chemin absolu.
- **strNameFilter** : caractères joker de type DOS (* et ?) utilisés pour filtrer les noms de fichiers.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
'Remove temporary files from server depot
dim l as long
l = DaSetOption( "file.recursive", 1 )
l = DaSetOption( "file.on_server", 1 )
l = DaDelete( "", "*.tmp" )
```

DaDownload()

Cette fonction copie un fichier, un répertoire ou un ensemble de fichiers sur le dépôt du serveur de déploiement. Les fichiers source doivent être stockés sur la cible de déploiement.

Vous pouvez paramétrer le comportement de cette fonction en fixant certaines options au moyen de la fonction **DaSetOption**(). Vous trouverez une liste complète de ces options dans le descriptif de la fonction **DaSetOption**().

Note :

Cette fonction effectue les mêmes opérations que l'activité **Réception de fichiers**.

Syntaxe Basic interne

Function DaDownload(strSrcPath As String, strDstPath As String, strSrcNameFilter As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	

Entrée

- **strSrcPath** : ce paramètre contient le chemin des fichiers à copier dans le dépôt du serveur de déploiement. Il s'agit d'un chemin absolu sur la cible de déploiement. Par exemple :

```
c:\program files\antivirus\update\file.exe
```

- **strDstPath** : ce paramètre contient le chemin du fichier copié sur le serveur de déploiement. Ce chemin est relatif par rapport au chemin absolu du serveur de déploiement. Par exemple, si le chemin absolu du dépôt est :

```
c:\files\depot
```

et que vous souhaitez copier un fichier à cet endroit :

```
c:\files\depot\software\antivirus\update\file.exe
```

alors ce paramètre aura la valeur suivante :

```
software\antivirus\update
```

- **strSrcNameFilter** : caractères joker de type DOS (***** et **?**) utilisés pour filtrer les noms de fichiers.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
'get target specific path within file depot
dim path as string
path = "logs/" & DaContext( "Computer.Name" )
```

```
' retrieve a log file from target
dim l as long
l = DaDownload( "c:\program files\application\setup", path, "setup.log
" )
```

DaDumpContext()

Cette fonction renvoie une chaîne contenant tous les contextes définis et est utilisée principalement en mode débogage.

Syntaxe Basic interne

Function DaDumpContext() As String

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
' Check that environment variables can be retrieved - this also retrie
ve all
' variables in one shot, improving performances
lRc = DaFirstEnv()

' Get usefull environment variables and store them in context
if lRc <> 0 then
 ' Get environment variables
 strProcArch      = DaGetEnv( "PROCESSOR_ARCHITECTURE" )
 strProcId        = DaGetEnv( "PROCESSOR_IDENTIFIER" )
 strProcLevel     = DaGetEnv( "PROCESSOR_LEVEL" )
 strProcRevision  = DaGetEnv( "PROCESSOR_REVISION" )
 strProcCount     = DaGetEnv( "NUMBER_OF_PROCESSORS" )
 strOs            = DaGetEnv( "OS" )
 strComputer      = DaGetEnv( "COMPUTERNAME" )

 ' Check that computer name is the right one
 if strComputer <> DaContext( "Computer.Name" ) then
   print "Warning: computer name does not look like the right one !"
 end if

 ' Store them in context
 DaSetContext "EnvInfo.ProcArch", strProcArch
 DaSetContext "EnvInfo.ProcId", strProcId
 DaSetContext "EnvInfo.ProcLevel", strProcLevel
 DaSetContext "EnvInfo.ProcRevision", strProcRevision
 DaSetContext "EnvInfo.ProcCount", strProcCount
 DaSetContext "EnvInfo.Os", strOs
 DaSetContext "EnvInfo.Computer", strComputer

 ' Dump
 print DaDumpContext()
end if
```

DaExec()

Cette fonction exécute un programme stocké sur la cible de déploiement.

Vous pouvez paramétrer le comportement de cette fonction en fixant certaines options au moyen de la fonction **DaSetOption**(). Vous trouverez une liste complète de ces options dans le descriptif de la fonction **DaSetOption**().


Note :

Cette fonction effectue les mêmes opérations que l'activité **Exécution**.

Syntaxe Basic interne

Function DaExec(strCmd As String, strPath As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strCmd** : ce paramètre contient le nom du programme à exécuter. Par exemple :

```
notepad.exe
```

- **strPath** : ce paramètre contient le chemin complet de l'exécutable. Dans le cas de l'exemple précédent :

```
c:\windows\system32
```

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
' Log ipconfig information
lErr = DaSetOption( "exec.log_output", 1 )
lErr = DaExec( "ipconfig /all", "" )
```

DaExecAction()

Cette fonction exécute une action, identifiée par son nom, sur une table de la base de données AssetCenter.

Note :

Cette fonction supporte un nombre limité de tables (workflows, agents, ordinateurs...).

Syntaxe Basic interne

Function DaExecAction(strAction As String, strTable As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strAction** : ce paramètre contient le nom SQL de l'action tel qu'il est défini sous AssetCenter.

- **strTable** : ce paramètre contient le nom SQL de la table sur laquelle l'action est exécutée.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim lErr as Long
lErr = DaExecAction( "MyAction", "amComputer")
```

DaExecuteActionByName()

Cette fonction exécute une action, identifiée par son nom, sur une table et un enregistrement de la base de données AssetCenter.

Syntaxe Basic interne

Function DaExecuteActionByName(strAction As String, strTable As String, lRecord As Long) As Long

Champ d'application

Version : 1.0

Utilisable
AssetCenter API

	Utilisable
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strAction** : ce paramètre contient le nom SQL de l'action tel qu'il est défini sous AssetCenter.
- **strTable** : ce paramètre contient le nom SQL de la table sur laquelle l'action est exécutée.
- **lRecord** : ce paramètre contient l'identifiant de l'enregistrement de la table sur lequel l'action est exécutée.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Remarques

Note :

Si aucun identifiant n'est spécifié, le numéro d'enregistrement lié à la table est utilisé.

Exemple

```
Dim lErr as Long
lErr = DaExecuteActionByName( "MyAction", "amComputer", 3)
```

DaFileATime()

Cette fonction renvoie la date et l'heure de dernier accès au fichier ou au répertoire.

Syntaxe Basic interne

Function DaFileATime(strPath As String) As Date

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strPath** : ce paramètre contient le chemin du fichier ou du répertoire concerné par l'opération. Si l'opération s'effectue sur le serveur de déploiement, il s'agit d'un chemin relatif par rapport au chemin absolu du dépôt de fichiers. Si l'opération s'effectue sur la cible de déploiement, il s'agit d'un chemin absolu.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

DaFileCRC()

Cette fonction renvoie le CRC (Contrôle de redondance cyclique) d'un fichier.

Syntaxe Basic interne

Function DaFileCRC(strPath As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strPath** : ce paramètre contient le chemin du fichier ou du répertoire concerné par l'opération. Si l'opération s'effectue sur le serveur de déploiement, il s'agit d'un chemin relatif par rapport au chemin absolu du

dépôt de fichiers. Si l'opération s'effectue sur la cible de déploiement, il s'agit d'un chemin absolu.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

DaFileCTime()

Cette fonction renvoie la date et l'heure de création du fichier ou du répertoire.

Syntaxe Basic interne

Function DaFileCTime(strPath As String) As Date

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	

Utilisable

Script FINISH.DO d'un assistant

Entrée

- **strPath** : ce paramètre contient le chemin du fichier ou du répertoire concerné par l'opération. Si l'opération s'effectue sur le serveur de déploiement, il s'agit d'un chemin relatif par rapport au chemin absolu du dépôt de fichiers. Si l'opération s'effectue sur la cible de déploiement, il s'agit d'un chemin absolu.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

DaFileLanguage()

Cette fonction renvoie la langue (si disponible) du fichier.

Syntaxe Basic interne

Function DaFileLanguage(strPath As String) As String

Champ d'application

Version : 1.0

Utilisable

AssetCenter API

	Utilisable
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strPath** : ce paramètre contient le chemin du fichier concerné par l'opération. Si l'opération s'effectue sur le serveur de déploiement, il s'agit d'un chemin relatif par rapport au chemin absolu du dépôt de fichiers. Si l'opération s'effectue sur la cible de déploiement, il s'agit d'un chemin absolu.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

DaFileMTime()

Cette fonction renvoie la date et l'heure de dernière modification du fichier ou du répertoire.

Syntaxe Basic interne

Function DaFileMTime(strPath As String) As Date

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strPath** : ce paramètre contient le chemin du fichier ou du répertoire concerné par l'opération. Si l'opération s'effectue sur le serveur de déploiement, il s'agit d'un chemin relatif par rapport au chemin absolu du dépôt de fichiers. Si l'opération s'effectue sur la cible de déploiement, il s'agit d'un chemin absolu.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

DaFileSize()

Cette fonction renvoie la taille d'un fichier en bits. Si la taille du fichier est trop importante (i.e elle ne tient pas dans un long), la valeur retournée est -1.

Syntaxe Basic interne

Function DaFileSize(strPath As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strPath** : ce paramètre contient le chemin du fichier concerné par l'opération. Si l'opération s'effectue sur le serveur de déploiement, il s'agit d'un chemin relatif par rapport au chemin absolu du dépôt de fichiers. Si l'opération s'effectue sur la cible de déploiement, il s'agit d'un chemin absolu.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

DaFileType()

Cette fonction renvoie le type du fichier sur lequel porte l'opération.

Syntaxe Basic interne

Function DaFileType(strPath As String) As String

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strPath** : ce paramètre contient le chemin du fichier ou du répertoire concerné par l'opération. Si l'opération s'effectue sur le serveur de déploiement, il s'agit d'un chemin relatif par rapport au chemin absolu du dépôt de fichiers. Si l'opération s'effectue sur la cible de déploiement, il s'agit d'un chemin absolu.

Sortie

La fonction retourne une des valeurs suivantes :

- "r" pour un fichier,
- "d" pour un répertoire,
- "D" pour un disque physique.

DaFileVersion()

Cette fonction renvoie la version (si disponible) du fichier.

Syntaxe Basic interne

Function DaFileVersion(strPath As String) As String

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strPath** : ce paramètre contient le chemin du fichier concerné par l'opération. Si l'opération s'effectue sur le serveur de déploiement, il s'agit d'un chemin relatif par rapport au chemin absolu du dépôt de fichiers. Si l'opération s'effectue sur la cible de déploiement, il s'agit d'un chemin absolu.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction

AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

DaFind()

Cette fonction effectue une recherche récursive de fichiers et de répertoires et constitue une liste des fichiers trouvés. La liste est ensuite parcourue au moyen de la fonction **DaFindNext**().

Syntaxe Basic interne

Function DaFind(strPath As String, strNameFilter As String, lDepth As Long) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strPath** : ce paramètre contient le chemin du répertoire de recherche.
- **strNameFilter** : ce paramètre contient des caractères jokers, au sens DOS (i.e * ou ?), utilisés pour filtrer la recherche.
- **lDepth** : ce paramètre contient la profondeur de la recherche récursive, en nombre de sous répertoires par rapport au répertoire **strPath**.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
' Builds a list of all bmp files
Dim lErr as Long
lErr = DaFind( "install", "*.bmp", 3)
```

DaFindNext()

Cette fonction récupère le nom suivant du fichier ou du répertoire dans la liste précédemment créée par **DaFind**(). Lorsque la fin de la liste est atteinte, la fonction renvoie une chaîne vide.

Syntaxe Basic interne

Function DaFindNext() As String

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓

Utilisable
Script d'un assistant
Script FINISH.DO d'un assistant

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim strFileName as String
strFileName = DaFindNext()
If strFileName <> 0 Then
  Print strFileName
Else
  Print "Reached end of file list"
End If
```

DaFirstEnv()

Cette fonction construit la liste des variables d'environnement disponible sur la cible de déploiement. La liste est alors parcourue au moyen des fonctions **DaNextEnv**() et **DaGetEnv**().

Syntaxe Basic interne

Function DaFirstEnv() As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
' Check that environment variables can be retrieved - this also retrie
ve all
' variables in one shot, improving performances
lRc = DaFirstEnv()

' Get usefull environment variables and store them in context
if lRc <> 0 then
 ' Get environment variables
 strProcArch      = DaGetEnv( "PROCESSOR_ARCHITECTURE" )
 strProcId        = DaGetEnv( "PROCESSOR_IDENTIFIER" )
 strProcLevel     = DaGetEnv( "PROCESSOR_LEVEL" )
 strProcRevision  = DaGetEnv( "PROCESSOR_REVISION" )
 strProcCount     = DaGetEnv( "NUMBER_OF_PROCESSORS" )
 strOs            = DaGetEnv( "OS" )
 strComputer      = DaGetEnv( "COMPUTERNAME" )

 ' Check that computer name is the right one
 if strComputer <> DaContext( "Computer.Name" ) then
   print "Warning: computer name does not look like the right one !"
 end if

 ' Store them in context
 DaSetContext "EnvInfo.ProcArch", strProcArch
```

```
DaSetContext "EnvInfo.ProcId", strProcId
DaSetContext "EnvInfo.ProcLevel", strProcLevel
DaSetContext "EnvInfo.ProcRevision", strProcRevision
DaSetContext "EnvInfo.ProcCount", strProcCount
DaSetContext "EnvInfo.Os", strOs
DaSetContext "EnvInfo.Computer", strComputer

' Dump
print DaDumpContext()
end if
```

DaGetEnv()

Cette fonction renvoie la valeur d'une variable d'environnement - contenue dans la liste construite au moyen de la fonction **DaFirstEnv**() - identifiée par son nom.

Syntaxe Basic interne

Function DaGetEnv(strEnv As String) As String

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strEnv** : ce paramètre contient le nom de la variable d'environnement tel qu'il est affiché par le système d'exploitation.

Astuce :

Pour afficher la liste des variables d'environnement déclarées sur votre ordinateur :

1 Cliquez-droit sur l'icône de votre ordinateur,

2 Sélectionnez l'onglet **Avancé**,

3 Cliquez sur le bouton **Variables d'environnement**

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
' Check that environment variables can be retrieved - this also retrie
ve all
' variables in one shot, improving performances
lRc = DaFirstEnv()

' Get useful environment variables and store them in context
if lRc <> 0 then
 ' Get environment variables
 strProcArch      = DaGetEnv( "PROCESSOR_ARCHITECTURE" )
 strProcId        = DaGetEnv( "PROCESSOR_IDENTIFIER" )
 strProcLevel     = DaGetEnv( "PROCESSOR_LEVEL" )
 strProcRevision  = DaGetEnv( "PROCESSOR_REVISION" )
 strProcCount     = DaGetEnv( "NUMBER_OF_PROCESSORS" )
 strOs            = DaGetEnv( "OS" )
 strComputer      = DaGetEnv( "COMPUTERNAME" )

 ' Check that computer name is the right one
 if strComputer <> DaContext( "Computer.Name" ) then
   print "Warning: computer name does not look like the right one !"
 end if

 ' Store them in context
 DaSetContext "EnvInfo.ProcArch", strProcArch
```

```
 DaSetContext "EnvInfo.ProcId", strProcId
 DaSetContext "EnvInfo.ProcLevel", strProcLevel
 DaSetContext "EnvInfo.ProcRevision", strProcRevision
 DaSetContext "EnvInfo.ProcCount", strProcCount
 DaSetContext "EnvInfo.Os", strOs
 DaSetContext "EnvInfo.Computer", strComputer

 ' Dump
 print DaDumpContext()
end if
```

DaGetFileInfo()

Cette fonction renvoie les propriétés d'un fichier ou d'un répertoire. Les propriétés en question peuvent alors être directement récupérées au moyen des fonctions **DaFile***. Elle possède deux modes opératoires :

- opération sur le serveur de déploiement (dans le dépôt de fichiers)
- opération sur la cible de déploiement

Le mode opératoire est déterminé par la valeur de l'option **file.on_server**, fixée au moyen de la fonction **DaSetOption**() :

- si cette option a pour valeur 1, l'opération porte sur le serveur de déploiement
- si cette option a pour valeur 0, l'opération porte sur la cible de déploiement.

 Note :

Pour plus d'informations, nous vous invitons à consulter le descriptif de la fonction **DaSetOption**().

Syntaxe Basic interne

Function DaGetFileInfo(strPath As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strPath** : ce paramètre contient le chemin du fichier ou du répertoire concerné par l'opération. Si l'opération s'effectue sur le serveur de déploiement, il s'agit d'un chemin relatif par rapport au chemin absolu du dépôt de fichiers. Si l'opération s'effectue sur la cible de déploiement, il s'agit d'un chemin absolu.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

DaImpersonate()

Cette fonction permet de réaliser une impersonification sur la cible de déploiement. Il s'agit de simuler la connexion d'un utilisateur à la cible. Cette fonction peut être particulièrement utile pour simuler la connexion d'un administrateur et pouvoir, par exemple, installer certains logiciels sur la cible.

Syntaxe Basic interne

Function DaImpersonate(strUser As String, strPassword As String, strDomain As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strUser** : ce paramètre contient le nom de l'utilisateur dont vous souhaitez simuler la connexion à la cible.
- **strPassword** : ce paramètre contient le mot de passe de l'utilisateur dont vous souhaitez simuler la connexion à la cible.
- **strDomain** : ce paramètre contient le domaine de l'utilisateur dont vous souhaitez simuler la connexion à la cible.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
' Upload whole 'install' directory
lErr = DaUpload( "install", "c:/tmp/install", "" )

' Impersonate
if lErr = 0 then lErr = DaImpersonate( "user", "password", "domain " )

' Execute setup as user, synchronous, and log errors
if lErr = 0 then
  lDummyErr = DaSetOption( "exec.synchronous", 1 )
  lDummyErr = DaSetOption( "exec.log_output", 1 )
  lDummyErr = DaSetOption( "exec.log_error", 1 )
end if
if lErr = 0 then lErr = DaExec( "c:/tmp/install/setup -i", "c:/tmp/ins
tall" )

' On error, raise 'error' event
if lErr <> 0 then DaSetReturnValue "ErrorEvent"
```

DaMkDir()

Cette fonction crée un répertoire. Elle possède deux modes opératoires :

- création sur le serveur de déploiement (dans le dépôt de fichiers)
- création sur la cible de déploiement

Le mode opératoire est déterminé par la valeur de l'option **file.on_server**, fixée au moyen de la fonction **DaSetOption**() :

- si cette option a pour valeur 1, la création porte sur le serveur de déploiement
- si cette option a pour valeur 0, la création porte sur la cible de déploiement.

Note :

Pour plus d'informations, nous vous invitons à consulter le descriptif de la fonction **DaSetOption**().

Syntaxe Basic interne

Function DaMkDir(strDirectory As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strDirectory** : ce paramètre contient le chemin du répertoire à créer. Si la création s'effectue sur le serveur de déploiement, il s'agit d'un chemin relatif par rapport au chemin absolu du dépôt de fichiers. Si la création s'effectue sur la cible de déploiement, il s'agit d'un chemin absolu.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

DaMove()

Cette fonction déplace un fichier, un ensemble de fichiers ou un répertoire. Cette fonction possède deux modes opératoires :

- déplacement local au serveur de déploiement (la source et la destination du déplacement se trouvent toutes deux sur le serveur de déploiement)

- déplacement local à la cible de déploiement (la source et la destination du déplacement se trouvent toutes deux sur la cible de déploiement)

Le mode opératoire est déterminé par la valeur de l'option **file.on_server**, fixée au moyen de la fonction **DaSetOption**() :

- si cette option a pour valeur 1, le déplacement porte sur le serveur de déploiement
- si cette option a pour valeur 0, le déplacement porte sur la cible de déploiement.

Note :

Pour plus d'informations, nous vous invitons à consulter le descriptif de la fonction **DaSetOption**().

Note :

Cette fonction effectue les mêmes opérations que l'activité **Déplacement de fichiers**.

Syntaxe Basic interne

Function DaMove(strSource As String, strDest As String, strNameFilter As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	

Utilisable

Script FINISH.DO d'un assistant

Entrée

- **strSource** : ce paramètre contient le chemin source des éléments à déplacer. Suivant le mode opératoire de la fonction il s'agit soit d'un chemin relatif au chemin du dépôt de fichier (dans le cas d'un déplacement local au serveur de déploiement), soit d'un chemin absolu (dans le cas d'un déplacement local à la cible de déploiement).
- **strDest** : ce paramètre contient le chemin destination des éléments à déplacer. Suivant le mode opératoire de la fonction il s'agit soit d'un chemin relatif au chemin du dépôt de fichier (dans le cas d'un déplacement local au serveur de déploiement), soit d'un chemin absolu (dans le cas d'un déplacement local à la cible de déploiement).
- **strNameFilter** : caractères joker de type DOS (***** et **?**) utilisés pour filtrer les noms de fichiers.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
' move old report files before getting new ones
Dim l as Long
l = DaSetOption( "file.on_server", 1 );
l = DaMove( "reports/current", "reports/old", "" )
```

DaNetIpFromName()

Cette fonction effectue une résolution DNS sur le nom d'une cible de déploiement et retourne l'adresse IP de la cible.

Syntaxe Basic interne

Function DaNetIpFromName(strHost As String, pstrIP As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strHost** : ce paramètre contient le nom de l'hôte (cible de déploiement).
- **pstrIP** : adresse IP de la cible

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

DaNetNBTName()

Cette fonction effectue une requête sur une cible de déploiement pour récupérer certaines informations réseau de la cible. Cette fonction est strictement équivalente à la commande DOS :

```
nbstat -A <Adresse IP de la cible >
```

Syntaxe Basic interne

Function DaNetNBTName(strHost As String, lTimeout As Long, pstrComputerName As String, pstrGroup As String, pstrMAC As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strHost** : ce paramètre contient le nom de l'hôte (cible de déploiement).
- **lTimeout** : ce paramètre contient la durée au bout de laquelle l'opération est considérée comme un échec, si la cible n'a pas répondu. Cette durée est exprimée en millisecondes.
- **pstrComputerName** : nom de la cible
- **pstrGroup** : groupe réseau de la cible
- **pstrMAC** : adresse MAC de la cible

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

DaNetPing()

Cette fonction réalise un ping sur la cible de déploiement. Le ping permet de tester la présence d'une machine sur le réseau.

Syntaxe Basic interne

Function DaNetPing(lTimeout As Long) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	

Entrée

- **lTimeout** : ce paramètre contient la durée au bout de laquelle l'opération est considérée comme un échec, si la cible n'a pas répondu. Cette durée est exprimée en millisecondes.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim lLive As Long
' Ping with 1000ms timeout
lLive = DaNetPing( 1000 )
If lLive = 0 Then
  DaSetReturnValue "NOK"
End If
```

DaNetWakeOnLan()

Cette fonction opère un éveil par réseau sur la cible de déploiement.

Note :

Cette fonction effectue les mêmes opérations que l'activité **Eveil par appel réseau**.

Syntaxe Basic interne

Function DaNetWakeOnLan(strMAC As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strMAC** : ce paramètre contient l'adresse MAC de la cible de déploiement, au format hexadécimal. L'adresse MAC est composée de six paires hexadécimales séparées (au moyen des caractères "-" ou ":") ou non.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

DaNetWinAddressByName()

Cette fonction effectue une résolution sur le nom de la cible en utilisant un mode de résolution spécifié. La fonction retourne l'adresse IP de la cible.

Syntaxe Basic interne

Function DaNetWinAddressByName(strHost As String, iNameSpace As Long, pstrIP As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strHost** : ce paramètre contient le nom de l'hôte (cible de déploiement).
- **iNameSpace** : ce paramètre précise le mode de résolution à utiliser. Les valeurs possibles sont les suivantes :
  - 0 : utilise le mode de résolution par défaut du système d'exploitation
  - 1 : utilise une résolution DNS
  - 2 : utilise une résolution WINS
- **pstrIP** : ce paramètre contient le nom de l'hôte (cible de déploiement).

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

DaNextEnv()

Cette fonction renvoie le nom de la prochaine variable d'environnement de la liste construite au moyen de la fonction **DaFirstEnv**(). Si la fin de la liste est atteinte, la fonction renvoie une chaîne de caractères vide.

Syntaxe Basic interne

Function DaNextEnv() As String

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
'
' retrieve and print all environment variables from target
'

dim lRc as long
dim VarName, VarValue as String

If DaFirstEnv() <> 0 Then
  Do
    ' get next variable
    VarName = DaNextEnv()
    If VarName <> "" Then
      VarValue = DaGetEnv( VarName )
      Print VarName & "=" & VarValue
    Else
     Exit Do
    End If
  Loop
End If
```

DaNTFileCopyTo()

Cette fonction copie un fichier d'un répertoire source à un répertoire destination, appartenant tous deux à la cible de déploiement.

Syntaxe Basic interne

Function DaNTFileCopyTo(strSrcPath As String, strSrc As String, strDst As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	

	Utilisable
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strSrcPath** : ce paramètre contient le chemin complet du répertoire dans lequel est stocké le fichier à copier.
- **strSrc** : ce paramètre contient le nom du fichier à copier.
- **strDst** : ce paramètre contient le chemin complet du répertoire destination du fichier à copier.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

DaNTFileCreateDir()

Cette fonction crée récursivement un répertoire sur une cible de déploiement.

Syntaxe Basic interne

Function DaNTFileCreateDir(strPath As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓

	Utilisable
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strPath** : ce paramètre contient le chemin complet du répertoire sur la machine distante.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

DaNTFileDelete()

Cette fonction efface des fichiers sur une cible de déploiement.

Syntaxe Basic interne

Function DaNTFileDelete(strFile As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	

	Utilisable
Script FINISH.DO d'un assistant	

Entrée

- **strFile** : chemin du fichier à effacer.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

DaNTFileDeleteDir()

Cette fonction efface un répertoire et son contenu sur la cible de déploiement.

Syntaxe Basic interne

Function DaNTFileDeleteDir(strPathConst As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	

	Utilisable
Script FINISH.DO d'un assistant	

Entrée

- **strPathConst** : chemin du répertoire à effacer.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

DaNTFileDirCopyTo()

Cette fonction copie un répertoire et son contenu sur une cible de déploiement.

Syntaxe Basic interne

Function DaNTFileDirCopyTo(strSrcPath As String, strDst As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	

	Utilisable
Script FINISH.DO d'un assistant	

Entrée

- **strSrcPath** : ce paramètre contient le chemin du répertoire sur le serveur.
- **strDst** : ce paramètre contient le chemin du répertoire destination sur la cible de déploiement.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

DaNTFileDirDownload()

Cette fonction copie un répertoire et son contenu depuis une cible de déploiement, sur le serveur de déploiement.

Syntaxe Basic interne

Function DaNTFileDirDownload(strSrcPath As String, strDst As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	

	Utilisable
Script FINISH.DO d'un assistant	

Entrée

- **strSrcPath** : ce paramètre contient le chemin du répertoire sur la cible de déploiement.
- **strDst** : ce paramètre contient le chemin du répertoire destination sur le serveur. Ce chemin est relatif par rapport au chemin du dépôt de fichiers.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

DaNTFileDirUpload()

Cette fonction copie un répertoire et son contenu sur une cible de déploiement.

Syntaxe Basic interne

Function DaNTFileDirUpload(strSrcPath As String, strDst As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	

	Utilisable
Script FINISH.DO d'un assistant	

Entrée

- **strSrcPath** : ce paramètre contient le chemin du répertoire sur le serveur de déploiement. Ce chemin est relatif par rapport au chemin du dépôt de fichiers.
- **strDst** : ce paramètre contient le chemin du répertoire destination sur la cible de déploiement.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

DaNTFileDownload()

Cette fonction réceptionne un fichier provenant de la cible de déploiement.

Syntaxe Basic interne

Function DaNTFileDownload(strSrcPath As String, strSrc As String, strDst As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓

Utilisable
Script d'un assistant
Script FINISH.DO d'un assistant

Entrée

- **strSrcPath** : ce paramètre contient le chemin complet du répertoire (sur la cible) dans lequel est stocké le fichier à réceptionner.
- **strSrc** : ce paramètre contient le nom du fichier à réceptionner.
- **strDst** : ce paramètre contient le chemin complet du répertoire (sur le serveur) dans lequel le fichier est réceptionné.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

DaNTFileUpload()

Cette fonction envoie un fichier sur la cible de déploiement.

Syntaxe Basic interne

Function DaNTFileUpload(strSrcPath As String, strSrc As String, strDst As String) As Long

Champ d'application

Version : 1.0

Utilisable
AssetCenter API
Script de configuration d'un champ ou d'un lien
Action de type "Script"

	Utilisable
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strSrcPath** : ce paramètre contient le chemin complet du répertoire (sur le serveur) dans lequel est stocké le fichier à envoyer.
- **strSrc** : ce paramètre contient le nom du fichier à envoyer.
- **strDst** : ce paramètre contient le chemin complet du répertoire (sur la cible) dans lequel le fichier est envoyé.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

DaNTRegistryLMAddStringValue()

Cette fonction ajoute une entrée de type **REG_SZ** ou **REG_EXPAND** pour une clé.

Syntaxe Basic interne

Function DaNTRegistryLMAddStringValue(strKey As String, strString As String, strValue As String, bExpand As Long) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	

	Utilisable
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strKey** : ce paramètre contient le nom complet de la clé contenant l'entrée à créer.
- **strString** : ce paramètre contient le nom de l'entrée à créer.
- **strValue** : ce paramètre contient la valeur de l'entrée à créer.
- **bExpand** : si ce paramètre a pour valeur 1, la fonction traite une entrée de type **REG_EXPAND** uniquement.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

DaNTRegistryLMCreateKey()

Cette fonction crée récursivement une clé sous la ruche **HKEY_LOCAL_MACHINE** de la base de registres.

Syntaxe Basic interne

Function DaNTRegistryLMCreateKey(strKey As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strKey** : ce paramètre contient le chemin complet de la clé sous la ruche **HKEY_LOCAL_MACHINE** de la base de registres.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

DaNTRegistryLMDeleteKey()

Cette fonction efface une clé de la base de registres de la cible de déploiement. Toutes les entrées de la clé sont effacées.

Syntaxe Basic interne

Function DaNTRegistryLMDeleteKey(strKey As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	

	Utilisable
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strKey** : ce paramètre contient le nom complet de la clé à effacer.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

DaNTRegistryLMDeleteValue()

Cette fonction efface une entrée d'une clé de la base de registres de la cible de déploiement.

Syntaxe Basic interne

Function DaNTRegistryLMDeleteValue(strKey As String, strString As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	

	Utilisable
Script FINISH.DO d'un assistant	

Entrée

- **strKey** : ce paramètre contient le nom complet de la clé contenant l'entrée à effacer.
- **strString** : ce paramètre contient le nom de l'entrée à effacer.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

DaNTRegistryLMGetLongValue()

Cette fonction retourne la valeur d'une entrée de type **REG_DWORD** d'une clé de la base de registres de la cible de déploiement.

Syntaxe Basic interne

Function DaNTRegistryLMGetLongValue(strKey As String, strString As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	

Entrée

- **strKey** : ce paramètre contient le nom complet de la clé contenant l'entrée dont vous souhaitez récupérer la valeur.
- **strString** : ce paramètre contient le nom de l'entrée dont vous souhaitez récupérer la valeur.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

DaNTRegistryLMGetStringValue()

Cette fonction retourne la valeur d'une entrée de type **REG_DWORD** ou **REG_EXPAND** d'une clé de la base de registres de la cible de déploiement.

Syntaxe Basic interne

Function DaNTRegistryLMGetStringValue(strKey As String, strString As String) As String

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strKey** : ce paramètre contient le nom complet de la clé contenant l'entrée dont vous souhaitez récupérer la valeur.
- **strString** : ce paramètre contient le nom de l'entrée dont vous souhaitez récupérer la valeur.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

DaNTRegistryLMSetLongValue()

Cette fonction fixe la valeur d'une entrée de type **REG_DWORD** pour une clé. L'entrée doit exister dans la clé.

Syntaxe Basic interne

Function DaNTRegistryLMSetLongValue(strKey As String, strString As String, lValue As Long) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strKey** : ce paramètre contient le nom complet de la clé contenant l'entrée dont vous souhaitez fixer la valeur.
- **strString** : ce paramètre contient le nom de l'entrée dont vous souhaitez fixer la valeur.
- **lValue** : ce paramètre contient la valeur de l'entrée dont vous souhaitez fixer la valeur.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

DaNTRegistryLMSetStringValue()

Cette fonction fixe la valeur d'une entrée de type **REG_SZ** ou **REG_EXPAND** pour une clé. L'entrée doit exister dans la clé.

Syntaxe Basic interne

Function DaNTRegistryLMSetStringValue(strKey As String, strString As String, strValue As String, bExpand As Long) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strKey** : ce paramètre contient le nom complet de la clé contenant l'entrée dont vous souhaitez fixer la valeur.
- **strString** : ce paramètre contient le nom de l'entrée dont vous souhaitez fixer la valeur.
- **strValue** : ce paramètre contient la valeur de l'entrée dont vous souhaitez fixer la valeur.
- **bExpand** : si ce paramètre a pour valeur 1, la fonction traite une entrée de type **REG_EXPAND** uniquement.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

DaNTServiceInstall()

Cette fonction installe un service NT sur la cible de déploiement.

Syntaxe Basic interne

Function DaNTServiceInstall(strServiceName As String, strDisplayName As String, strBinaryFullName As String, bAutoStart As Long, bInteractWithDesktop As Long) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strServiceName** : ce paramètre contient le nom interne du service, tel qu'il sera affiché dans les propriétés du service (accessibles via le menu contextuel **Propriétés**).
- **strDisplayName** : ce paramètre contient le nom du service tel qu'il sera affiché dans la liste principale des services.
- **strBinaryFullName** : ce paramètre contient le chemin complet, sur la cible de déploiement, de l'exécutable du service.
- **bAutoStart** : lorsque ce paramètre a pour valeur 1, le démarrage du service est automatique.
- **bInteractWithDesktop** : lorsque ce paramètre a pour valeur 1, le service peut interagir avec le bureau du système d'exploitation (envoi de message et de boîte de dialogues à l'utilisateur).

Sortie

- 0 : La fonction s'est exécutée normalement.

- Non nul : Code d'erreur.

DaNTServiceStart()

Cette fonction démarre un service NT nommé sur la cible de déploiement.

Syntaxe Basic interne

Function DaNTServiceStart(strServiceName As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strServiceName** : ce paramètre contient le nom du service à démarrer. Le nom utilisé ici est le nom interne du service et non le nom affiché dans la console des services du système d'exploitation. Pour trouver ce nom interne :
  1. Cliquez-droit sur le nom du service,
  2. Sélectionnez le menu **Propriétés**,
  3. Le **Nom du service** (nom interne) est affiché dans l'onglet **Général**.

Sortie

- 0 : La fonction s'est exécutée normalement.

- Non nul : Code d'erreur.

DaNTServiceStatus()

Cette fonction renvoie l'état d'un service NT sur la cible de déploiement.

Syntaxe Basic interne

Function DaNTServiceStatus(strServiceName As String) As String

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strServiceName** : ce paramètre contient le nom interne du service, tel qu'il sera affiché dans les propriétés du service (accessibles via le menu contextuel **Propriétés**).

Sortie

La fonction renvoie une des valeurs suivantes, correspondant chacune à un état pour le service :

- NOT FOUND
- STOPPED

- START_PENDING
- STOP_PENDING
- RUNNING
- CONTINUE_PENDING
- PAUSE_PENDING
- PAUSED

DaNTServiceStop()

Cette fonction arrête l'exécution d'un service NT nommé sur la cible de déploiement.

Syntaxe Basic interne

Function DaNTServiceStop(strServiceName As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strServiceName** : ce paramètre contient le nom du service à stopper. Le nom utilisé ici est le nom interne du service et non le nom affiché dans la console des services du système d'exploitation. Pour trouver ce nom interne :
  1 Cliquez-droit sur le nom du service,

2 Sélectionnez le menu **Propriétés**,

3 Le **Nom du service** (nom interne) est affiché dans l'onglet **Général**.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

DaNTServiceUninstall()

Cette fonction désinstalle un service NT sur la cible de déploiement.

Syntaxe Basic interne

Function DaNTServiceUninstall(strServiceName As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strServiceName** : ce paramètre contient le nom interne du service, tel qu'il est affiché dans les propriétés du service (accessibles via le menu contextuel **Propriétés**).

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

DaNTWMIExecMethod()

Cette fonction permet d'utiliser une méthode WMI sur un objet ou une classe.

Syntaxe Basic interne

Function DaNTWMIExecMethod(strObjectPath As String, strMethod As String, strInParams As String, pstrOutParam As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strObjectPath** : nom de l'objet ou de la classe.
- **strMethod** : nom de la méthode (CIMV2 namespace).
- **strInParams** : liste des paramètres (input) de la méthode
- **pstrOutParam** : description au format texte du résultat de sortie (output result).

  Par exemple le résultat de la méthode **Créer** appliquée à la classe WIN32_Process sera de la forme :

```
instance of __PARAMETERS
{
 ProcessId = 400;
 ReturnValue = 0;
};
```

Note :

La chaîne de caractère retournée contient des sauts de ligne.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
lCount = DaNTWMIGetInstanceCount( "Win32_Process" )
 For l = lCount - 1 to 0 Step -1
   strName = DaNTWMIGetPropertyValue( "Win32_Process.Name", l )
   If strName = "NOTEPAD.EXE" Then
     strPID = DaNTWMIGetPropertyValue( "Win32_Process.Handle", l )
     lErr = DaNTWMIExecMethod( "Win32_Process.Handle='" & strPID & "'"
, "Terminate", strDummy )
   End If
 Next l
```

Ce script arrête l'ensemble des processus NOTEPAD.EXE grâce à la méthode **Terminate**.

DaNTWMIExecQuery()

Cette fonction permet d'énumérer les objets WMI par l'utilisation d'une requête WMI en langage WQL.

Syntaxe Basic interne

Function DaNTWMIExecQuery(strQuery As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strQuery** : requête WQL

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

Exemple

```
lErr = DaNTWMIExecQuery( "select * from Win32_NTLogEvent WHERE Logfile
='Security'" )
While DaNTWMINextItem()
 print DaNTWMIGetCurrentPropertyValue( "Message", i )
Wend
lErr = DaNTWMIResetEnumeration()
```

Ce script énumère l'ensemble des événements du journal d'événements de la sécurité NT et recherche l'événement spécifié.

DaNTWMIGetCurrentArrayValue()

Cette fonction est utilisée en relation avec la fonction **DANTWMIExecQuery**. Elle récupère les valeurs des propriétés (de type tableau) de l'objet énuméré.

La fonction retourne la liste des valeurs de la propriété, séparées par un caractère défini dans le paramètre **strSeparator**.

Syntaxe Basic interne

Function DaNTWMIGetCurrentArrayValue(strProperty As String, strSeparator As String) As String

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strProperty** : ce paramètre contient le nom de la propriété (de l'objet WMI) dont vous souhaitez récupérer la valeur.
- **strSeparator** : ce paramètre contient le caractère utilisé comme séprateur pour les valeurs de la propriété.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim lErr As Long
lErr = DaNTWMIExecQuery( "SELECT * FROM Win32_NetworkAdapterConfigurat
ion WHERE IPEnabled=TRUE" )
While DaNTWMINextItem()
  print DaNTWMIGetCurrentArrayValue( "IPAddress", ";" )
Wend
```

DaNTWMIGetCurrentPropertyValue()

Cette fonction est utilisée en relation avec la fonction **DANTWMIExecQuery**. Elle récupère les valeurs des propriétés de l'objet énuméré.

Note :

Pour traiter une propriété de type tableau (de valeurs), utilisez la fonction **DaNTWMIGetCurrentArrayValue**().

Syntaxe Basic interne

Function DaNTWMIGetCurrentPropertyValue(strProperty As String) As Variant

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	

	Utilisable
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strProperty** : ce paramètre contient le nom de la propriété (de l'objet WMI) dont vous souhaitez récupérer la valeur.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

DaNTWMIGetInstanceCount()

Cette fonction renvoie le nombre d'objets d'une classe donnée (par exemple le nombre de disques durs, de processeurs, ... d'une machine).

Syntaxe Basic interne

Function DaNTWMIGetInstanceCount(strShortName As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strShortName** : ce paramètre contient le nom de l'objet dont vous souhaitez compter le nombre d'instances. La liste des objets disponibles est répertoriée dans le tableau ci-dessous :

Propriété
Win32_Processor
Win32_Processor
Win32_PhysicalMemory
Win32_OperatingSystem
Win32_DiskDrive
Win32_LogicalDisk
Win32_Operatingsystem
Win32_Operatingsystem.BuildNumber
Win32_NetworkAdapter
Win32_NetworkAdapterConfiguration

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

Exemple

L'exemple suivant renvoie le nombre de disques durs sur la machine :

```
Dim iHDCount as Integer
iHDCount = DaNTWMIGetInstanceCount("Win32_LogicalDisk")
print "Number of hard drives = " & iHDCount
```

DaNTWMIGetPropertyValue()

Cette fonction récupère la valeur d'une propriété d'un objet WMI.

Syntaxe Basic interne

Function DaNTWMIGetPropertyValue(strShortName As String, idx As Long) As Variant

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strShortName** : ce paramètre contient le nom de la propriété dont vous souhaitez récupérer la valeur. La syntaxe utilisée est de la forme :

  ```
  <objet>.<propriété>
  ```

  Vous pouvez également utiliser les alias répertoriés dans le tableau ci-dessous :

Alias	Propriété référencée
CPUType	Win32_Processor.Family

Alias	Propriété référencée
CPUInternal	Win32_Processor.Caption
CPUSpeed	Win32_Processor.CurrentClockSpeed
MemorySize	Win32_PhysicalMemory.Capacity
OSMemorySize	Win32_OperatingSystem.TotalVisibleMemorySize
TotalDiskSize	Wind32_DiskDrive.Size
DiskSize	Win32_LogicalDisk.Size
DiskFreeSpace	Win32_LogicalDisk.FreeSpace
OperatingSystem	Win32_Operatingsystem.Caption
OSServiceLevel	Win32_Operatingsystem.ServicePackMajorVersion
OSBuildNumber	Win32_Operatingsystem.BuildNumber
PhysAddress	Win32_NetworkAdapter.MACAddress
DNSHostName	Win32_NetworkAdapterConfiguration.DNSHostName
DNSDomain	Win32_NetworkAdapterConfiguration.DNSDomain

- **idx** : ce paramètre contient l'index de l'objet concerné par l'opération. Si par exemple vous possédez plusieurs disques durs, chacun d'eux possède un numéro d'index.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

L'exemple suivant récupère la fréquence du processeur de la machine :

```
Dim vSpeed as Variant
vSpeed = DaNTWMIGetPropertyValue("CPUSPeed", 0)
print "Speed = " & vSpeed
```

DaNTWMIGetTotalPropertiesValue()

Cette fonction renvoie la somme des valeurs d'une propriété pour tous les objets possédant cette propriété. Vous pouvez ainsi récupérer l'espace disque disponible sur l'ensemble des disques logiques de la machine.

Syntaxe Basic interne

Function DaNTWMIGetTotalPropertiesValue(strShortName As String) As Variant

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strShortName** : ce paramètre contient le nom de la propriété dont vous souhaitez récupérer la valeur. La syntaxe utilisée est de la forme :

```
<objet>.<propriété>
```

Vous pouvez également utiliser les alias répertoriés dans le tableau ci-dessous :

Alias	Propriété référencée
CPUType	Win32_Processor.Family
CPUInternal	Win32_Processor.Caption
CPUSpeed	Win32_Processor.CurrentClockSpeed

Alias	Propriété référencée
MemorySize	Win32_PhysicalMemory.Capacity
OSMemorySize	Win32_OperatingSystem.TotalVisibleMemorySize
TotalDiskSize	Wind32_DiskDrive.Size
DiskSize	Win32_LogicalDisk.Size
DiskFreeSpace	Win32_LogicalDisk.FreeSpace
OperatingSystem	Win32_Operatingsystem.Caption
OSServiceLevel	Win32_Operatingsystem.ServicePackMajorVersion
OSBuildNumber	Win32_Operatingsystem.BuildNumber
PhysAddress	Win32_NetworkAdapter.MACAddress
DNSHostName	Win32_NetworkAdapterConfiguration.DNSHostName
DNSDomain	Win32_NetworkAdapterConfiguration.DNSDomain

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

L'exemple suivant renvoie la capacité totale de tous les disques logiques de la machine :

```
Dim vCapacity as Variant
vCapacity = DaNTWMIGetTotalPropertiesValue("WIN32_PhysicalMemory.Capac
ity")
print "Physical memory = " & vCapacity
```

DaNTWMINextItem()

Cette fonction est utilisée en relation avec la fonction **DANTWMIExecQuery**. Elle énumère l'ensemble des éléments retournés par une requête WQL.

Syntaxe Basic interne

Function DaNTWMINextItem() As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

DaNTWMIResetEnumeration()

Cette fonction est utilisée en relation avec la fonction **DANTWMIExecQuery**. Elle réinitialise l'énumération en cours d'ouverture ainsi que la mémoire libre correspondante.

Syntaxe Basic interne

Function DaNTWMIResetEnumeration() As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

DaRegCreateKey()

Cette fonction crée une une clé dans la base de registres de la cible de déploiement.

Syntaxe Basic interne

Function DaRegCreateKey(strKey As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	

	Utilisable
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strKey** : ce paramètre contient le nom complet de la clé à créer.

  Exemple :

```
HKEY_LOCAL_MACHINE\SOFTWARE\Peregrine Systems\Automated Desktop Adm
inistration\6.0.0
```

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

DaRegDeleteKey()

Cette fonction efface une une clé de la base de registres de la cible de déploiement.

Syntaxe Basic interne

Function DaRegDeleteKey(strKey As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strKey** : ce paramètre contient le nom complet de la clé à effacer.

```
HKEY_LOCAL_MACHINE\SOFTWARE\Peregrine Systems\Automated Desktop Administration\6.0.0
```

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

DaRegExec()

Cette fonction exécute plusieurs opérations (définies dans un script) sur la base de registres de la cible. Le script utilise la même syntaxe que les fichiers **.reg** du système d'exploitation.

 Note :

Cette fonction correspond à l'activité de type **Registre**.

Syntaxe Basic interne

Function DaRegExec(strScript As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strScript** : ce paramètre contient le script à exécuter sur la base de registres.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

DaRegGetValue()

Cette fonction retourne la valeur d'une entrée d'une clé de la base de registres de la cible de déploiement.

Syntaxe Basic interne

Function DaRegGetValue(strKey As String, strValue As String) As Variant

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strKey** : ce paramètre contient le nom complet de la clé concernée par l'opération.
- **strValue** : ce paramètre contient le nom de l'entrée de la clé dont vous souhaitez récupérer la valeur.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

DaRegOutputValue()

Cette fonction récupère la valeur du contenu de **DaRegOutput**.

Syntaxe Basic interne

Function DaRegOutputValue(strKey As String, strValue As String) As Variant

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strKey** : ce paramètre contient le nom complet de la clé concernée par l'opération. Si ce paramètre est vide, n'importe qu'elle clé sera utilisée.
- **strValue** : ce paramètre contient la chaîne à convertir. Si ce paramètre est vide, la valeur par défaut sera utilisée.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction

AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

DaRegSetValue()

Cette fonction définit une valeur pour une entrée d'une clé de la base de registres de la cible de déploiement.

Syntaxe Basic interne

Function DaRegSetValue(strKey As String, strValue As String, vValue As Variant) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strKey** : ce paramètre contient le nom complet de la clé dont vous souhaitez fixer la valeur.
- **strValue** : ce paramètre contient le nom de l'entrée à définir.
- **vValue** : ce paramètre contient la valeur à affecter à l'entrée. Si vous ne précisez aucune valeur pour ce paramètre, la valeur existante pour l'entrée est effacée.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

DaRegStrValue()

Cette fonction convertit une valeur exprimée sous la forme d'un variant en une expression compatible avec le format de la base de registres (sous la forme d'une chaîne).

Syntaxe Basic interne

Function DaRegStrValue(vValue As Variant) As String

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **vValue** : ce paramètre contient la valeur à convertir

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

DaRegVarValue()

Cette fonction convertit une valeur de type chaîne sous la forme d'un variant.

Syntaxe Basic interne

Function DaRegVarValue(strValue As String) As Variant

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strValue** : ce paramètre contient la chaîne à convertir.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

DaRename()

Cette fonction renomme un fichier ou un répertoire. Elle possède deux modes opératoires :

- renommage sur le serveur de déploiement (dans le dépôt de fichiers)
- renommage sur la cible de déploiement

Le mode opératoire est déterminé par la valeur de l'option **file.on_server**, fixée au moyen de la fonction **DaSetOption**() :

- si cette option a pour valeur 1, le renommage porte sur le serveur de déploiement
- si cette option a pour valeur 0, le renommage porte sur la cible de déploiement.

Note :

Pour plus d'informations, nous vous invitons à consulter le descriptif de la fonction **DaSetOption**().

Syntaxe Basic interne

Function DaRename(strSource As String, strDest As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strSource** : ce paramètre contient le chemin du fichier source ou du répertoire à renommer. Si l'opération s'effectue sur le serveur de déploiement, il s'agit d'un chemin relatif par rapport au chemin absolu du dépôt de fichiers. Si l'opération s'effectue sur la cible de déploiement, il s'agit d'un chemin absolu.
- **strDest** : ce paramètre contient le nouveau nom du fichier ou du répertoire sous la forme d'un chemin complet. Si l'opération s'effectue sur le serveur de déploiement, il s'agit d'un chemin relatif par rapport au chemin absolu du dépôt de fichiers. Si l'opération s'effectue sur la cible de déploiement, il s'agit d'un chemin absolu.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

DaReturnValue()

Cette fonction renvoie le nom de l'événement de sortie de l'activité courante du workflow de déploiement.

Syntaxe Basic interne

Function DaReturnValue() As String

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
' Set a return value if not already set
If DaReturnValue = "" Then
  SetDaReturnValue( "MyEvent" )
End If
```

DaRmDir()

Cette fonction efface un répertoire et son contenu. Elle possède deux modes opératoires :

- effacement sur le serveur de déploiement (dans le dépôt de fichiers)
- effacement sur la cible de déploiement

Le mode opératoire est déterminé par la valeur de l'option **file.on_server**, fixée au moyen de la fonction **DaSetOption**() :

- si cette option a pour valeur 1, l'effacement porte sur le serveur de déploiement
- si cette option a pour valeur 0, l'effacement porte sur la cible de déploiement.

Note :

Pour plus d'informations, nous vous invitons à consulter le descriptif de la fonction **DaSetOption**().

Syntaxe Basic interne

Function DaRmDir(strDirectory As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	

Entrée

- **strDirectory** : ce paramètre contient le chemin du répertoire à effacer. Si l'effacement s'effectue sur le serveur de déploiement, il s'agit d'un chemin relatif par rapport au chemin absolu du dépôt de fichiers. Si l'effacement s'effectue sur la cible de déploiement, il s'agit d'un chemin absolu.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

DaSendMail()

Cette fonction envoie un e-mail. L'opération étant effectuée par le serveur de déploiement, toutes les paramètres de configuration associés à l'envoi d'un e-mail doivent être configurés sur le serveur.

Ls protocoles supportés sont les suivants :

- SMTP : les adresses sont alors du type :

```
SMTP:[nom@adresse.domaine]
```

- MAPI : les adresses sont alors du type :

```
MAPI:[nom de la boîte aux lettres]
```

- VIM : les adresses sont alors du type :

```
VIM:[nom/domaine]
```

- AM : messagerie interne (correspondant aux personnes référencées dans la base de données). Les adresses sont alors du type :

```
AM:[login du destinataire]
```

Note :

En cas de destinataires multiples, chaque adresse est séparée par une virgule (",").

Syntaxe Basic interne

Function DaSendMail(strTo As String, strCc As String, strBcc As String, strSubject As String, strMessage As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strTo** : Ce paramètre contient la liste des adresses des destinataires du message.
- **strCc** : Ce paramètre contient la liste des adresses des destinataires en copie du message.
- **strBcc** : Ce paramètre contient la liste des adresses des destinataires en copie cachée du message (ils n'apparaissent pas dans la liste des destinataires)
- **strSubject** : Ce paramètre contient l'intitulé du message.
- **strMessage** : Ce paramètre contient le corps du message.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

DaSetContext()

Cette fonction définit un nouveau contexte dans la liste des contextes disponibles pour un workflow de déploiement.

Syntaxe Basic interne

Function DaSetContext(strField As String, vValue As Variant)

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strField** : ce paramètre contient le nom du contexte que vous définissez.
- **vValue** : ce paramètre contient la valeur du contexte que vous définissez.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
' Check that environment variables can be retrieved - this also retrie
ve all
' variables in one shot, improving performances
lRc = DaFirstEnv()

' Get usefull environment variables and store them in context
if lRc <> 0 then
 ' Get environment variables
 strProcArch      = DaGetEnv( "PROCESSOR_ARCHITECTURE" )
 strProcId        = DaGetEnv( "PROCESSOR_IDENTIFIER" )
 strProcLevel     = DaGetEnv( "PROCESSOR_LEVEL" )
 strProcRevision  = DaGetEnv( "PROCESSOR_REVISION" )
 strProcCount     = DaGetEnv( "NUMBER_OF_PROCESSORS" )
 strOs            = DaGetEnv( "OS" )
 strComputer      = DaGetEnv( "COMPUTERNAME" )

 ' Check that computer name is the right one
 if strComputer <> DaContext( "Computer.Name" ) then
   print "Warning: computer name does not look like the right one !"
 end if

 ' Store them in context
 DaSetContext "EnvInfo.ProcArch", strProcArch
 DaSetContext "EnvInfo.ProcId", strProcId
 DaSetContext "EnvInfo.ProcLevel", strProcLevel
 DaSetContext "EnvInfo.ProcRevision", strProcRevision
 DaSetContext "EnvInfo.ProcCount", strProcCount
 DaSetContext "EnvInfo.Os", strOs
 DaSetContext "EnvInfo.Computer", strComputer

 ' Dump
 print DaDumpContext()
end if
```

DaSetOption()

Cette fonction permet de définir des options pour certaines autres fonctions. Les fonctions concernées sont :

- Toutes les fonctions du domaine **Gestion de fichiers**
- Toutes les fonctions du domaine **Exécution distante**

Syntaxe Basic interne

Function DaSetOption(strOption As String, lValue As Long) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strOption** : ce paramètre contient le nom de l'option (entourée par des doubles guillemets) sur laquelle vous souhaitez intervenir. La liste des options disponibles est la suivante :
  - **find.showdir** : lorsque cette option est sélectionnée, la fonction **DaFind**() inclut les répertoires dans la liste retournée.
  - **file.resume** : lorsque cette option est sélectionnée, les fonctions réalisant du transfert de fichiers - **DaUpload**() et **DaDownload**() - tentent de reprendre le transfert à l'endroit où il a été interrompu.
  - **file.mkdir** : lorsque cette option est sélectionnée, les fonctions de **Gestion des fichiers** autorisent la création de répertoires.

- **file.overwrite** : lorsque cette option est sélectionnée, les fonctions de **Gestion des fichiers** autorisent l'écrasement de fichiers de même nom sur la cible.
- **file.recursive** : lorsque cette option est sélectionnée, les fonctions de **Gestion des fichiers** deviennent récursives.
- **file.on_server** : lorsque cette option est sélectionnée, les fonctions **DaMove**() **DaCopy**() et **DaDelete**() opèrent sur le dépôt de fichiers du serveur de déploiement et non sur la cible de déploiement.
- **file.force** : lorsque cette option est sélectionnée, le fichier est forcé. Par exemple, associée à la fonction **DaDelete** cette option permet d'effacer un fichier en lecture seule.
- **exec.synchronous** : lorsque cette option est sélectionnée, l'exécution d'un programme sur la cible de déploiement est synchrone. Toute exécution ultérieure attend la fin de l'exécution précédente.
- **exec.log_output** : lorsque cette option est sélectionnée, les messages envoyés par le programme lors de son exécution sur la cible sont consignés dans un fichier journal.
- **exec.log_error** : lorsque cette option est sélectionnée, les messages résultant d'une erreur d'exécution du programme sur la cible sont consignés dans un fichier journal.
- **exec.force_visibility** : dans le cas de l'exécution d'un programme possédant une interface graphique, le système d'exploitation décide d'afficher ou non cette dernière. Si cette option est sélectionnée, l'affichage est conditionné par la valeur de l'option **exec.visibility**.
- **exec.visibility** : lorsque cette option est sélectionnée, l'exécution d'un programme sur la cible provoque l'affichage sur la cible de l'interface graphique du programme. Cette option est à utiliser conjointement à l'option **exec.force_visibility**.

- **lValue** : ce paramètre peut prendre deux valeurs :
  - 0 : l'option n'est pas sélectionnée
  - 1 : l'option est sélectionnée

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.

- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
' Further invocations of DaUpload, DaDownload, ... will not be recursi
ve
lErr = DaSetOption( "file.recursive", 0 )
```

DaSetReturnValue()

Cette fonction permet de changer l'événement déclenché à la fin de l'exécution d'une activité de type **Script Basic**. Le workflow de déploiement se poursuit en utilisant la transition associée à l'événement défini par le paramètre **strValue**.

Note :

L'événement de sortie de l'activité est invoqué par son nom, et non par son titre.

Syntaxe Basic interne

Function DaSetReturnValue(strValue As String)

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓

Utilisable
Script d'un assistant
Script FINISH.DO d'un assistant

Entrée

- **strValue** : ce paramètre contient le nom de l'événement déclenché à la fin de l'exécution du script Basic.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
dim errCode as long

' call a function that may return error code
errCode = DaFind( "c:\images", "*.gif" )

If errCode <> 0 Then
  DaSetReturnValue( "Error" )
End If
```

Date()

Renvoie la date courante du système.

Syntaxe Basic interne

Function Date() As Date

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

DateAdd()

Cette fonction calcule une nouvelle date en fonction d'une date de départ à laquelle est ajoutée une durée réelle.

Syntaxe Basic interne

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

Champ d'application

Version : 2.51

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **tmStart** : Ce paramètre contient la date à laquelle sera ajoutée une durée.
- **tsDuration** : Ce paramètre contient la durée (exprimée en secondes) à ajouter à la date **tmStart**.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

DateAddLogical()

Cette fonction calcule une nouvelle date en fonction d'une date de départ à laquelle est ajoutée une durée logique (un mois comporte 30 jours).

Syntaxe Basic interne

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

Champ d'application

Version : 2.51

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **tmStart** : Ce paramètre contient la date à laquelle sera ajoutée une durée.
- **tsDuration** : Ce paramètre contient la durée, exprimée en secondes, à ajouter à la date **tmStart**.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

DateDiff()

Cette fonction calcule en secondes la durée écoulée entre deux dates.

Syntaxe Basic interne

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

Champ d'application

Version : 2.51

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **tmEnd** : Ce paramètre contient la date de fin de la période sur laquelle est effectué le calcul.
- **tmStart** : Ce paramètre contient la date de début de la période sur laquelle est effectué le calcul.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

L'exemple suivant calcule la durée écoulée entre le 01/01/98 et le 01/01/99.

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

DateSerial()

Cette fonction renvoie une date formatée en fonction des paramètres **iYear**, **iMonth** et **iDay**.

Syntaxe Basic interne

Function DateSerial(iYear As Long, iMonth As Long, iDay As Long) As Date

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **iYear** : Année. Si sa valeur est comprise entre 0 et 99, ce paramètre décrit les années de 1900 à 1999. Pour toutes les autres années, vous devez utiliser un nombre de quatre chiffres (par exemple 1800).
- **iMonth** : Mois.
- **iDay** : Jour.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.

- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

Chacun de ces paramètres peut prendre pour valeur une expression numérique représentant un nombre de jours, de mois ou d'années. Ainsi l'exemple suivant :

```
DateSerial(1999-10, 3-2, 15-8)
```

renvoie la valeur :

```
1989/1/7
```

Lorsque la valeur d'un paramètre est en dehors de l'intervalle de valeurs généralement admis (c'est à dire 1-31 pour les jours, 1-12 pour les mois, ...), la fonction renvoie une date vide.

DateValue()

Cette fonction renvoie la partie date d'une valeur "Date+Heure"

Syntaxe Basic interne

Function DateValue(tmDate As Date) As Date

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **tmDate** : Date au format "Date+Heure".

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

L'exemple suivant :

```
DateValue ("1999/09/24 15:00:00")
```

renvoie la valeur :

```
1999/09/24
```

DaTrackingDelete()

Cette fonction efface un enregistrement de la table de suivi.

Syntaxe Basic interne

Function DaTrackingDelete(strDomain As String, strCategory As String, strSection As String, strName As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strDomain** : ce paramètre contient le domaine de l'enregistrement de la table de suivi.
- **strCategory** : ce paramètre contient la catégorie de l'enregistrement de la table de suivi.
- **strSection** : ce paramètre contient la section de l'enregistrement de la table de suivi.
- **strName** : ce paramètre contient le nom de l'enregistrement de la table de suivi.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

DaTrackingGet()

Cette fonction récupère la valeur d'un champ de la table de suivi.

Syntaxe Basic interne

Function DaTrackingGet(strDomain As String, strCategory As String, strSection As String, strName As String, strField As String) As String

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strDomain** : ce paramètre contient le domaine de l'enregistrement de la table de suivi.
- **strCategory** : ce paramètre contient la catégorie de l'enregistrement de la table de suivi.
- **strSection** : ce paramètre contient la section de l'enregistrement de la table de suivi.
- **strName** : ce paramètre contient le nom de l'enregistrement de la table de suivi.
- **strField** : ce paramètre contient le nom du champ de la table de suivi.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

DaTrackingSet()

Cette fonction renseigne un champ de la table de suivi. Si l'enregistrement n'existe pas, il est créé.

Syntaxe Basic interne

Function DaTrackingSet(strDomain As String, strCategory As String, strSection As String, strName As String, strField As String, strValue As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	

Utilisable

Script FINISH.DO d'un assistant

Entrée

- **strDomain** : ce paramètre contient le domaine de l'enregistrement de la table de suivi.
- **strCategory** : ce paramètre contient la catégorie de l'enregistrement de la table de suivi.
- **strSection** : ce paramètre contient la section de l'enregistrement de la table de suivi.
- **strName** : ce paramètre contient le nom de l'enregistrement de la table de suivi.
- **strField** : ce paramètre contient le nom du champ de la table de suivi.
- **strValue** : ce paramètre contient la valeur du champ de la table de suivi.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

DaTrackingTest()

Cette fonction retourne la valeur TRUE si un enregistrement existe avec ces propriétés dans la table de suivi (amDaTracking).

Syntaxe Basic interne

Function DaTrackingTest(strDomain As String, strCategory As String, strSection As String, strName As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strDomain** : ce paramètre contient le domaine de l'enregistrement de la table de suivi.
- **strCategory** : ce paramètre contient la catégorie de l'enregistrement de la table de suivi.
- **strSection** : ce paramètre contient la section de l'enregistrement de la table de suivi.
- **strName** : ce paramètre contient le nom de l'enregistrement de la table de suivi.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

DaUpload()

Cette fonction copie un fichier, un répertoire ou un ensemble de fichiers sur la cible de déploiement. Les fichiers source doivent être stockés dans le dépôt de fichiers du serveur de déploiement.

Vous pouvez paramétrer le comportement de cette fonction en fixant certaines options au moyen de la fonction **DaSetOption**(). Vous trouverez une liste complète de ces options dans le descriptif de la fonction **DaSetOption**().

Note :

Cette fonction effectue les mêmes opérations que l'activité **Envoi de fichiers**.

Syntaxe Basic interne

Function DaUpload(strSrcPath As String, strDstPath As String, strSrcNameFilter As String) As Long

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	✓
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **strSrcPath** : ce paramètre contient le chemin des fichiers à copier sur la cible de déploiement. Il s'agit d'un chemin relatif par rapport au chemin absolu du dépôt de fichiers sur le serveur de déploiement. Par exemple si le chemin absolu du dépôt est :

```
c:\files\depot
```

et que vous souhaitez copier un fichier dont le chemin absolu est :

```
c:\files\depot\software\antivirus\update\file.exe
```

alors ce paramètre aura la valeur suivante :

```
software\antivirus\update
```

- **strDstPath** : ce paramètre contient le chemin absolu des fichiers copiés sur la cible de déploiement. Soit, dans le cadre de l'exemple précédent :

```
c:\program files\antivirus\update
```

- **strSrcNameFilter** : caractères joker de type DOS (***** et **?**) utilisés pour filtrer les noms de fichiers.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
' Upload whole 'install' directory
lErr = DaUpload( "install", "c:/tmp/install", "" )

' Impersonate
if lErr = 0 then lErr = DaImpersonate( "user", "password", "domain " )

' Execute setup as user, synchronous, and log errors
if lErr = 0 then
  lDummyErr = DaSetOption( "exec.synchronous", 1 )
  lDummyErr = DaSetOption( "exec.log_output", 1 )
  lDummyErr = DaSetOption( "exec.log_error", 1 )
end if
if lErr = 0 then lErr = DaExec( "c:/tmp/install/setup -i", "c:/tmp/ins
tall" )

' On error, raise 'error' event
if lErr <> 0 then DaSetReturnValue "ErrorEvent"
```

DaWait()

Cette fonction effectue une pause, exprimée en secondes, dans l'exécution d'un workflow de déploiement.

Syntaxe Basic interne

Function DaWait(flSecDelay As Double)

Champ d'application

Version : 1.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	
Script FINISH.DO d'un assistant	

Entrée

- **flSecDelay** : ce paramètre contient le nombre de secondes que dure la pause.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
' Waits half a second, prints a message and waits 10 seconds
DaWait (0.5)
print "Waiting again..."
DaWait (10)
```

Day()

Renvoie le jour contenu dans le paramètre **tmDate**.

Syntaxe Basic interne

Function Day(tmDate As Date) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **tmDate** : Paramètre au format Date+Heure à traiter.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim strDay as String
  strDay=Day(Date())
  RetVal=strDay
```

EnumToComboBox()

Cette fonction réorganise les éléments d'une énumération libre dans un format compatible avec le contrôle de liste des assistants. Vous pouvez ainsi afficher les valeurs d'une énumération libre dans une liste déroulante au sein d'un assistant.

Syntaxe Basic interne

Function EnumToComboBox(strFormat As String) As String

Champ d'application

Version : 4.3.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **strFormat** : Ce paramètre contient la liste des entrées de l'énumération système. Il est préférable que ce paramètre contienne le résultat de l'exécution de la fonction **AmDbGetList**().

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

L'exemple suivant prend les valeurs de l'énumération libre amWOPriority et la réorganise dans un format compatible avec le contrôle de liste des assistants :

```
Dim strValues As String
  strValues = AmDbGetList("SELECT Value FROM amItemListVal WHERE Itemi
zedList.Identifier = 'amWOPriority'", "", ",", "")
  RetVal = EnumToComboBox(strValues)
```

EscapeSeparators()

Préfixe un ou plusieurs caractère(s) défini(s) comme séparateur(s) par un caractère d'échappement.

Syntaxe Basic interne

Function EscapeSeparators(strSource As String, strSeparators As String, strEscChar As String) As String

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strSource** : Chaîne de caractères à traiter.
- **strSeparators** : Liste des séparateurs à préfixer. Si vous souhaitez déclarer plusieurs séparateurs, vous devez les séparer par le caractère utilisé comme caractère d'échappement (indiqué dans le paramètre **strEscChar**.
- **strEscChar** : Caractère d'échappement. Il préfixera tous les séparateurs définis dans **strSeparators**.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim MyStr
  MyStr=EscapeSeparators("toi|moi|toi,moi|toi", "|\,", "\") :'Renvoie
la valeur "toi\|moi\|toi\,moi\|toi"
```

ExeDir()

Cette fonction renvoie le chemin complet de l'exécutable.

Syntaxe Basic interne

Function ExeDir() As String

Champ d'application

Version : 3.60

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim strPath as string
  strPath=ExeDir()
```

Exp()

Renvoie l'exponentielle d'un nombre.

Syntaxe Basic interne

Function Exp(dValue As Double) As Double

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **dValue** : Nombre dont vous souhaitez connaître l'exponentielle.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.

- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

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

ExtractValue()

Extrait d'une chaîne de caractères les valeurs délimitées par un séparateur. La valeur récupérée est alors effacée de la chaîne source. Cette opération tient compte d'un éventuel caractère d'échappement. Si le séparateur n'est pas trouvé dans la chaîne source, l'intégralité de la chaîne est renvoyée et la chaîne source est entièrement effacée.

Syntaxe Basic interne

Function ExtractValue(pstrData As String, strSeparator As String, strEscChar As String) As String

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **pstrData** : Chaîne source à traiter.
- **strSeparator** : Caractère utilisé comme séparateur dans la chaîne source.
- **strEscChar** : Caractère d'échappement. Si ce caractère préfixe le séparateur, ce dernier est ignoré.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim MyStr
  MyStr=ExtractValue("toi,moi", ",", "\") :'Renvoie "toi" et laisse "m
oi" dans la chaîne source
  MyStr=ExtractValue(",toi,moi", ",", "\") :'Renvoie "" et laisse "toi
,moi" dans la chaîne source
  MyStr=ExtractValue("toi", ",", "\") :'Renvoie "toi" et laisse "" dan
s la chaîne source
  MyStr=ExtractValue("toi\,moi", ",", "\") :'Renvoie "toi\,moi" et lai
sse "" dans la chaîne source
  MyStr=ExtractValue("toi\,moi", ",", "") :'Renvoie "toi\" et laisse "
moi" dans la chaîne source
  RetVal=""
```

FileCopy()

Copie un fichier ou un répertoire.

Syntaxe Basic interne

Function FileCopy(strSource As String, strDest As String) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strSource** : Chemin complet du fichier ou du répertoire à copier.
- **strDest** : chemin complet du fichier ou du répertoire de destination.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

FileDateTime()

Renvoie la date et l'heure d'un fichier sous la forme d'un "Long".

Syntaxe Basic interne

Function FileDateTime(strFileName As String) As Date

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strFileName** : Chemin complet du fichier concerné par l'opération.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

FileExists()

Cette fonction teste l'existence d'un fichier. La fonction renvoie les valeurs suivantes :

- 0 : le fichier n'existe pas.
- 1 : le fichier existe.

Syntaxe Basic interne

Function FileExists(strFileName As String) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strFileName** : Ce paramètre contient le chemin complet du fichier dont vous souhaitez tester l'existence.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
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()

Renvoie la taille d'un fichier.

Syntaxe Basic interne

Function FileLen(strFileName As String) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strFileName** : Chemin complet du fichier concerné par l'opération.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction

AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Fix()

Renvoie la partie entière (premier entier supérieur dans le cas d'un nombre négatif) d'un nombre à virgule.

Syntaxe Basic interne

Function Fix(dValue As Double) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **dValue** : Nombre dont vous souhaitez connaître la partie entière.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.

- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim dSeed as Double
  dSeed = (10*Rnd)-5
  RetVal = Fix(dSeed)
```

FormatDate()

Formate une date en fonction de l'expression contenue dans le paramètre **strFormat**.

Syntaxe Basic interne

Function FormatDate(tmFormat As Date, strFormat As String) As String

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **tmFormat** : Date à formater.

- **strFormat** : Expression contenant les instructions de formatage.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

L'exemple de code suivant montre comment formater une date :

```
Dim MyDate
  MyDate="2000/03/14"
  RetVal=FormatDate(MyDate, "dddd d mmmm yyyy") :'Renvoie "Tuesday 14
March 2000"
```

FormatResString()

Cette fonction traite une chaîne source en remplaçant les variables $1, $2, $3, $4 et $5 respectivement par les chaînes contenues dans les paramètres **strParamOne**, **strParamTwo**, **strParamThree**, **strParamFour** et **strParamFive**.

Syntaxe Basic interne

Function FormatResString(strResString As String, strParamOne As String, strParamTwo As String, strParamThree As String, strParamFour As String, strParamFive As String) As String

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strResString** : Chaîne source à traiter.
- **strParamOne** : Chaîne de remplacement de la variable $1.
- **strParamTwo** : Chaîne de remplacement de la variable $2.
- **strParamThree** : Chaîne de remplacement de la variable $3.
- **strParamFour** : Chaîne de remplacement de la variable $4.
- **strParamFive** : Chaîne de remplacement de la variable $5.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

L'exemple suivant :

```
FormatResString("je$1il$2vous$3", "tu", "nous", "ils")
```

renvoie "jetuilnousvousils".

FV()

Cette fonction renvoie le futur montant d'une annuité basée sur des versements constants et périodiques, et sur un taux d'intérêt fixe.

Syntaxe Basic interne

Function FV(dblRate As Double, iNper As Long, dblPmt As Double, dblPV As Double, iType As Long) As Double

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **dblRate** : Ce paramètre indique le taux d'intérêt par échéance. Par exemple pour un prêt à taux d'intérêt annuel de 6%, remboursé suivant une périodicité mensuelle, le taux par échéance est de :

```
0,06/12=0,005 soit 0,5%
```

- **iNper** : Ce paramètre contient le nombre total d'échéances de l'opération financière.
- **dblPmt** : Ce paramètre indique le montant du paiement à effectuer à chaque échéance. Le paiement comporte généralement le principal et les intérêts.
- **dblPV** : Ce paramètre contient la valeur actuelle (ou somme globale) d'une série de paiements devant être effectués dans le futur.

- **iType** : Ce paramètre indique la date d'échéance des paiements. Il peut prendre les valeurs suivantes :
  - **0** si les paiements sont dus à terme échu (c'est à dire en fin de période)
  - **1** si les paiements sont dus à terme à échoir (c'est à dire en début de période)

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Remarques

Note :

- Les paramètres **Rate** et **Nper** doivent être calculés à l'aide d'échéances exprimées dans les mêmes unités.
- Les sommes versées (exprimées notamment par le paramètre **Pmt**) sont représentées par des nombres négatifs. Les sommes reçues sont représentées par des nombres positifs.

GetEnvVar()

Cette fonction renvoie la valeur d'une variable d'environnement. Une valeur vide est renvoyée si la variable d'environnement n'existe pas.

Syntaxe Basic interne

Function GetEnvVar(strVar As String, bExpand As Long) As String

Champ d'application

Version : 3.2.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strVar** : Ce paramètre contient le nom de la variable d'environnement.
- **bExpand** : Ce paramètre booléen est utile quand la variable d'environnement fait référence à une ou plusieurs autres variables d'environnement. Dans ce cas, quand ce paramètre a pour valeur 1 (valeur par défaut), chaque variable référencée est remplacée par sa valeur. Dans le cas contraire, elle ne l'est pas.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
RetVal = getEnvVar("PROMPT")
```

GetListItem()

Renvoie la **lNb**ième portion d'une chaîne délimitée par des séparateurs.

Syntaxe Basic interne

Function GetListItem(strFrom As String, strSep As String, lNb As Long, strEscChar As String) As String

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strFrom** : Chaîne source à traiter.
- **strSep** : Caractère utilisé comme séparateur dans la chaîne source.
- **lNb** : Position de la chaîne à récupérer.
- **strEscChar** : Caractère d'échappement. Si ce caractère préfixe un séparateur, ce dernier sera ignoré.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.

- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

L'exemple suivant :

```
GetListItem("ceci_est_un_test", "_", 2, "%")
```

renvoie "est".

```
GetListItem("ceci%_est_un_test", "_", 2, "%")
```

renvoie "un".

Hex()

Renvoie la valeur hexadécimale d'un nombre.

Syntaxe Basic interne

Function Hex(dValue As Double) As String

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **dValue** : Nombre dont vous souhaitez connaître la valeur hexadécimale.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Hour()

Renvoie la valeur de l'heure contenue dans le paramètre **tmTime**.

Syntaxe Basic interne

Function Hour(tmTime As Date) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓

	Utilisable
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **tmTime** : Paramètre au format Date+Heure à traiter.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim strHour as String
  strHour=Hour(Date())
  RetVal=strHour
```

InStr()

Renvoie la position de la première occurrence d'une chaîne de caractères à l'intérieur d'une autre chaîne de caractères.

Syntaxe Basic interne

Function InStr(iPosition As Long, strSource As String, strPattern As String) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **iPosition** : Position de départ de la recherche. Ce paramètre ne peut être négatif et ne doit pas dépasser 65.535.
- **strSource** : Chaîne dans laquelle s'effectue la recherche.
- **strPattern** : Chaîne de caractères à rechercher.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim strSource as String
Dim strToSearch as String
Dim iPosition
strSource = "Good Bye"
strToSearch = "Bye"
iPosition = Instr(2, strSource, strToSearch)
RetVal=iPosition
```

Int()

Renvoie la partie entière (premier nombre entier inférieur dans le cas d'un nombre négatif) d'un nombre à virgule.

Syntaxe Basic interne

Function Int(dValue As Double) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **dValue** : Nombre dont vous souhaitez connaître la partie entière.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

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

IPMT()

Cette fonction renvoie le montant des intérêts pour une échéance donnée d'une annuité.

Syntaxe Basic interne

Function IPMT(dblRate As Double, iPer As Long, iNper As Long, dblPV As Double, dblFV As Double, iType As Long) As Double

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **dblRate** : Ce paramètre indique le taux d'intérêt par échéance. Par exemple pour un prêt à taux d'intérêt annuel de 6%, remboursé suivant une périodicité mensuelle, le taux par échéance est de :

```
0,06/12=0,005 soit 0,5%
```

- **iPer** : Ce paramètre indique la période concernée par le calcul, comprise entre 1 et la valeur du paramètre **Nper**.
- **iNper** : Ce paramètre contient le nombre total d'échéances de l'opération financière.
- **dblPV** : Ce paramètre contient la valeur actuelle (ou somme globale) d'une série de paiements devant être effectués dans le futur.
- **dblFV** : Ce paramètre contient la valeur future ou le solde que vous souhaitez obtenir après avoir effectué le dernier paiement. En règle générale, et dans le cas d'un remboursement d'un emprunt en particulier, ce paramètre prend la valeur "0". En effet, une fois toutes les échéances remboursées, la valeur de l'emprunt est nulle.
- **iType** : Ce paramètre indique la date d'échéance des paiements. Il peut prendre les valeurs suivantes :
  - **0** si les paiements sont dus à terme échu (c'est à dire en fin de période)
  - **1** si les paiements sont dus à terme à échoir (c'est à dire en début de période)

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Remarques

Note :

- Les paramètres **Rate** et **Nper** doivent être calculés à l'aide d'échéances exprimées dans les mêmes unités.
- Les sommes versées (exprimées notamment par le paramètre **Pmt**) sont représentées par des nombres négatifs. Les sommes reçues sont représentées par des nombres positifs.

IsNumeric()

Cette fonction permet de déterminer si une chaîne de caractères contient une valeur numérique.

Syntaxe Basic interne

Function IsNumeric(strString As String) As Long

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strString** : Ce paramètre contient la chaîne de caractères à analyser.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Kill()

Efface un fichier.

Syntaxe Basic interne

Function Kill(strKilledFile As String) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strKilledFile** : Chemin complet du fichier concerné par l'opération.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

LCase()

Passe tous les caractères d'une chaîne en minuscules.

Syntaxe Basic interne

Function LCase(strString As String) As String

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strString** : Chaîne de caractères à passer en minuscules.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
' This example uses the LTrim and RTrim functions to strip leading ' a
nd trailing spaces, respectively, from a string variable.
' It uses the Trim function alone to strip both types of spaces.
' LCase and UCase are also shown in this example as well as the use
' of nested function calls
```

```
Dim strString as String
Dim strTrimString as String
  strString = "  <-Trim->  " :' Initialize string.
  strTrimString = LTrim(strString) :' strTrimString = "<-Trim->  ".
  strTrimString = LCase(RTrim(strString)) :' strTrimString = "  <-trim
->".
  strTrimString = LTrim(RTrim(strString)) :' strTrimString = "<-Trim->
".
  ' Using the Trim function alone achieves the same result.
  strTrimString = UCase(Trim(strString)) :' strTrimString = "<-TRIM->"
.
  RetVal= "|" & strTrimString & "|"
```

Left()

Renvoie les iNumber premiers caractères d'une chaîne en partant de la gauche.

Syntaxe Basic interne

Function Left(strString As String, iNumber As Long) As String

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strString** : Chaîne de caractères à traiter.
- **iNumber** : Nombre de caractères à renvoyer.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim lWord, strMsg, rWord, iPos :' Declare variables.
  strMsg = "Left() Test."
  iPos = InStr(1, strMsg, " ") :' Find space.
  lWord = Left(strMsg, iPos - 1) :' Get left word.
  rWord = Right(strMsg, Len(strMsg) - iPos) :' Get right word.
  strMsg=rWord+lWord :' And swap them
  RetVal=strMsg
```

LeftPart()

Extrait la portion d'une chaîne de caractères située à gauche du séparateur précisé dans le paramètre **strSep**.

La recherche du séparateur s'effectue de la gauche vers la droite.

La recherche peut tenir compte ou non de la casse en fonction de la valeur du paramètre **bCaseSensitive**.

Syntaxe Basic interne

Function LeftPart(strFrom As String, strSep As String, bCaseSensitive As Long) As String

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strFrom** : Chaîne source à traiter.
- **strSep** : Caractère utilisé comme séparateur dans la chaîne source.
- **bCaseSensitive** : En fonction de la valeur de ce paramètre, la recherche respecte (=1) ou non (=0) la casse.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

Ces exemples illustrent l'utilisation des fonctions **LeftPart**, **LeftPartFromRight**, **RightPart** et **RightPartFromLeft** sur une même chaîne de caractères: "Ceci_est_un_test" :

```
LeftPart("Ceci_est_un_test","_",0)
```

Renvoie la chaîne "Ceci".

```
LeftPartFromRight("Ceci_est_un_test","_",0)
```

Renvoie la chaîne "Ceci_est_un".

```
RightPart("Ceci_est_un_test","_",0)
```

Renvoie la chaîne "test".

```
RightPartFromLeft("Ceci_est_un_test","_",0)
```

Renvoie la chaîne "est_un_test".

LeftPartFromRight()

Extrait la portion d'une chaîne de caractères située à gauche du séparateur précisé dans le paramètre **strSep**.

La recherche du séparateur s'effectue de la droite vers la gauche.

La recherche peut tenir compte ou non de la casse en fonction de la valeur du paramètre **bCaseSensitive**.

Syntaxe Basic interne

Function LeftPartFromRight(strFrom As String, strSep As String, bCaseSensitive As Long) As String

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strFrom** : Chaîne source à traiter.
- **strSep** : Caractère utilisé comme séparateur dans la chaîne source.

- **bCaseSensitive** : En fonction de la valeur de ce paramètre, la recherche respecte (=1) ou non (=0) la casse.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

Ces exemples illustrent l'utilisation des fonctions **LeftPart**, **LeftPartFromRight**, **RightPart** et **RightPartFromLeft** sur une même chaîne de caractères: "Ceci_est_un_test" :

```
LeftPart("Ceci_est_un_test","_",0)
```

Renvoie la chaîne "Ceci".

```
LeftPartFromRight("Ceci_est_un_test","_",0)
```

Renvoie la chaîne "Ceci_est_un".

```
RightPart("Ceci_est_un_test","_",0)
```

Renvoie la chaîne "test".

```
RightPartFromLeft("Ceci_est_un_test","_",0)
```

Renvoie la chaîne "est_un_test".

Len()

Renvoie le nombre de caractères d'une chaîne ou d'un variant.

Syntaxe Basic interne

Function Len(vValue As Variant) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **vValue** : Variant concerné par l'opération.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim strTest as String
Dim iLength as Integer
  strTest = "Peregrine Systems"
  iLength = Len(strTest) :'The value of iLength is 17
  RetVal=iLength
```

LocalToBasicDate()

Cette fonction convertit une date au format chaîne (telle qu'elle est affichée dans le "control panel" de Windows) en une date au format Basic.

Syntaxe Basic interne

Function LocalToBasicDate(strDateLocal As String) As String

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strDateLocal** : Date au format chaîne à convertir.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

LocalToBasicTime()

Cette fonction convertit une heure au format chaîne (telle qu'elle est affichée dans le "control panel" de Windows) en une heure au format Basic.

Syntaxe Basic interne

Function LocalToBasicTime(strTimeLocal As String) As String

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strTimeLocal** : Heure au format chaîne à convertir.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

LocalToBasicTimeStamp()

Cette fonction convertit un ensemble Date+Heure au format chaîne (telle qu'elle est affichée dans le "control panel" de Windows) en un ensemble Date+Heure au format Basic.

Syntaxe Basic interne

Function LocalToBasicTimeStamp(strTSLocal As String) As String

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strTSLocal** : Date+Heure au format chaîne à convertir.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

LocalToUTCDate()

Cette fonction convertit une date au format "Date+Heure" en une date au format UTC (indépendante d'un quelconque fuseau horaire).

Syntaxe Basic interne

Function LocalToUTCDate(tmLocal As Date) As Date

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **tmLocal** : Date au format "Date+Heure".

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Log()

Renvoie le logarithme néperien d'un nombre.

Syntaxe Basic interne

Function Log(dValue As Double) As Double

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **dValue** : Nombre dont vous souhaitez connaître le logarithme.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim dSeed as Double
  dSeed = Int((10*Rnd)-5)
  RetVal = Log(dSeed)
```

LTrim()

Supprime tous les espaces précédant le premier caractère (différent d'un espace) d'une chaîne.

Syntaxe Basic interne

Function LTrim(strString As String) As String

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strString** : Chaîne de caractères à traiter.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
' This example uses the LTrim and RTrim functions to strip leading ' a
nd trailing spaces, respectively, from a string variable.
' It uses the Trim function alone to strip both types of spaces.
' LCase and UCase are also shown in this example as well as the use
' of nested function calls

Dim strString as String
Dim strTrimString as String
  strString = "  <-Trim->  " :' Initialize string.
  strTrimString = LTrim(strString) :' strTrimString = "<-Trim->  ".
  strTrimString = LCase(RTrim(strString)) :' strTrimString = "  <-trim
->".
  strTrimString = LTrim(RTrim(strString)) :' strTrimString = "<-Trim->
".
  ' Using the Trim function alone achieves the same result.
  strTrimString = UCase(Trim(strString)) :' strTrimString = "<-TRIM->"
.
  RetVal= "|" & strTrimString & "|"
```

MakeInvertBool()

Cette fonction renvoie un booléen inversé (0 devient 1, tout autre nombre devient 0).

Syntaxe Basic interne

Function MakeInvertBool(lValue As Long) As Long

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **lValue** : Nombre concerné par l'opération.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim MyValue
  MyValue=MakeInvertBool(0) :'Renvoie la valeur 1
  MyValue=MakeInvertBool(1) :'Renvoie la valeur 0
  MyValue=MakeInvertBool(254) :'Renvoie la valeur 0
```

Mid()

Extrait une chaîne de caractères contenue dans une autre chaîne.

Syntaxe Basic interne

Function Mid(strString As String, iStart As Long, iLen As Long) As String

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strString** : Chaîne de caractères concernée par l'opération.
- **iStart** : Position de départ de la chaîne à extraire à l'intérieur de strString.
- **iLen** : longueur de la chaîne à extraire.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim strTest as String
  strTest="One Two Three" :' Defines the test string
```

```
strTest=Mid(strTest,5,3) :' strTest="Two"
RetVal=strTest
```

Minute()

Renvoie le nombre de minutes contenues l'heure exprimée par le paramètre **tmTime**.

Syntaxe Basic interne

Function Minute(tmTime As Date) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **tmTime** : Paramètre au format Date+Heure à traiter.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction

AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim strMinute
  strMinute=Minute(Date())
  RetVal=strMinute :'Renvoie le nombre de minutes écoulées dans l'heur
e courante par exemple "45" s'il est actuellement 15:45:30
```

MkDir()

Crée un répertoire.

Syntaxe Basic interne

Function MkDir(strMkDirectory As String) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strMkDirectory** : Chemin complet du répertoire à créer.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

Month()

Renvoie le mois contenu dans la date exprimée par le paramètre **tmDate**.

Syntaxe Basic interne

Function Month(tmDate As Date) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **tmDate** : Paramètre au format Date+Heure à traiter.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.

- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim strMonth
  strMonth=Month(Date())
  RetVal=strMonth :'Renvoie le mois courant
```

Name()

Renomme un fichier.

Syntaxe Basic interne

Function Name(strSource As String, strDest As String)

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strSource** : Chemin complet du fichier à renommer.
- **strDest** : Nouveau nom du fichier.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Now()

Renvoie la date et l'heure courantes.

Syntaxe Basic interne

Function Now() As Date

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.

- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

NPER()

Cette fonction renvoie le nombre d'échéances d'une annuité basée sur des versements constants et périodiques, et sur un taux d'intérêt fixe.

Syntaxe Basic interne

Function NPER(dblRate As Double, dblPmt As Double, dblPV As Double, dblFV As Double, iType As Long) As Double

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **dblRate** : Ce paramètre indique le taux d'intérêt par échéance. Par exemple pour un prêt à taux d'intérêt annuel de 6%, remboursé suivant une périodicité mensuelle, le taux par échéance est de :

```
0,06/12=0,005 soit 0,5%
```

- **dblPmt** : Ce paramètre indique le montant du paiement à effectuer à chaque échéance. Le paiement comporte généralement le principal et les intérêts.

- **dblPV** : Ce paramètre contient la valeur actuelle (ou somme globale) d'une série de paiements devant être effectués dans le futur.
- **dblFV** : Ce paramètre contient la valeur future ou le solde que vous souhaitez obtenir après avoir effectué le dernier paiement. En règle générale, et dans le cas d'un remboursement d'un emprunt en particulier, ce paramètre prend la valeur "0". En effet, une fois toutes les échéances remboursées, la valeur de l'emprunt est nulle.
- **iType** : Ce paramètre indique la date d'échéance des paiements. Il peut prendre les valeurs suivantes :
  - **0** si les paiements sont dus à terme échu (c'est à dire en fin de période)
  - **1** si les paiements sont dus à terme à échoir (c'est à dire en début de période)

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Remarques

Note :

Les sommes versées (exprimées notamment par le paramètre **Pmt**) sont représentées par des nombres négatifs. Les sommes reçues sont représentées par des nombres positifs.

Oct()

Renvoie la valeur octale d'un nombre.

Syntaxe Basic interne

Function Oct(dValue As Double) As String

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **dValue** : Nombre dont vous souhaitez connaître la valeur octale.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim dSeed as Double
  dSeed = Int((10*Rnd)-5)
  RetVal = Oct(dSeed)
```

ParseDate()

Cette fonction convertit une date exprimée sous la forme d'une chaîne de caractères en un objet date au sens Basic du terme.

Syntaxe Basic interne

Function ParseDate(strDate As String, strFormat As String, strStep As String) As Date

Champ d'application

Version : 3.60

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strDate** : Date au format chaîne de caractères.
- **strFormat** : Ce paramètre contient le format de la date contenue dans la chaîne de caractères. Les valeurs possibles sont les suivantes :
  - jj/mm/aa
  - jj/mm/aaaa
  - mm/jj/aa
  - mm/jj/aaaa
  - aaaa/mm/jj
  - Date : date exprimée suivant les paramètres de date du poste client.
  - DateInter : date exprimée au format international

- **strStep** : Ce paramètre optionnel contient le séparateur de date utilisé dans la chaîne de caractères. Les séparateurs autorisés sont "\" et "-".

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim dDate as date
  dDate=ParseDate("2001/05/01", "aaaa/mm/jj")
```

ParseDMYDate()

Cette fonction renvoie un objet Date (au sens Basic) à partir d'une date formatée comme suit :

```
jj/mm/aaaa
```

Syntaxe Basic interne

Function ParseDMYDate(strDate As String) As Date

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	

	Utilisable
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strDate** : Date stockée sous la forme d'une chaîne.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

ParseMDYDate()

Cette fonction renvoie un objet Date (au sens Basic) à partir d'une date formatée comme suit :

```
mm/jj/aaaa
```

Syntaxe Basic interne

Function ParseMDYDate(strDate As String) As Date

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strDate** : Date stockée sous la forme d'une chaîne.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

ParseYMDDate()

Cette fonction convertit une chaîne de caractères représentant une date au format aaaa/mm/jj en une variable Basic de type Date.

Syntaxe Basic interne

Function ParseYMDDate(strDate As String) As Date

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strDate** : Date stockée sous la forme d'une chaîne.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

PMT()

Cette fonction renvoie le montant d'une annuité basée sur des versements constants et périodiques, et sur un taux d'intérêt fixe.

Syntaxe Basic interne

Function PMT(dblRate As Double, iNper As Long, dblPV As Double, dblFV As Double, iType As Long) As Double

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **dblRate** : Ce paramètre indique le taux d'intérêt par échéance. Par exemple pour un prêt à taux d'intérêt annuel de 6%, remboursé suivant une périodicité mensuelle, le taux par échéance est de :

```
0,06/12=0,005 soit 0,5%
```

- **iNper** : Ce paramètre contient le nombre total d'échéances de l'opération financière.
- **dblPV** : Ce paramètre contient la valeur actuelle (ou somme globale) d'une série de paiements devant être effectués dans le futur.
- **dblFV** : Ce paramètre contient la valeur future ou le solde que vous souhaitez obtenir après avoir effectué le dernier paiement. En règle générale, et dans le cas d'un remboursement d'un emprunt en particulier, ce paramètre prend la valeur "0". En effet, une fois toutes les échéances remboursées, la valeur de l'emprunt est nulle.
- **iType** : Ce paramètre indique la date d'échéance des paiements. Il peut prendre les valeurs suivantes :
  - **0** si les paiements sont dus à terme échu (c'est à dire en fin de période)
  - **1** si les paiements sont dus à terme à échoir (c'est à dire en début de période)

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction

AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Remarques

Note :

- Les paramètres **Rate** et **Nper** doivent être calculés à l'aide d'échéances exprimées dans les mêmes unités.
- Les sommes versées (exprimées notamment par le paramètre **Pmt**) sont représentées par des nombres négatifs. Les sommes reçues sont représentées par des nombres positifs.

PPMT()

Cette fonction renvoie le montant du remboursement du capital, pour une échéance donnée, d'une annuité basée sur des versements constants et périodiques et sur un taux d'intérêt fixe.

Syntaxe Basic interne

Function PPMT(dblRate As Double, iPer As Long, iNper As Long, dblPV As Double, dblFV As Double, iType As Long) As Double

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **dblRate** : Ce paramètre indique le taux d'intérêt par échéance. Par exemple pour un prêt à taux d'intérêt annuel de 6%, remboursé suivant une périodicité mensuelle, le taux par échéance est de :

```
0,06/12=0,005 soit 0,5%
```

- **iPer** : Ce paramètre indique la période concernée par le calcul, comprise entre 1 et la valeur du paramètre **Nper**.
- **iNper** : Ce paramètre contient le nombre total d'échéances de l'opération financière
- **dblPV** : Ce paramètre contient la valeur actuelle (ou somme globale) d'une série de paiements devant être effectués dans le futur.
- **dblFV** : Ce paramètre contient la valeur future ou le solde que vous souhaitez obtenir après avoir effectué le dernier paiement. En règle générale, et dans le cas d'un remboursement d'un emprunt en particulier, ce paramètre prend la valeur "0". En effet, une fois toutes les échéances remboursées, la valeur de l'emprunt est nulle.
- **iType** : Ce paramètre indique la date d'échéance des paiements. Il peut prendre les valeurs suivantes :
  - **0** si les paiements sont dus à terme échu (c'est à dire en fin de période)
  - **1** si les paiements sont dus à terme à échoir (c'est à dire en début de période)

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Remarques

Note :

- Les paramètres **Rate** et **Nper** doivent être calculés à l'aide d'échéances exprimées dans les mêmes unités.
- Les sommes versées (exprimées notamment par le paramètre **Pmt**) sont représentées par des nombres négatifs. Les sommes reçues sont représentées par des nombres positifs.

PV()

Cette fonction renvoie le montant actuel d'une annuité basée sur des échéances futures constantes et périodiques, et sur un taux d'intérêt fixe.

Syntaxe Basic interne

Function PV(dblRate As Double, iNper As Long, dblPmt As Double, dblFV As Double, iType As Long) As Double

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **dblRate** : Ce paramètre indique le taux d'intérêt par échéance. Par exemple pour un prêt à taux d'intérêt annuel de 6%, remboursé suivant une périodicité mensuelle, le taux par échéance est de :

```
0,06/12=0,005 soit 0,5%
```

- **iNper** : Ce paramètre contient le nombre total d'échéances de l'opération financière.
- **dblPmt** : Ce paramètre indique le montant du paiement à effectuer à chaque échéance. Le paiement comporte généralement le principal et les intérêts.
- **dblFV** : Ce paramètre contient la valeur future ou le solde que vous souhaitez obtenir après avoir effectué le dernier paiement. En règle générale, et dans le cas d'un remboursement d'un emprunt en particulier, ce paramètre prend la valeur "0". En effet, une fois toutes les échéances remboursées, la valeur de l'emprunt est nulle.
- **iType** : Ce paramètre indique la date d'échéance des paiements. Il peut prendre les valeurs suivantes :
  - **0** si les paiements sont dus à terme échu (c'est à dire en fin de période)
  - **1** si les paiements sont dus à terme à échoir (c'est à dire en début de période)

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Remarques

Note :

- Les paramètres **Rate** et **Nper** doivent être calculés à l'aide d'échéances exprimées dans les mêmes unités.
- Les sommes versées (exprimées notamment par le paramètre **Pmt**) sont représentées par des nombres négatifs. Les sommes reçues sont représentées par des nombres positifs.

Randomize()

Initialise le générateur de nombres aléatoires.

Syntaxe Basic interne

Function Randomize(lValue As Long)

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **lValue** : Paramètre optionnel utilisé pour initialiser le générateur de nombres aléatoires de la fonction **Rnd** en lui donnant une nouvelle valeur initiale.

Si ce paramètre est omis, la valeur renvoyée par l'horloge système est utilisée comme valeur initiale.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim MyNumber
  Randomize
  MyNumber= Int((10*Rnd)+1) :'Renvoie une valeur aléatoire comprise en
tre 1 et 10.
  RetVal=MyNumber
```

RATE()

Cette fonction renvoie le taux d'intérêt par échéance pour une annuité.

Syntaxe Basic interne

Function RATE(iNper As Long, dblPmt As Double, dblFV As Double, dblPV As Double, iType As Long, dblGuess As Double) As Double

Champ d'application

Version : 3.00

Utilisable
AssetCenter API

	Utilisable
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **iNper** : Ce paramètre contient le nombre total d'échéances de l'opération financière.
- **dblPmt** : Ce paramètre indique le montant du paiement à effectuer à chaque échéance. Le paiement comporte généralement le principal et les intérêts.
- **dblFV** : Ce paramètre contient la valeur future ou le solde que vous souhaitez obtenir après avoir effectué le dernier paiement. En règle générale, et dans le cas d'un remboursement d'un emprunt en particulier, ce paramètre prend la valeur "0". En effet, une fois toutes les échéances remboursées, la valeur de l'emprunt est nulle.
- **dblPV** : Ce paramètre contient la valeur actuelle (ou somme globale) d'une série de paiements devant être effectués dans le futur.
- **iType** : Ce paramètre indique la date d'échéance des paiements. Il peut prendre les valeurs suivantes :
  - **0** si les paiements sont dus à terme échu (c'est à dire en fin de période)
  - **1** si les paiements sont dus à terme à échoir (c'est à dire en début de période)
- **dblGuess** : Ce paramètre contient la valeur estimée du taux d'intérêt par échéance.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction

AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Remarques

Note :

- Les sommes versées (exprimées notamment par le paramètre **Pmt**) sont représentées par des nombres négatifs. Les sommes reçues sont représentées par des nombres positifs.
- Cette fonction effectue les calculs par itération, en commençant par la valeur attribuée au paramètre **Guess**. Si aucun résultat n'est trouvé au bout de vingt itérations, la fonction échoue.

RemoveRows()

Supprime dans une liste les lignes identifiées par le paramètre **strRowNames**.

Cette fonction est utile lors du traitement des valeurs d'un contrôle de type "ListBox". Les valeurs d'un tel contrôle sont représentées par des chaînes bi-dimensionnelles dont les caractéristiques sont les suivantes :

- Le caractère "|" est utilisé comme séparateur de colonnes.
- Le caractère "," est utilisé comme séparateur de lignes.
- Chaque ligne est terminée par un identifiant unique situé à droite du signe "="

Syntaxe Basic interne

Function RemoveRows(strList As String, strRowNames As String) As String

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strList** : Chaîne source contenant les valeurs d'un contrôle "ListBox" à traiter.
- **strRowNames** : Identifiants des lignes à supprimer. Les identifiants sont séparés par des virgules.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim MyStr
  MyStr=RemoveRows("a1|a2=a0,b1|b2=b0", "a0,c0") :'Renvoie "b1|b2=b0"
  RetVal=MyStr
```

Replace()

Remplace toutes les occurrences du paramètre **strOldPattern** par la valeur du paramètre **strNewPattern** au sein de la chaîne de caractères contenue dans

strData. La recherche de **strOldPattern** peut tenir compte ou non de la casse en fonction de la valeur du paramètre **bCaseSensitive**.

Syntaxe Basic interne

Function Replace(strData As String, strOldPattern As String, strNewPattern As String, bCaseSensitive As Long) As String

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strData** : Chaîne de caractères contenant les occurrences à remplacer.
- **strOldPattern** : Occurrence à recherche dans la chaîne de caractères contenue dans **strData**.
- **strNewPattern** : Texte remplaçant toute occurrence trouvée.
- **bCaseSensitive** : En fonction de la valeur de ce paramètre, la recherche respecte (=1) ou non (=0) la casse.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.

- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim MyStr
MyStr=Replace("toimoitoimoitoi", "toi", "moi",0) :'Renvoie "moimoimoim
oimoi"
MyStr=Replace("toimoitoimoitoi", "Toi", "moi",1) :'Renvoie "toimoitoim
oitoi"
MyStr=Replace("toimoiToimoitoi", "Toi", "moi",1) :'Renvoie "toimoimoim
oitoi"
```

Right()

Renvoie iNumber caractères d'une chaîne en partant de la droite.

Syntaxe Basic interne

Function Right(strString As String, iNumber As Long) As String

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **strString** : Chaîne de caractères à traiter.
- **iNumber** : Nombre de caractères à renvoyer.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim lWord, strMsg, rWord, iPos :' Declare variables.
  strMsg = "Left() Test."
  iPos = InStr(1, strMsg, " ") :' Find space.
  lWord = Left(strMsg, iPos - 1) :' Get left word.
  rWord = Right(strMsg, Len(strMsg) - iPos) :' Get right word.
  strMsg=rWord+lWord :' And swap them
  RetVal=strMsg
```

RightPart()

Extrait la portion d'une chaîne de caractères située à droite du séparateur précisé dans le paramètre **strSep**.

La recherche du séparateur s'effectue de la droite vers la gauche.

La recherche peut tenir compte ou non de la casse en fonction de la valeur du paramètre **bCaseSensitive**.

Syntaxe Basic interne

Function RightPart(strFrom As String, strSep As String, bCaseSensitive As Long) As String

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strFrom** : Chaîne source à traiter.
- **strSep** : Caractère utilisé comme séparateur dans la chaîne source.
- **bCaseSensitive** : En fonction de la valeur de ce paramètre, la recherche respecte (=1) ou non (=0) la casse.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

Ces exemples illustrent l'utilisation des fonctions **LeftPart**, **LeftPartFromRight**, **RightPart** et **RightPartFromLeft** sur une même chaîne de caractères: "Ceci_est_un_test" :

```
LeftPart("Ceci_est_un_test","_",0)
```

Renvoie la chaîne "Ceci".

```
LeftPartFromRight("Ceci_est_un_test","_",0)
```

Renvoie la chaîne "Ceci_est_un".

```
RightPart("Ceci_est_un_test","_",0)
```

Renvoie la chaîne "test".

```
RightPartFromLeft("Ceci_est_un_test","_",0)
```

Renvoie la chaîne "est_un_test".

RightPartFromLeft()

Extrait la portion d'une chaîne de caractères située à droite du séparateur précisé dans le paramètre **strSep**.

La recherche du séparateur s'effectue de la gauche vers la droite.

La recherche peut tenir compte ou non de la casse en fonction de la valeur du paramètre **bCaseSensitive**.

Syntaxe Basic interne

Function RightPartFromLeft(strFrom As String, strSep As String, bCaseSensitive As Long) As String

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	

	Utilisable
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strFrom** : Chaîne source à traiter.
- **strSep** : Caractère utilisé comme séparateur dans la chaîne source.
- **bCaseSensitive** : En fonction de la valeur de ce paramètre, la recherche respecte (=1) ou non (=0) la casse.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

Ces exemples illustrent l'utilisation des fonctions **LeftPart**, **LeftPartFromRight**, **RightPart** et **RightPartFromLeft** sur une même chaîne de caractères: "Ceci_est_un_test" :

```
LeftPart("Ceci_est_un_test","_",0)
```

Renvoie la chaîne "Ceci".

```
LeftPartFromRight("Ceci_est_un_test","_",0)
```

Renvoie la chaîne "Ceci_est_un".

```
RightPart("Ceci_est_un_test","_",0)
```

Renvoie la chaîne "test".

```
RightPartFromLeft("Ceci_est_un_test","_",0)
```

Renvoie la chaîne "est_un_test".

RmDir()

Détruit un répertoire.

Syntaxe Basic interne

Function RmDir(strRmDirectory As String) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strRmDirectory** : Chemin complet du répertoire à détruire.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

Rnd()

Renvoie une valeur contenant un nombre aléatoire.

Syntaxe Basic interne

Function Rnd(dValue As Double) As Double

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **dValue** : Paramètre optionnel dont la valeur définit le mode de génération adopté par la fonction :
  - Inférieur à zéro : Le même nombre est généré à chaque fois.
  - Supérieur à zéro : Nombre aléatoire suivant dans la série.
  - Egal à zéro : Dernier nombre aléatoire généré.
  - Omis : Nombre aléatoire suivant dans la série.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.

- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Remarques

Note :

Avant d'appeler cette fonction, vous devez utiliser la fonction **Randomize**, sans aucun paramètre, pour initialiser le générateur de nombres aléatoires.

Exemple

```
Dim MyNumber
  Randomize
  MyNumber= Int((10*Rnd)+1) :'Renvoie une valeur aléatoire comprise en
tre 1 et 10.
  RetVal=MyNumber
```

RTrim()

Supprime tous les espaces suivant le dernier caractère (qui n'est pas un espace) d'une chaîne.

Syntaxe Basic interne

Function RTrim(strString As String) As String

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	

	Utilisable
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strString** : Chaîne de caractères à traiter.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
' This example uses the LTrim and RTrim functions to strip leading ' a
nd trailing spaces, respectively, from a string variable.
' It uses the Trim function alone to strip both types of spaces.
' LCase and UCase are also shown in this example as well as the use
' of nested function calls

Dim strString as String
Dim strTrimString as String
  strString = "  <-Trim->  " :' Initialize string.
  strTrimString = LTrim(strString) :' strTrimString = "<-Trim->  ".
  strTrimString = LCase(RTrim(strString)) :' strTrimString = "  <-trim
->".
  strTrimString = LTrim(RTrim(strString)) :' strTrimString = "<-Trim->
".
  ' Using the Trim function alone achieves the same result.
  strTrimString = UCase(Trim(strString)) :' strTrimString = "<-TRIM->"
.
  RetVal= "|" & strTrimString & "|"
```

Second()

Renvoie le nombre de secondes contenu dans la l'heure exprimée par le paramètre **tmTime**.

Syntaxe Basic interne

Function Second(tmTime As Date) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **tmTime** : Paramètre au format Date+Heure à traiter.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim strSecond
  strSecond=Second(Date())
  RetVal=strSecond :'Renvoie le nombre de secondes écoulées dans l'heu
re courante par exemple "30" s'il est actuellement 15:45:30
```

SetMaxInst()

Cette fonction permet de fixer le nombre maximal d'instructions qu'un script Basic peut exécuter. Par défaut, le nombre d'instructions est limité à 10000.

Syntaxe API

long SetMaxInst(long lMaxInst);

Syntaxe Basic interne

Function SetMaxInst(lMaxInst As Long) As Long

Champ d'application

Version : 4.3.0

	Utilisable
AssetCenter API	✓
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **lMaxInst** : Ce paramètre contient le nombre d'instructions maximal exécutables par un script.

Sortie

- 0 : La fonction s'est exécutée normalement.
- Non nul : Code d'erreur.

Remarques

Note :

Si vous affectez la valeur "0" au paramètre **lMaxInst**, le nombre d'instructions exécutables par un script est illimité.

SetSubList()

Définit les valeurs d'une sous-liste pour un contrôle "ListBox".

Syntaxe Basic interne

Function SetSubList(strValues As String, strRows As String, strRowFormat As String) As String

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strValues** : Chaîne source contenant les valeurs d'un contrôle "ListBox" à traiter.
- **strRows** : Liste de valeurs à ajouter ou à substituer à celles de la chaîne contenue dans le paramètre **strValues**. Les valeurs sont séparées par le caractère "|". Les lignes traitées sont identifiées par leur identifiant, situé à droite du signe "=". Les lignes inconnues ne sont pas traitées.
- **strRowFormat** : Instructions de formatage de la sous-liste. Les instructions sont séparées par le caractère "|". Ce paramètre possède les caractéristiques suivantes :
  - "1" représente les informations contenues dans la première colonne de la sous-liste.
  - "i-j" peut être utilisé pour définir un ensemble de colonnes.
  - "-" prend en compte toutes les colonnes.
  - Une colonne inconnue ne renvoie aucune valeur.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim MyStr
  MyStr=SetSubList("a1|a2|a3=a0,b1|b2|b3=b0,c1|c2|c3=c0", "A2|A1=a0, B
2|B1=b0", "2|1") :'Renvoie "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") :'Renvoie "a1|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") :'Renvoie "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", "-") :'Renvoie "a1|a2|a3=a0,B1|B2|B3|B4=b0,c1|c2|c3=c0"
  MyStr=SetSubList("A|B|C,D|E|F", "X=*", "2") :'Renvoie "A|X|C,D|X|F"
  RetVal=""
```

Sgn()

Renvoie une valeur indiquant le signe d'un nombre.

Syntaxe Basic interne

Function Sgn(dValue As Double) As Double

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **dValue** : Nombre dont vous souhaitez connaître le signe.

Sortie

La fonction peut renvoyer une des valeurs suivante :

- 1 : Le nombre est supérieur à zéro.
- 0 : Le nombre est égal à zéro.
- -1 : Le nombre est inférieur à zéro.

Exemple

```
Dim dNumber as Double
  dNumber=-256
  RetVal=Sgn(dNumber)
```

Shell()

Lance un programme exécutable.

Syntaxe Basic interne

Function Shell(strExec As String) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **strExec** : Chemin complet de l'exécutable à lancer.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim MyId
  MyId=Shell("C:\WinNT\explorer.exe")
  RetVal=""
```

Sin()

Renvoie le sinus d'un nombre, exprimé en radians.

Syntaxe Basic interne

Function Sin(dValue As Double) As Double

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **dValue** : Nombre dont vous souhaitez connaître le sinus.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim dCalc as Double
  dCalc=Sin(150)
  RetVal=dCalc
```

Space()

Crée une chaînes de caractères comprenant le nombre d'espaces indiqué par le paramètre **iSpace**.

Syntaxe Basic interne

Function Space(iSpace As Long) As String

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **iSpace** : Nombre d'espaces à insérer dans la chaîne.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Remarques

Note :

Cette fonction peut servir à formater des chaînes ou à effacer des données dans des chaînes de longueur fixe.

Exemple

```
Dim MyString
' Renvoie une chaîne de 10 espaces.
  MyString = Space(10)
 :' Insère 10 espaces entre deux chaînes.
  MyString = "Espace" & Space(10) & "inséré"
  RetVal=MyString
```

Sqr()

Renvoie la racine carrée d'un nombre.

Syntaxe Basic interne

Function Sqr(dValue As Double) As Double

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **dValue** : Nombre dont vous souhaitez connaître la racine carrée.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim dCalc as Double
  dCalc=Sqr(81)
  RetVal=dCalc
```

Str()

Convertit un nombre en une chaîne de caractères.

Syntaxe Basic interne

Function Str(strValue As String) As String

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strValue** : nombre à convertir en chaîne.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim dNumber as Double
  dNumber=Cos(150)
  RetVal=Str(dCalc)
```

StrComp()

Effectue la comparaison entre deux chaînes de caractères.

Syntaxe Basic interne

Function StrComp(strString1 As String, strString2 As String, iOptionCompare As Long) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strString1** : Première chaîne de caractères.
- **strString2** : Deuxième chaîne de caractères.
- **iOptionCompare** : type de comparaison. Ce paramètre peut prendre la valeur "0" pour une comparaison binaire, ou "1" pour une comparaison du texte des deux chaînes.

Sortie

- -1 : **strString1** est supérieure à **strString2**.
- 0 : **strString1** est égale à **strString2**.
- 1 : **strString1** est inférieure à **strString2**.

String()

Renvoie une chaîne composée de **iCount** fois le caractère **strString**.

Syntaxe Basic interne

Function String(iCount As Long, strString As String) As String

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **iCount** : Nombre d'occurrences du caractère **strString**.
- **strString** : caractère utilisé pour la composition de la chaîne.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim iCount as Integer
Dim strTest as String
  strTest="T"
```

```
iCount=5
RetVal=String(iCount,strTest)
```

SubList()

Renvoie une sous-liste d'une liste de valeurs contenue dans une chaîne de caractères représentant les valeurs d'un contrôle "ListBox".

Syntaxe Basic interne

Function SubList(strValues As String, strRows As String, strRowFormat As String) As String

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strValues** : Chaîne source contenant les valeurs d'un contrôle "ListBox" à traiter.
- **strRows** : Identifiants des lignes à inclure dans la sous-liste. Les identifiants sont séparés par une virgule. Certains jokers sont acceptés :
  - "*" inclut tous les identifiants dans la sous-liste.
  - Un identifiant inconnu renvoie une valeur vide pour la sous-liste.

- **strRowFormat** : Instructions de formatage de la sous-liste. Les instructions sont séparées par le caractère "|". Ce paramètre possède les caractéristiques suivantes :
  - "1" représente les informations contenues dans la première colonne de la liste dont on extrait une sous-liste.
  - "0" représente l'identifiant de la ligne de la liste dont on extrait une sous-liste.
  - "*" représente les informations contenues dans toutes les colonnes (à l'exception de l'identifiant de la ligne).
  - Une colonne inconnue ne renvoie aucune valeur.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim MyStr
  MyStr=SubList("a1|a2|a3=a0,b1|b2|b3=b0,c1|c2|c3=c0", "a0,b0,a0", "3|
2|3") :'Renvoie "a3|a2|a3,b3|b2|b3,a3|a2|a3"
  MyStr=SubList("a1|a2|a3=a0,b1|b2|b3=b0,c1|c2|c3=c0", "*", "*|0") :'R
envoie "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") :'R
envoie "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") :
'Renvoie "=a0,=b0,=c0"
  MyStr=SubList("a1|a2|a3=a0,b1|b2|b3=b0,c1|c2|c3=c0", "z0", "*=0") :'
Renvoie ""
  MyStr=SubList("a1|a2|a3=a0,b1|b2|b3=b0,c1|c2|c3=c0", "*", "=1") :'Re
nvoie "=a1,=b1,=c1"
  MyStr=SubList("A|B|C,D|E|F", "*", "2=0") :'Renvoie "B,E"
  RetVal=""
```

SysEnumToComboBox()

Cette fonction réorganise les éléments d'une énumération système dans un format compatible avec le contrôle de liste des assistants. Vous pouvez ainsi afficher les valeurs d'une énumération système dans une liste déroulante au sein d'un assistant.

Syntaxe Basic interne

Function SysEnumToComboBox(strFormat As String) As String

Champ d'application

Version : 4.3.0

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	
Action de type "Script"	
Workflow de déploiement	
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strFormat** : Ce paramètre contient la liste des entrées de l'énumération système. Il est préférable que ce paramètre contienne le résultat de l'exécution de la fonction **AmGetFieldFormat**().

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.

- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

L'exemple suivant prend les valeurs de l'énumération système seStatus de la table amWorkOrder et la réorganise dans un format compatible avec le contrôle de liste des assistants :

```
Dim strFormat As String
  strFormat = AmGetFieldFormat(AmGetFieldFromName(AmGetTableFromName("
amWorkOrder"), "seStatus"))
  RetVal = SysEnumToComboBox(strFormat)
```

Tan()

Renvoie la tangente d'un nombre, exprimé en radians.

Syntaxe Basic interne

Function Tan(dValue As Double) As Double

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **dValue** : Nombre dont vous souhaitez connaître la tangente.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim dCalc as Double
  dCalc=Tan(150)
  RetVal=dCalc
```

Time()

Renvoie l'heure courante.

Syntaxe Basic interne

Function Time() As Date

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Timer()

Renvoie le nombre de secondes écoulées depuis 12:00 AM.

Syntaxe Basic interne

Function Timer() As Double

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓

	Utilisable
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

TimeSerial()

Cette fonction renvoie une heure formatée en fonction des paramètres **iHour**, **iMinute** et **iSecond**.

Syntaxe Basic interne

Function TimeSerial(iHour As Long, iMinute As Long, iSecond As Long) As Date

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓

	Utilisable
Script FINISH.DO d'un assistant	✓

Entrée

- **iHour** : Heure.
- **iMinute** : Minutes.
- **iSecond** : Secondes.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

Chacun de ces paramètres peut prendre pour valeur une expression numérique représentant un nombre d'heures, de minutes ou de secondes. Ainsi l'exemple suivant :

```
TimeSerial(12-8, -10, 0)
```

renvoie la valeur :

```
3:50:00
```

Lorsque la valeur d'un paramètre est en dehors de l'intervalle de valeurs généralement admis (c'est à dire 0-59 pour les minutes et les secondes et 0-23 pour les heures), elle est convertie vers le paramètre immédiatement supérieur. Ainsi, si vous entrez "75" comme valeur pour le paramètre **iMinute**, ce dernier sera interprété comme 1 heure et 15 minutes.

L'exemple suivant :

```
TimeSerial (16, 50, 45)
```

renvoie la valeur :

```
16:50:45
```

TimeValue()

Cette fonction renvoie la partie heure d'une valeur "Date+Heure"

Syntaxe Basic interne

Function TimeValue(tmTime As Date) As Date

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **tmTime** : Date au format "Date+Heure".

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

L'exemple suivant :

```
TimeValue ("1999/09/24 15:00:00")
```

renvoie la valeur :

```
15:00:00
```

ToSmart()

Cette fonction reformate un chaîne source en mettant des majuscules au début de chaque mot.

Syntaxe Basic interne

Function ToSmart(strString As String) As String

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strString** : Chaîne source à reformater.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Trim()

Supprime tous les espaces précédant le premier caractère (qui n'est pas un espace) d'une chaîne et tous les espaces suivant le dernier caractère (qui n'est pas un espace) d'une chaîne.

Syntaxe Basic interne

Function Trim(strString As String) As String

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strString** : Chaîne de caractères à traiter.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
' This example uses the LTrim and RTrim functions to strip leading ' a
nd trailing spaces, respectively, from a string variable.
' It uses the Trim function alone to strip both types of spaces.
' LCase and UCase are also shown in this example as well as the use
' of nested function calls

Dim strString as String
Dim strTrimString as String
  strString = "  <-Trim->  " :' Initialize string.
  strTrimString = LTrim(strString) :' strTrimString = "<-Trim->  ".
  strTrimString = LCase(RTrim(strString)) :' strTrimString = "  <-trim
->".
  strTrimString = LTrim(RTrim(strString)) :' strTrimString = "<-Trim->
".
   ' Using the Trim function alone achieves the same result.
  strTrimString = UCase(Trim(strString)) :' strTrimString = "<-TRIM->"
.
  RetVal= "|" & strTrimString & "|"
```

UCase()

Passe tous les caractères d'une chaîne en majuscules.

Syntaxe Basic interne

Function UCase(strString As String) As String

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strString** : Chaîne de caractères à passer en majuscules.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
' This example uses the LTrim and RTrim functions to strip leading ' a
nd trailing spaces, respectively, from a string variable.
' It uses the Trim function alone to strip both types of spaces.
' LCase and UCase are also shown in this example as well as the use
' of nested function calls

Dim strString as String
Dim strTrimString as String
  strString = "  <-Trim->  " :' Initialize string.
  strTrimString = LTrim(strString) :' strTrimString = "<-Trim->  ".
  strTrimString = LCase(RTrim(strString)) :' strTrimString = "  <-trim
```

```
->".
  strTrimString = LTrim(RTrim(strString)) :' strTrimString = "<-Trim->
".
  ' Using the Trim function alone achieves the same result.
  strTrimString = UCase(Trim(strString)) :' strTrimString = "<-TRIM->"
.
  RetVal= "|" & strTrimString & "|"
```

UnEscapeSeparators()

Supprime tous les caractères d'échappement d'une chaîne de caractères.

Syntaxe Basic interne

Function UnEscapeSeparators(strSource As String, strEscChar As String) As String

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strSource** : Chaîne de caractères à traiter.
- **strEscChar** : Caractère d'échappement à supprimer.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim MyStr
  MyStr=UnEscapeSeparators("toi\|moi\|toi\|", "\") :'Renvoie la valeur
 "toi|moi|toi|"
  RetVal=""
```

Union()

Rassemble deux chaînes de caractères délimitées par des séparateurs. Les doublons sont supprimés.

Syntaxe Basic interne

Function Union(strListOne As String, strListTwo As String, strSeparator As String, strEscChar As String) As String

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓

	Utilisable
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strListOne** : Première chaîne de caractères.
- **strListTwo** : Deuxième chaîne de caractères.
- **strSeparator** : Séparateur utilisé pour délimiter les éléments contenus dans les chaînes.
- **strEscChar** : Caractère d'échappement. Si ce caractère préfixe le séparateur, ce dernier est ignoré.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim MyStr
  MyStr=Union("a1|a2,b1|b2", "a1|a3,b1|b2", ",", "\") :'Renvoie la val
eur "a1|a2,b1|b2,a1|a3"
  MyStr=Union("a1|a2,b1|b2", "a1|a3\,b1|b2", ",", "\") :'Renvoie la va
leur "a1|a2,b1|b2,a1|a3\,b1|b2"
  RetVal=""
```

UTCToLocalDate()

Cette fonction convertit une date au format UTC (indépendante d'un quelconque fuseau horaire) en une date au format "Date+Heure".

Syntaxe Basic interne

Function UTCToLocalDate(tmUTC As Date) As Date

Champ d'application

Version : 3.5

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **tmUTC** : Date au format UTC.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Val()

Convertit une chaîne de caractères représentant un nombre en un nombre de type "Double".

Syntaxe Basic interne

Function Val(strString As String) As Double

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **strString** : Chaîne à convertir.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

Exemple

```
Dim strYear
Dim dYear as Double
  strYear=Year(Date())
 dYear=Val(strYear)
  RetVal=dYear :'Renvoie l'année en cours
```

WeekDay()

Renvoie le jour de la semaine contenu dans la date exprimée par le paramètre **tmDate**.

Syntaxe Basic interne

Function WeekDay(tmDate As Date) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **tmDate** : Paramètre au format Date+Heure à traiter.

Sortie

Le nombre retourné correspond à un jour de la semaine, le "1" représentant le dimanche, le "2" le lundi, ..., le "7" le samedi.

Exemple

```
Dim strWeekDay
  strWeekDay=WeekDay(Date())
  RetVal=strWeekDay :'Renvoie le jour de la semaine
```

Year()

Renvoie l'année contenue dans la date exprimée par le paramètre **tmDate**.

Syntaxe Basic interne

Function Year(tmDate As Date) As Long

Champ d'application

Version : 3.00

	Utilisable
AssetCenter API	
Script de configuration d'un champ ou d'un lien	✓
Action de type "Script"	✓
Workflow de déploiement	✓
Script d'un assistant	✓
Script FINISH.DO d'un assistant	✓

Entrée

- **tmDate** : Paramètre au format Date+Heure à traiter.

Sortie

En cas d'erreur, deux cas de figure se présentent :

- Sous AssetCenter, l'exécution du script contenant la fonction est interrompue et un message d'erreur est envoyé à l'utilisateur.
- Dans le cas d'un appel par le biais d'un programme externe, vous devez appeler la fonction AmLastError() [page 369] (et éventuellement la fonction AmLastErrorMsg() [page 370]) pour savoir si une erreur s'est produite (et son message associé).

IV Index

PARTIE

Fonctions disponibles - Liste complète des fonctions

Index

- 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
- AmDaDepAddComputers
- AmDaDepCopyInstance
- AmDaDepCreateInstance
- AmDateAdd
- AmDateAddLogical
- AmDateDiff
- AmDbExecAql
- AmDbGetDate
- AmDbGetDouble
- AmDbGetList
- AmDbGetListEx
- AmDbGetLong
- AmDbGetPk
- AmDbGetString
- AmDbGetStringEx
- AmDeadLine
- AmDecrementLogLevel
- AmDefAssignee
- AmDefaultCurrency
- AmDefEscalationScheme
- AmDefGroup
- AmDeleteLink
- AmDeleteRecord
- AmDisconnectTrace
- AmDuplicateRecord

- AmEndOfNthBusinessDay
- AmEnumValList
- 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
- AmGetSelfFromMainId
- AmGetSourceTable
- AmGetTable
- AmGetTableCount
- AmGetTableDescription
- AmGetTableFromName
- AmGetTableLabel
- AmGetTableName
- AmGetTableRights
- AmGetTableSqlName
- AmGetTargetTable
- AmGetTrace
- AmGetTraceFromHist
- AmGetTypedLinkField
- 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
- AmRestoreRecord
- AmReturnAsset
- AmReturnContract
- AmReturnPortfolioItem
- AmReturnTraining
- AmReturnWorkOrder
- AmRevCryptPassword
- AmRgbColor
- AmRollback
- AmSetFieldDateOnlyValue
- AmSetFieldDateValue
- AmSetFieldDoubleValue
- AmSetFieldLongValue
- AmSetFieldStrValue
- AmSetLinkFeatureValue
- AmSetProperty
- 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
- DaContext
- DaCopy

- DaDbDeleteList
- DaDbGetList
- DaDbSetList
- DaDelete
- DaDownload
- DaDumpContext
- DaExec
- DaExecAction
- DaExecuteActionByName
- DaFileATime
- DaFileCRC
- DaFileCTime
- DaFileLanguage
- DaFileMTime
- DaFileSize
- DaFileType
- DaFileVersion
- DaFind
- DaFindNext
- DaFirstEnv
- DaGetEnv
- DaGetFileInfo
- DaImpersonate
- DaMkDir
- DaMove
- DaNetIpFromName
- DaNetNBTName
- DaNetPing
- DaNetWakeOnLan
- DaNetWinAddressByName
- DaNextEnv
- DaNTFileCopyTo
- DaNTFileCreateDir
- DaNTFileDelete
- DaNTFileDeleteDir
- DaNTFileDirCopyTo
- DaNTFileDirDownload
- DaNTFileDirUpload
- DaNTFileDownload
- DaNTFileUpload
- DaNTRegistryLMAddStringValue
- DaNTRegistryLMCreateKey
- DaNTRegistryLMDeleteKey
- DaNTRegistryLMDeleteValue
- DaNTRegistryLMGetLongValue
- DaNTRegistryLMGetStringValue
- DaNTRegistryLMSetLongValue
- DaNTRegistryLMSetStringValue
- DaNTServiceInstall
- DaNTServiceStart
- DaNTServiceStatus
- DaNTServiceStop
- DaNTServiceUninstall
- DaNTWMIExecMethod
- DaNTWMIExecQuery
- DaNTWMIGetCurrentArrayValue
- DaNTWMIGetCurrentPropertyValue
- DaNTWMIGetInstanceCount
- DaNTWMIGetPropertyValue
- DaNTWMIGetTotalPropertiesValue
- DaNTWMINextItem
- DaNTWMIResetEnumeration
- DaRegCreateKey
- DaRegDeleteKey
- DaRegExec
- DaRegGetValue
- DaRegOutputValue
- DaRegSetValue

- DaRegStrValue
- DaRegVarValue
- DaRename
- DaReturnValue
- DaRmDir
- DaSendMail
- DaSetContext
- DaSetOption
- DaSetReturnValue
- Date
- DateAdd
- DateAddLogical
- DateDiff
- DateSerial
- DateValue
- DaTrackingDelete
- DaTrackingGet
- DaTrackingSet
- DaTrackingTest
- DaUpload
- DaWait
- 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
- RmDir
- Rnd
- 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

Fonctions disponibles - Module 'Achats'

Index

- **AmReturnPortfolioItem**
- **AmReturnTraining**
- **AmReturnWorkOrder**

Fonctions disponibles - Module 'Câblage'

Index

Fonctions disponibles - Actions

Index

Fonctions disponibles - Builtin

Index

- 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
- RmDir
- Rnd
- RTrim
- Second
- SetSubList
- Sgn
- Shell
- Sin
- Space
- Sqr
- Str
- StrComp
- String
- SubList
- Tan
- Time
- Timer
- TimeSerial
- TimeValue
- ToSmart
- Trim
- UCase
- UnEscapeSeparators
- Union
- UTCToLocalDate
- Val
- WeekDay
- Year

Fonctions disponibles - Assistants

Index