

Programmation OpenOffice.org et LibreOffice



Macros OOoBASIC et API

Bernard Marcelly

Laurent Godard

Préface de l'Aful



EYROLLES

Programmation OpenOffice.org et LibreOffice

Suites bureautiques libres, gratuites et multi-plates-formes, OpenOffice.org et LibreOffice comportent plusieurs langages de script dont OOoBasic. Elles sont également munies d'une API permettant de manipuler ses objets. Automatisables et extensibles, elles s'intègrent parfaitement au système d'information de l'entreprise.

Le livre de référence sur la programmation OpenOffice.org et LibreOffice

Écrit par deux contributeurs majeurs de la communauté francophone, ce livre est une référence incontournable sur le puissant langage de macros OOoBASIC et sur l'API d'OpenOffice.org et de LibreOffice. Il explique comment utiliser l'interface utilisateur liée aux macros et aux scripts afin d'automatiser des tâches répétitives, mais aussi comment tirer parti du langage de OOoBasic pour manipuler des documents, créer des boîtes de dialogue et des formulaires, exploiter des bases de données externes ou intégrées, intercepter des événements. Chaque point de l'API est décrit grâce à de nombreux exemples de macros et de routines réutilisables.

Au sommaire

Les scripts • Niveaux de sécurité • Enregistrer et exécuter une macro • Langages de script • Extensions • **OOoBasic** • Gérer les bibliothèques • EDI • Types de données, tableaux, structures • Instructions conditionnelles, sous-programmes • Instructions de traitement • Traitement d'erreurs • **Les documents OpenOffice.org** • Ouvrir, enregistrer, imprimer, fermer • Filtres d'import/export • Propriétés de document • Formats de nombre • Styles • Configuration • **Writer** • Zone de texte, curseurs • Insérer du texte, le mettre en forme • Rechercher, remplacer • Tableaux • Cadres • Sections • Champs de texte • Signets et renvois • Formes et images • Imprimer • **Calc** • Feuille, cellule, zone, ligne, colonne • Lire, modifier une cellule • Déplacer, recopier • Rechercher, remplacer • Trier, filtrer • Fonctions pour Calc • Liens • Diagrammes • Styles • Formes et images • Imprimer • **Draw et Impress** • Page, arrière-plan, couche • Insérer une forme • Propriétés des formes • Types de formes • Points de colle • Grouper des formes • Insérer des images • Objets OLE2 • Spécificités d'Impress par rapport à Draw • Diaporamas • **Boîtes de dialogue** • Construire un dialogue avec l'EDI • Aspects ergonomiques • Les champs de saisie • Événements • Pages multiples • Dialogues multilingues • Dialogues de l'API • **Sources de données** • Le concept Base • Les sources de données • Se connecter • Accès par SQL, par ResultSet, par RowSet • Trier, filtrer • Transactions • Publipostage • **Formulaires** • Les contrôles • Contrôles et base de données • Événements • Macros et document Base • Formulaires et document Base • **Techniques avancées** • Répertoires d'installation • Lire et modifier la configuration • Les fichiers et l'API • Aspects utilisateur • Le Dispatcher • Méthodes propres à MS-Windows • **Comprendre l'API** • L'API depuis OOoBasic • La documentation API • Xray • **Routines utiles** • **Ressources de l'Internet** • Listes de diffusion • Forums • BugZilla.

À qui s'adresse cet ouvrage ?

- Aux utilisateurs d'OpenOffice.org et LibreOffice souhaitant automatiser et étendre leur suite bureautique ;
- À ceux qui migrent vers OpenOffice.org ou LibreOffice et souhaitent recréer des macros existantes ;
- Aux développeurs d'applications d'entreprise et services informatiques ayant à intégrer la suite OpenOffice.org et LibreOffice ;
- Aux étudiants et à tous ceux qui souhaitent s'initier à l'API d'OpenOffice.org et LibreOffice et au langage de macros OOoBASIC.



**Bernard Marcelly
Laurent Godard**

Bernard Marcelly a effectué sa carrière dans la R&D de grands projets logiciels de télécommunications. Parmi ses nombreuses contributions au projet OpenOffice.org, un document d'introduction à l'API, des macros et des outils pour faciliter la programmation d'OOo.

Laurent Godard est impliqué depuis de nombreuses années dans la communauté OpenOffice.org et le développement d'outils complémentaires reposant sur l'API d'OOo Expert reconnu, il a piloté le projet international Extensions d'OpenOffice.org et reste actif dans l'aide et le développement d'outils à destination des suites bureautiques libres OpenOffice.org et LibreOffice

Tous deux membres des communautés OpenOffice.org et LibreOffice, les auteurs ont choisi de reverser leurs droits à des associations aidant les logiciels libres.

Programmation OpenOffice.org et LibreOffice

Macros OOoBASIC et API

Dans la même collection

J. VERNUS. — **3DVIA Studio**. *Le guide officiel. La 3D en temps réel pour tous*.
N°12174, 2011, 212 pages.

R. GOETTER. — **CSS avancées**. *Vers HTML 5 et CSS 3*.
N°12826, 2011, 400 pages.

C. BLAESS. — **Développement système sous Linux**. *Ordonnancement multitâches, gestion mémoire, communications, programmation réseau*.
N°12881, 2011, 1004 pages.

F. DAOUST, D. HAZAËL-MASSIEUX. — **Basculer vers le Web mobile**. *Bonnes pratiques de conception et de développement*.
N°12828, 2011, 300 pages.

P. FICHEUX. — **Linux embarqué**.
N°12452, 2010, 3^e édition, 378 pages.

A. VANNIEUWENHYUZE. — **Programmation Flex 4**. *Applications Internet riches avec Flash ActionScript 3, Spark, MXML et Flash Builder*.
N°12725, 2010, 604 pages.

P. CHATELIER. — **Objective-C pour le développeur avancé**. *Le langage iPhone/iPad et Mac OS X pour les développeurs C++/Java/C#*.
N°12751, 2010, 240 pages.

E. SARRION. — **XHTML/CSS et JavaScript pour le Web mobile**. *Développement iPhone et Android avec et iUI et XUI*.
N°12775, 2010, 274 pages.

J. STARK. — **Applications iPhone avec HTML, CSS et JavaScript**. *Conversion en natifs avec PhoneGap*.
N°12745, 2010, 190 pages.

D. GUIGNARD, E. ROBLES, N. SOREL, J. CHABLE. — **Programmation Android**.
N°12587, 2010, 486 pages.

P. BORGHINO, O. DASINI, A. GADAL. — **Audit et optimisation MySQL 5**. *Bonnes pratiques pour l'administrateur*.
N°12634, 2010, 266 pages.

Chez le même éditeur

S. ROUKINE. — **Améliorer ses taux de conversion web**. *Vers la performance des sites web au-delà du webmarketing*.
N°12858, 2011, 260 pages.

C. HUC. — **Préserver son patrimoine numérique**. *Classer, archiver et sauvegarder ses e-mails, photos et vidéos, contacts, documents administratifs – Guide à l'usage des particuliers et des entrepreneurs individuels*.
N°12789, 2010, 344 pages.

F.-X. BOIS, L. BOIS. — **WordPress 3 pour le blogueur efficace**. *Installation, personnalisation, administration – Référencement et marketing – Monétisation – Usage nomade (iPhone/iPad, Android...)*
N°12829, 2010, 360 pages.

Y. BRAULT. — **Concevoir et déployer ses sites web avec Drupal**. *Drupal 6 – Présentation de Drupal 7*.
N°12780, 2010, 420 pages.

Programmation OpenOffice.org et LibreOffice

Macros OOoBASIC et API

Laurent Godard
Bernard Marcellly

Préface de l'Aful



EYROLLES

ÉDITIONS EYROLLES
61, bld Saint-Germain
75240 Paris Cedex 05
www.editions-eyrolles.com



Le code de la propriété intellectuelle du 1^{er} juillet 1992 interdit en effet expressément la photocopie à usage collectif sans autorisation des ayants droit. Or, cette pratique s'est généralisée notamment dans les établissements d'enseignement, provoquant une baisse brutale des achats de livres, au point que la possibilité même pour les auteurs de créer des œuvres nouvelles et de les faire éditer correctement est aujourd'hui menacée.

En application de la loi du 11 mars 1957, il est interdit de reproduire intégralement ou partiellement le présent ouvrage, sur quelque support que ce soit, sans l'autorisation de l'Éditeur ou du Centre Français d'exploitation du droit de copie, 20, rue des Grands Augustins, 75006 Paris.

© Groupe Eyrolles, 2011, ISBN : 978-2-212-13247-2

Préface

Depuis le début des années 2000, le monde de l'informatique assiste à une révolution attendue depuis des décennies par les professionnels : la standardisation.

Certes les premiers standards ont pris naissance dès les années 1960, sous l'impulsion des industriels tels IBM ou Bull mais, avec l'apparition massive des logiciels propriétaires dans les années 1980, l'informatique est devenue peu à peu une science exacte non écrite.

En effet, les éditeurs, au prétexte de protéger leurs investissements, ont pris l'habitude de livrer à leurs clients des suites logicielles d'une grande richesse fonctionnelle, mais sans fournir la moindre documentation sur le fonctionnement interne des composants, ni sur la manière dont ils communiquent entre eux.

Or en matière de communication, les experts savent parfaitement que l'absence de standards conduit inexorablement à l'émergence de monopoles. Ce qui devait arriver arriva, et la fin des années 1980 a vu se profiler la domination d'un système d'exploitation, d'une suite bureautique, d'un outil de CAO et de bien d'autres logiciels dans des domaines plus spécialisés.

Heureusement, quelques années plus tard, l'explosion d'Internet, une technologie construite entièrement sur des standards ouverts, a fait comprendre qu'il était possible d'imaginer des modèles économiques particulièrement lucratifs basés sur le partage de la connaissance.

C'est ainsi que des organisations telles que l'IETF, le W3C, OASIS et bien d'autres ont collaboré avec les principaux acteurs du marché pour spécifier des standards de communication et des formats d'échange. Beaucoup d'entre eux sont devenus des normes ISO et permettent tous les jours à de nouveaux arrivants de se positionner sur le marché, d'apporter un sang neuf et d'innover.

Parmi ces standards, ODF occupe une place de choix. Validé par OASIS en 2005 puis adopté comme norme par l'ISO en 2006, il s'agit du seul standard bureautique

qui soit totalement spécifié et effectivement adopté nativement par des éditeurs. Grâce à ce standard, une dizaine de suites bureautiques concurrentes, libres ou propriétaires, sont totalement compatibles entre elles. La plus connue d'entre elles, bien sûr, est OpenOffice.org.

Malheureusement l'existence d'un standard, si elle est nécessaire pour lutter contre un monopole, est loin d'être suffisante. Les alternatives ont surtout besoin d'être attractives pour donner envie aux utilisateurs de changer leurs habitudes. OpenOffice.org apporte aux utilisateurs l'ensemble des fonctionnalités qu'ils ont l'habitude de trouver dans une suite bureautique. Mais en tant que challenger, il faut proposer plus. Beaucoup plus. Il faut étendre ses fonctionnalités. C'est ici que ce livre joue un rôle essentiel.

Les API sont précisément un moyen d'étendre, à différents niveaux, les capacités d'un logiciel. Au fil de cet ouvrage, le lecteur se familiarisera avec les mécanismes qui permettent de prendre le contrôle du logiciel. Non seulement il deviendra possible de manipuler les documents par d'autres moyens que ceux proposés par la classique interface utilisateur, mais on pourra également effectuer des opérations impossibles à mettre en œuvre autrement que par programmation.

Pour s'en convaincre il suffit de consulter la liste déjà bien longue des extensions disponibles pour OpenOffice.org qui comprend par exemple une interface pour convertir un fichier PDF en feuilles de dessin, un générateur de rapports, un colorieur syntaxique, un générateur de formulaires, de codes barres, des fonctionnalités de conception assistée, et des centaines d'autres. Certes, nombre de ces extensions dépassent un peu les objectifs de cet ouvrage, mais il y a fort à parier que bon nombre de lecteurs, enthousiasmés par le potentiel et par la grande cohérence des API d'OpenOffice.org et motivés par le succès de leurs premières expérimentations avec l'interface Basic, auront envie de s'aventurer plus loin. Ils pourront alors apprendre à naviguer dans la documentation en ligne qui détaille l'interface Java, très similaire bien que moins abordable, bien sûr, mais ouvrant sur un univers qui ne sera limité que par l'imagination.

Mais il ne faudrait pas s'arrêter en si bon chemin. L'adoption d'OpenOffice.org par le plus grand nombre d'utilisateurs passe par la disponibilité d'un maximum d'extensions, et surtout par leur pertinence par rapport à des besoins réels. Aussi le développeur doit toujours garder à l'esprit que si son travail est utile pour lui ou pour ses collègues, il l'est certainement pour beaucoup d'autres. Et c'est là le principal succès que l'Association Francophone des Utilisateurs de Logiciels libres (AFUL) souhaite à cet ouvrage : parvenir à créer parmi ses lecteurs une dynamique qui leur donne vraiment envie de partager leur travail. Du code, en Basic ou dans n'importe quel autre langage, est avant tout de la connaissance formalisée, qu'il est toujours bon de diffuser.

Selon cette logique, le raisonnement est simple : plus ce livre aura de lecteurs, plus on pourra espérer voir remonter de nouvelles fonctionnalités pour OpenOffice.org sous

forme d'extensions, plus la suite deviendra attrayante, et plus on trouvera de documents au format ODF. La victoire de ce standard s'en trouvera alors largement facilitée.

Cette cohérence entre l'utilisation des standards et le partage de la connaissance est à la base des actions de l'AFUL qui défend depuis plus de douze ans l'utilisation des logiciels libres et l'adoption de standards ouverts.

OpenOffice.org est un logiciel libre qui, grâce au potentiel développé par l'existence de ses API, est susceptible de gagner là où toute l'industrie a finalement renoncé, c'est-à-dire redonner à l'utilisateur les trois atouts qui font la devise de l'AFUL : la Liberté de choisir sa suite bureautique, la Pérennité des documents grâce aux standards ouverts et l'Interopérabilité entre les suites concurrentes qui respectent la norme ODF.

En conclusion, il s'avère que dans la société de l'information, dans laquelle le XXI^e siècle nous a fait entrer, la maîtrise des formats et logiciels, canalisateurs de notre savoir et de sa diffusion n'est pas seulement une affaire de professionnels. L'Histoire n'est pas encore écrite et chacun peut devenir acteur, comme le montre d'ailleurs la naissance récente de LibreOffice, soutenue d'abord par une communauté d'utilisateurs et de développeurs, soucieux de préserver sur le long terme les fondamentaux d'une liberté égalitaire et fraternelle.

*Association Francophone des Utilisateurs de Logiciels libres (AFUL),
par la plume de Philippe Allart (trésorier)
et les lumières de Véronique Fritière et Jean Peyratout*

Table des matières

Avant-propos	1
---------------------------	----------

PREMIÈRE PARTIE

Introduction à la programmation OpenOffice.org	9
---	----------

CHAPITRE 1

Les scripts dans OpenOffice.org	11
--	-----------

De l'automatisation d'OOo à l'application d'entreprise	11
--	----

Des macros pour les utilisateurs d'OpenOffice.org	12
---	----

Des applications à part entière pour l'entreprise	12
---	----

Les macros et la sécurité	13
--	-----------

Les différents niveaux de sécurité	13
--	----

<i>Niveau faible</i>	14
----------------------------	----

<i>Niveau moyen</i>	14
---------------------------	----

<i>Niveau élevé</i>	15
---------------------------	----

<i>Niveau très élevé</i>	16
--------------------------------	----

Les sources de confiance	16
--------------------------------	----

Les signatures numériques	17
---------------------------------	----

L'enregistreur de macros	18
---------------------------------------	-----------

Comment enregistrer une macro ?	18
---------------------------------------	----

Un outil limité	19
-----------------------	----

Les différents langages de script	20
--	-----------

Basic OpenOffice.org	20
----------------------------	----

Particularités des autres langages de script	21
--	----

Java compilé	21
--------------------	----

JavaScript	21
------------------	----

BeanShell	23
-----------------	----

Python	25
--------------	----

Exécuter une macro depuis OpenOffice.org	27
---	-----------

Exécuter une macro depuis le menu Outils	27
--	----

Exécuter une macro depuis un raccourci clavier	29
--	----

Exécuter une macro avec un bouton de barre d'outils	30
Exécuter une macro par une entrée de menu	32
Exécuter une macro depuis une extension	32
Exécuter une macro depuis un document	33
Exécuter une macro sur un événement	33
Exécuter une macro en ligne de commande	34
Sous Windows	34
<i>Lancer une macro Basic résidente</i>	34
<i>Arguments d'appels de la macro</i>	36
<i>Lancer une macro Basic contenue dans un document</i>	36
<i>Lancer un script autre que Basic</i>	37
Sous Linux	39
Les extensions	39
De quoi s'agit-il ?	39
Installer une extension	41
<i>Installation depuis OpenOffice.org</i>	41
<i>Installation depuis la ligne de commande</i>	42
Concevoir une extension	42
<i>Exporter une extension minimale</i>	43
<i>L'outil BasicAddonBuilder</i>	44
<i>L'outil Extension Compiler</i>	45
Conclusion	47

DEUXIÈME PARTIE

Le langage OOoBasic 49

CHAPITRE 2

Introduction au Basic 51

Premier aperçu du langage OpenOffice.org Basic	51
OOoBasic, langage de script d'OpenOffice.org	51
OOoBasic et VBA	53
<i>Le Basic avec un goût de VBA</i>	53
Premiers pas dans l'environnement de développement Basic	55
Anatomie de la fenêtre Macros Basic	55
Souplesse et modularité de l'organisation	58
Gérer les bibliothèques de macros Basic	59
Renommer une bibliothèque	60
Exporter une bibliothèque	60
Importer une bibliothèque	61
Copier une bibliothèque	61

Protéger une bibliothèque par mot de passe	62
Déplacer un module d'une bibliothèque à une autre	62
Créer, supprimer un module ou une boîte de dialogue	63
Gérer les macros Basic	63
La fenêtre d'édition de macros Basic	64
La coloration syntaxique	65
Ma première macro Basic	66
Exécuter une macro depuis l'éditeur Basic	67
Aides à la mise au point	67
<i>Voir le contenu d'une variable</i>	67
<i>Poser des points d'arrêt</i>	68
<i>Vérifier la syntaxe</i>	69
Modules et macros	70
Autres fonctionnalités de l'éditeur	70
Introduction à la programmation Basic	71
Les éléments de base	71
<i>Les éléments non interprétés</i>	72
<i>Longueur des lignes</i>	73
<i>Instructions Basic</i>	73
<i>Noms des variables et des routines</i>	74
<i>Syntaxe élémentaire</i>	74
<i>Exécution Basic</i>	75
Recommandations de programmation	76
<i>Le choix des mots</i>	76
<i>La forme : commentaires et espaces</i>	77
<i>Modularité</i>	77
<i>Simplicité</i>	78
<i>Robustesse</i>	78
<i>Relectures</i>	79
Conclusion	79

CHAPITRE 3

Variables et tableaux de variables 81

Déclarer des variables	81
La déclaration Explicit	82
La déclaration Dim	83
Valeur initiale des variables	84
Portée des variables	85
<i>Variable commune à un module</i>	85
<i>Variable commune à une bibliothèque</i>	86
<i>Variable commune à plusieurs bibliothèques</i>	86

Les chaînes de caractères	87
Concaténation	88
Les variables numériques	89
Les entiers	89
Les nombres réels	90
<i>Les inconvénients du calcul en flottant</i>	91
Les variables monétaires	92
Les variables décimales	93
Les opérateurs numériques	94
<i>Ordre d'évaluation des opérateurs numériques</i>	94
Les booléens	95
Les variables booléennes	95
Les opérateurs booléens	96
Les calculs booléens sur des nombres	98
Les opérateurs de comparaison	98
Calculs booléens	99
Ordre d'évaluation des opérateurs booléens et de comparaison	99
Les variables de date	100
Les objets	101
Le type Variant	102
La valeur Null	103
Comment connaître le type d'un Variant ?	104
Les constantes	106
Les tableaux	107
Les tableaux unidimensionnels (vecteurs)	108
Les tableaux multidimensionnels	109
Redimensionner un tableau	110
Connaître les limites d'index	112
Les affectations entre tableaux	113
Les variables Variant et les tableaux	114
<i>Les tableaux de Variant</i>	114
<i>La fonction Array()</i>	115
<i>Les tableaux de tableaux</i>	115
Structures de données complexes	116
Les collections	116
Les types définis par l'utilisateur	119
Diverses fonctionnalités concernant les variables Basic	122
Les caractères de déclaration de type	122
Les caractères de type pour variables non déclarées	122
DimArray()	123
Option Base	124

Conclusion	124
------------------	-----

CHAPITRE 4

Altérer le cours du programme..... 125

Instructions conditionnelles	125
------------------------------------	-----

If Then Else	125
--------------------	-----

<i>Évaluation d'une expression booléenne</i>	128
--	-----

<i>Éliminer temporairement une partie de code</i>	129
---	-----

Select Case	129
-------------------	-----

IIf	131
-----------	-----

Choose	133
--------------	-----

Switch	134
--------------	-----

Structures de boucle	135
----------------------------	-----

For Next	135
----------------	-----

For Each	137
----------------	-----

Do Loop	138
---------------	-----

While Wend	139
------------------	-----

Branchements inconditionnels	140
------------------------------------	-----

Stop, End	140
-----------------	-----

GoTo	140
------------	-----

On Goto	142
---------------	-----

Les sous-programmes	142
---------------------------	-----

Sub	143
-----------	-----

<i>Appeler un sous-programme</i>	143
--	-----

<i>Les paramètres d'appel de sous-programme</i>	144
---	-----

<i>Paramètres optionnels</i>	145
------------------------------------	-----

<i>Nombre d'arguments transmis au sous-programme</i>	146
--	-----

<i>Transmission par référence ou par valeur</i>	147
---	-----

<i>Transmettre un tableau dans un paramètre</i>	148
---	-----

<i>Portée des variables d'un sous-programme</i>	149
---	-----

<i>Fin prématurée d'un sous-programme Sub</i>	151
---	-----

<i>GoSub : le sous-programme interne</i>	151
--	-----

Fonction : le sous-programme fonction	154
---	-----

<i>Fin prématurée d'un sous-programme Fonction</i>	156
--	-----

Sous-programmes et bibliothèques	156
--	-----

<i>Les conteneurs de bibliothèques Basic</i>	157
--	-----

<i>Charger une bibliothèque Basic du même conteneur</i>	158
---	-----

<i>Charger une bibliothèque Basic d'un autre conteneur</i>	158
--	-----

<i>Charger une bibliothèque de soffice au démarrage</i>	159
---	-----

<i>Appeler une macro d'une autre bibliothèque</i>	159
---	-----

Conclusion	161
------------------	-----

CHAPITRE 5

Les instructions de traitement 163

Les chaînes de caractères	164
Longueur d'une chaîne	164
Comparer deux chaînes de caractères	164
Comparaison à un modèle générique	164
Rechercher une chaîne dans une autre chaîne	166
Le couteau à découper : Mid	167
Supprimer des caractères à gauche et à droite	167
Découper et recoller une chaîne	168
Remplacer partout dans une chaîne de caractères	168
Créer une chaîne de caractères	168
Aligner à gauche ou à droite	169
Les fonctions numériques	169
Signe et valeur absolue	169
Fonctions trigonométriques	169
Autres fonctions mathématiques	169
Nombre aléatoire	170
Les fonctions de date et heure	170
Fonctions de modification de date-heure	171
<i>DateDiff</i>	171
<i>DateAdd</i>	171
<i>DatePart</i>	171
Les fonctions de conversion	172
Les conversions automatiques de type	172
<i>Type Boolean vers type numérique</i>	172
<i>Type Boolean vers type String</i>	172
<i>Type numérique vers type Boolean</i>	172
<i>Type numérique vers type String</i>	172
<i>Type String vers type numérique</i>	173
<i>Type String vers type Boolean</i>	173
<i>L'ambiguïté de l'opérateur +</i>	173
Les conversions explicites	174
<i>Chaîne de caractères vers nombre</i>	174
<i>Type numérique vers chaîne de caractères</i>	174
<i>La fonction Format</i>	175
<i>Conversions vers un type numérique</i>	176
<i>Conversion d'un nombre réel vers un nombre entier</i>	177
Changement de casse	178
Conversions de date et heure	178
<i>Fonctions renvoyant une date-heure interne</i>	178

<i>Fonctions prenant pour argument une date-heure interne</i>	179
Conversion vers une valeur booléenne	179
Test de contenu de variable	179
Interface utilisateur : écran, clavier	180
MsgBox	180
Print	182
InputBox	182
Codage des couleurs	183
Traitement des fichiers	184
Syntaxe des adresses de fichiers	184
Gestion de fichiers	185
<i>Explorer un répertoire</i>	185
<i>Lire et modifier des attributs de fichier ou répertoire</i>	189
Écrire et lire un fichier	189
<i>Fichier texte</i>	189
<i>Fichier texte pour sauver des données</i>	191
<i>Fichier binaire à accès direct</i>	192
<i>Fichier binaire pur</i>	194
<i>Autres instructions pour fichiers ouverts</i>	194
Fonctions système	194
Lancer un programme externe	195
Conclusion	195

CHAPITRE 6

Le traitement des erreurs d'exécution 197

Évitez les erreurs d'inattention	197
Le mécanisme d'interception d'erreur	198
Un exemple typique	198
Conséquence d'une instruction non exécutée	199
Reprise du traitement à un autre endroit	200
L'instruction Resume est indispensable	201
Ignorer les erreurs	201
Informations sur l'erreur	202
Liste des numéros d'erreur	202
Comprendre les messages d'erreur OOoBasic	206
<i>Variable non définie</i>	206
<i>Variable d'objet non définie</i>	206
<i>Utilisation incorrecte d'un objet</i>	208
<i>Valeur de propriété incorrecte</i>	208
<i>Propriété ou méthode introuvable</i>	209
<i>Sous-procédure ou procédure de fonction non définie</i>	209

<i>Une exception s'est produite</i>	209
Portée d'un traitement d'erreur	210
Le traitement d'erreur pour simplifier le codage	212
Déclencher une erreur	212
Conclusion	212

TROISIÈME PARTIE

Manipuler les documents OpenOffice.org 213

CHAPITRE 7

Les documents OpenOffice.org 215

Ce qu'il faut savoir sur l'API	215
Accéder au document	217
Accéder au document en cours	217
Accéder à un autre document	218
<i>Propriétés d'ouverture de document</i>	220
Créer un nouveau document	227
<i>Créer un nouveau document selon modèle</i>	227
Afficher le document en plein écran	228
Sauvegarde des modifications	228
L'état actuel du document	228
Sauver le document	229
<i>Enregistrer une copie</i>	230
Recharger le document	230
Fermer le document	231
Exemples récapitulatifs	231
Imprimer un document	234
L'objet Printer	234
Changer la configuration d'impression	237
Lancer l'impression	238
Énumérer les documents OpenOffice ouverts	239
Les filtres d'import/export	240
Exporter ou importer un document dans un format particulier	241
Exporter en PDF	244
<i>Exemple d'export PDF</i>	248
Importer un document PDF	249
Importer et exporter au format CSV	249
<i>Les paramètres du filtre CSV</i>	250
<i>Créer un document Calc à partir d'un CSV</i>	253
<i>Importer un CSV dans une feuille de Calc</i>	253

Importer et exporter du texte pur	254
Exporter une forme ou une image	255
Liste complète des filtres	260
Les informations du document	261
La notion de Locale dans OpenOffice.org	261
Les propriétés du document	261
<i>DocumentProperties</i>	262
<i>Les champs définis par l'utilisateur</i>	265
Comment le document a-t-il été chargé ?	270
Les formats de nombre	271
Les événements du document	274
Les styles	275
Trouver les styles	275
Récupérer ou supprimer un style du document	277
Récupérer des styles d'un autre document	277
Créer un nouveau style	278
Modifier un style	279
Les propriétés communes des styles	279
Configuration d'affichage d'un document	279
Configuration d'un document	280
Conclusion	281

CHAPITRE 8

Les documents Writer	283
L'objet Text	283
Le curseur d'écriture	284
Déplacer le curseur d'écriture	285
Autres initialisations d'un curseur	287
Lire une zone de texte	288
Insérer du texte	289
Avec la propriété String	289
Avec la méthode insertString	290
<i>Insérer des caractères spéciaux</i>	290
<i>Insérer un saut de page ou de colonne</i>	292
<i>Insérer le texte d'un autre document</i>	293
Supprimer des paragraphes	293
Supprimer une marque de paragraphe	294
Supprimer tout un paragraphe	294
Appliquer un formatage	295
Appliquer un style à un paragraphe	295

Appliquer un style à un ou plusieurs caractères	296
Formatage d'un paragraphe	297
Formatage local des caractères	300
<i>Graisse</i>	300
<i>Italique</i>	301
<i>Soulignement et surlignement</i>	301
<i>Accentuation</i>	302
<i>Relief</i>	303
<i>Changement de casse</i>	303
<i>Exposant et indice</i>	304
<i>Couleurs</i>	304
<i>Rotation de caractères</i>	305
<i>Barrer un caractère</i>	305
<i>Autres propriétés de caractère</i>	306
Supprimer tout formatage de caractère	307
Curseur visible et zone sélectionnée par l'utilisateur	307
Obtenir le curseur visible	307
Zone sélectionnée par l'utilisateur	307
<i>Modifier le contenu de la zone</i>	307
<i>Définir un curseur d'écriture sur la zone sélectionnée</i>	308
<i>Obtenir l'objet texte de la sélection</i>	309
<i>Où se trouve le curseur ?</i>	310
<i>Explorer la zone sélectionnée par l'utilisateur</i>	311
Sélectionner de manière visible une zone de texte	312
Déplacer le curseur visible	313
La page courante, le nombre de pages	314
Définir des positions de tabulation	314
Rechercher – remplacer	316
Le descripteur de recherche	317
Rechercher des attributs particuliers	318
Limiter le champ de la recherche	319
Rechercher pour remplacer	322
Tout remplacer	323
Rechercher des paragraphes	325
Les tableaux	326
Insérer un tableau	326
<i>Insérer plusieurs tableaux</i>	327
<i>Trouver un tableau existant</i>	327
Supprimer un tableau	328
Propriétés de tableau	329
<i>Bordures de tableau</i>	330

<i>Ombre de tableau</i>	332
<i>Largeur du tableau</i>	332
<i>Lignes</i>	334
<i>Colonnes</i>	336
Trouver une cellule ou une zone de cellules	338
<i>Se déplacer dans un tableau</i>	338
<i>Zone sélectionnée par l'utilisateur</i>	341
Les cellules	342
<i>Bordures de cellule</i>	342
<i>Écrire un texte dans la cellule</i>	343
<i>Formules et valeurs numériques</i>	344
Trier un tableau	346
Tableaux irréguliers	350
<i>Scinder et fusionner des cellules</i>	350
<i>Les coordonnées dans un tableau irrégulier</i>	350
Les cadres	352
Insérer un cadre	352
<i>Insérer plusieurs cadres</i>	353
<i>Trouver un cadre</i>	354
Supprimer un cadre	355
Dimensionner un cadre	355
Positionner le cadre	356
<i>Les différents ancrages de cadre</i>	356
<i>Positionnement horizontal</i>	357
<i>Positionnement vertical</i>	358
Adaptation du texte	361
Texte en colonnes	361
Autres propriétés de cadre	362
Écrire du texte dans un cadre	364
Les sections	364
Créer une section, écrire dedans	365
Naviguer dans les sections	366
Propriétés des sections	368
Les styles	369
Style de paragraphe	369
Style de caractère	369
Style de page	370
Les en-têtes et pieds de page	373
Style de cadre	375
Style de numérotation	375
Les champs de texte	375

Le champ de texte FileName	376
Le champ de commentaire, ou note	377
<i>Retrouver un commentaire</i>	379
Les champs dépendants d'un champ maître	380
<i>Variables champ d'utilisateur</i>	381
<i>Champ de base de données</i>	384
Champ masqué	385
Champ de texte conditionnel	387
Les signets et renvois	388
Utiliser un signet existant	388
Insérer un signet	391
Trouver les signets	391
Liens hypertextes	392
<i>Lien externe</i>	392
<i>Lien vers un autre document</i>	392
<i>Lien interne au document</i>	393
<i>Lien vers un endroit dans un autre document</i>	393
Les formes et les images	394
La page de dessin	394
Les formes	395
<i>Insérer une forme à la position du curseur</i>	395
<i>Insérer plusieurs formes</i>	396
<i>Retrouver et supprimer une forme</i>	397
<i>Interaction entre la forme et les autres objets</i>	398
Les images	398
<i>Insérer une image à la position du curseur</i>	398
<i>Insérer plusieurs images</i>	399
<i>Retrouver, supprimer une image</i>	399
Actualiser le document	400
Les informations sur le document	400
Configuration d'affichage du document	401
Options d'impression	403
Conclusion	404

CHAPITRE 9

Les documents Calc 405

Lecture et manipulation de feuilles	405
Les limites d'un document Calc	405
Protéger le document Calc	406
Accéder aux feuilles existantes	406
Ajouter une nouvelle feuille	408

Supprimer une feuille	409
Dupliquer une feuille	410
Déplacer une feuille dans le classeur	410
La feuille visible par l'utilisateur	411
Détecter les événements de feuille	412
Colorer l'onglet d'une feuille	412
Protéger une feuille	412
Cellules et zones de cellules	413
Obtenir une cellule	413
Obtenir une zone de cellules	414
Obtenir une cellule ou zone d'une feuille quelconque	415
Obtenir les coordonnées d'une cellule	416
Obtenir les coordonnées d'une zone de cellules	417
Les zones nommées	418
Les sélections visuelles	420
<i>Sélection faite par l'utilisateur</i>	<i>420</i>
<i>Afficher une zone sélectionnée</i>	<i>421</i>
Zone visible dans la feuille	422
<i>Figurer des lignes ou colonnes</i>	<i>422</i>
<i>Première ligne et première colonne affichées</i>	<i>422</i>
Propriétés globales de la cellule	423
<i>Protéger une cellule</i>	<i>423</i>
<i>Le style de la cellule</i>	<i>424</i>
<i>Aspect général</i>	<i>424</i>
<i>Formater des caractères</i>	<i>425</i>
<i>Format d'affichage</i>	<i>426</i>
<i>Alignement horizontal</i>	<i>426</i>
<i>Alignement vertical</i>	<i>426</i>
<i>Orientation du contenu</i>	<i>426</i>
<i>Ombre de la cellule</i>	<i>427</i>
<i>Bordures et diagonales de la cellule</i>	<i>428</i>
<i>Commentaire (note) de cellule</i>	<i>430</i>
Bordures d'un tableau	432
Cellules fusionnées	435
Lignes et colonnes	436
Les lignes	436
Les colonnes	437
Lire et écrire dans une cellule	438
Les différents contenus d'une cellule	438
Le curseur d'écriture dans la cellule	441
<i>Déplacer le curseur d'écriture</i>	<i>441</i>

<i>Créer un curseur à partir d'un autre curseur</i>	442
Lire un texte dans une cellule	443
Insérer un texte dans une cellule	444
<i>Insérer des caractères spéciaux</i>	445
<i>Formatage de certains caractères</i>	446
Méthodes applicables à une zone de cellules	446
Effacer une zone de cellules	447
Énumérer les cellules d'une zone	448
Trouver les cellules utilisées	449
Trouver les cellules vides	449
Formule matricielle	450
Fonctions mathématiques sur une zone de cellules	451
Lire et écrire les données d'un tableau	452
Recopier uniquement les valeurs et non le format	453
Déplacer ou recopier des cellules avec références	454
<i>Déplacer une zone de cellules</i>	454
<i>Recopier une zone de cellules</i>	455
<i>Recopier une cellule dans une zone</i>	455
<i>Recopier une formule dans une zone</i>	457
Rechercher et remplacer	458
<i>Rechercher toutes les occurrences</i>	459
<i>Rechercher successivement</i>	460
<i>Rechercher pour remplacer</i>	461
<i>Remplacer systématiquement</i>	461
Trier une zone	462
<i>Tri sur plusieurs colonnes</i>	465
Filtrer une zone de cellules	466
Fonctionnalités générales de Calc	472
Attendre le chargement complet du document	472
Activer le calcul des formules	472
Utiliser une fonction de Calc	473
Créer une nouvelle fonction pour Calc	474
<i>Créer une fonction matricielle</i>	476
<i>Limitations des fonctions pour Calc</i>	477
Insérer un lien hypertexte	478
Les liens vers un autre classeur	478
<i>Lier à une feuille d'un autre classeur</i>	479
<i>Lier à une zone d'un autre classeur</i>	480
<i>Lier par DDE</i>	481
<i>Rafraîchir les liens</i>	481
<i>Supprimer les liens</i>	482

Utiliser un Listener	483
<i>Surveiller la modification d'une cellule</i>	483
<i>Laisser l'utilisateur sélectionner une zone de cellules</i>	484
<i>Autres surveillances</i>	486
Contrôles de formulaires liés à Calc	487
Imprimer	487
Zones d'impression	487
Répéter les en-têtes	488
Les diagrammes	489
Obtenir un diagramme existant	489
Les propriétés d'un diagramme	491
Changer la zone de données d'un diagramme	492
Les styles	493
Style de cellule	493
Style de page	493
<i>Comment utiliser les styles de page</i>	495
Les en-têtes et pieds de page	497
Les formes et les images	499
La page de dessin	499
Les formes	500
<i>Insérer une forme</i>	500
<i>Insérer plusieurs formes</i>	501
<i>Retrouver et supprimer une forme</i>	502
Les images	503
<i>Insérer une image</i>	503
<i>Insérer plusieurs images</i>	504
<i>Retrouver et supprimer une image</i>	504
Configuration du document	504
Configuration d'affichage du document	505
Configuration de Calc	506
Conclusion	509

CHAPITRE 10

Les documents Draw et Impress 511

Les pages de dessin	512
Accéder aux pages existantes	512
<i>Les risques des pages non renommées</i>	512
Renommer une page	513
Ajouter une nouvelle page	514
Supprimer une page	515
Dupliquer une page	516

Déplacer une page dans la liste des pages	516
La page visible par l'utilisateur	516
Les arrières-plans	517
Accéder aux arrières-plans existants	517
Renommer un arrière-plan	518
Ajouter un arrière-plan	519
Supprimer un arrière-plan	520
Dupliquer un arrière-plan	521
Déplacer un arrière-plan dans la liste des arrières-plans	521
Affecter un arrière-plan à une page	521
Les couches	522
Accéder aux couches existantes	522
<i>Les couches et l'interface utilisateur</i>	<i>523</i>
Renommer une couche	523
Ajouter une couche	524
Supprimer une couche	525
Dupliquer une couche	525
Déplacer une couche dans la liste des arrières-plans	526
Les propriétés d'une couche	526
Changement du mode d'affichage	526
Les propriétés d'une page de dessin	527
Les commentaires de page de dessin	528
Dessiner une forme	530
Trouver une forme existante	531
Trouver une forme nommée	531
Trouver les formes sélectionnées par l'utilisateur	532
Sélectionner visiblement une forme	533
Lister les formes d'une page	534
Supprimer une forme	534
Propriétés des formes	535
Type de la forme	535
Position et taille de la forme	535
Les liens entre forme et couche	536
Le contour de la forme	537
<i>Les lignes tiretées</i>	<i>538</i>
Le fond et la forme	540
<i>Les styles de remplissage du fond</i>	<i>540</i>
<i>Couleur de fond</i>	<i>540</i>
<i>Un peu de transparence</i>	<i>543</i>
<i>Fond hachuré</i>	<i>545</i>
<i>Fond à motif bitmap</i>	<i>546</i>

L'ombre d'une forme	548
Angle de rotation de la forme	549
Cisaillement de la forme	549
Écrire un texte dans une forme	550
Gestion globale du texte	550
<i>Position du texte dans la forme</i>	<i>551</i>
<i>Texte animé</i>	<i>553</i>
Utilisation d'un curseur d'écriture	554
Les différentes formes	556
Le rectangle et le carré	557
L'ellipse et le cercle	557
Le texte	558
La ligne simple	558
La ligne brisée	559
Le polygone	560
Le poly-polygone	561
Le connecteur	562
L'étiquette	564
La ligne de cote	565
Les formes de Bézier	567
Collages	570
Les points de colle	570
Ajouter un point de colle	573
Supprimer un point de colle	574
Relier deux formes par un connecteur	575
Manipuler plusieurs formes	576
L'ordre Z	576
Grouper des formes	576
Combiner plusieurs formes	577
Connecter plusieurs formes	578
Les images	578
Insérer une image	579
<i>Insérer plusieurs images</i>	<i>579</i>
<i>Retrouver, supprimer une image</i>	<i>580</i>
Propriétés des images	580
Les autres objets insérés dans une page de dessin	583
Les objets OLE2	583
<i>Les formules, ou équations</i>	<i>584</i>
Les objets vidéo et son	585
Les styles	585
Imprimer	586

Configuration d'impression	586
Configuration du document	587
Spécificités d'Impress par rapport à Draw	588
Couches de dessin	588
Page de diapo	589
La configuration d'exécution des diaporamas	589
Lancer un diaporama	590
Intervenir pendant un diaporama	591
La page de notes	593
La page prospectus	594
Les styles Impress	594
Configuration de document	594
Configuration d'impression	595
Conclusion	595

QUATRIÈME PARTIE

Construire des applications avec OpenOffice.org ... 597

CHAPITRE 11

Les boîtes de dialogue 599

Construire une boîte de dialogue avec l'EDI	600
L'onglet de dialogue	600
La fenêtre des propriétés	601
<i>L'onglet Général</i>	601
<i>L'onglet Événements</i>	602
Votre première boîte de dialogue	605
L'étiquette	605
Le Bouton	606
<i>Le bouton Standard</i>	606
<i>Le bouton OK</i>	606
<i>Le bouton Annuler</i>	607
<i>Le bouton Aide</i>	607
Ajuster les éléments du dialogue	607
Tester le dialogue	607
Exécuter le dialogue	608
Les principaux champs de saisie	612
La zone de texte	612
Le champ numérique	614
La zone de liste	615
<i>La zone de liste simple</i>	615
<i>La zone de liste à sélection multiple</i>	617

<i>La zone de liste combinée</i>	618
Les cases à cocher	619
Les cases de choix 1 parmi N	621
L'aspect visuel des dialogues	623
Un dialogue clair	623
La notion de focus	624
<i>Le bouton par défaut</i>	625
<i>Lettre accélératrice</i>	625
Les éléments visuels	626
<i>La zone de groupe</i>	626
<i>Les lignes horizontale et verticale</i>	626
Les champs de saisie spécialisés	627
Le champ de date	627
Le champ horaire	629
Le champ monétaire	630
Le champ masqué	631
Le champ formaté	633
<i>Format Date</i>	633
<i>Format Pourcentage</i>	635
<i>Format Monétaire</i>	636
<i>Format Booléen</i>	636
<i>Format Fractionnaire</i>	637
La sélection de fichiers	637
Le contrôle Image ou contrôle Picto	638
Les barres de défilement	640
La barre de progression	641
Principes à connaître pour des dialogues élaborés	643
Le contrôle et son modèle	643
Relations entre contrôles et dialogue	644
Les autres propriétés de contrôle et de dialogue	645
<i>Propriété Activé</i>	645
<i>Propriété Visible</i>	645
<i>Propriété Page (pas)</i>	645
<i>Propriété Complément d'information</i>	645
Remettre à « vide » un champ numérique	645
Gérer dynamiquement les contrôles de dialogue	646
Déclencher une routine dans un dialogue	646
Gestionnaire d'événements commun à plusieurs contrôles	648
Modifier le contenu d'une zone de liste	650
<i>Modifier dynamiquement les sélections multiples</i>	652
Modifier le contenu d'une zone de liste combinée	654

Liste de choix avec éléments à images	655
Les principaux événements	656
Lors du déclenchement	657
Statut modifié	658
Changement de focus	658
Touche du clavier	659
Souris	661
Texte Modifié	662
Ajouter des contrôles par programme	662
Gestion du panneau de dialogue	664
Position et dimensions d'un dialogue	664
Dialogues emboîtés	666
Dialogues à pages multiples	667
Les dialogues multilingues	668
<i>Définir un dialogue multilingue</i>	668
<i>Définir des messages multilingues</i>	671
Les services de dialogues de l'API	673
Sélectionner un fichier existant	674
Sélectionner plusieurs fichiers existants	676
Enregistrer un fichier	677
Les différents dialogues de FilePicker	678
Choisir un répertoire	679
Conclusion	680

CHAPITRE 12

Les sources de données..... 683

Le concept Base	684
Limitations de la base de donnée embarquée	684
Les bases de données du Zip téléchargeable	685
Les sources de données	686
Lister les sources de données	686
Propriétés d'une source de données	688
Créer et supprimer une source de données	689
<i>L'URL d'une source de données</i>	689
<i>Créer une source de données</i>	690
<i>Supprimer une source de données</i>	694
Se connecter à une source de données	695
<i>Se connecter à une base non enregistrée</i>	698
Propriétés d'une connexion	700
Les tables	702
<i>Les colonnes</i>	704

<i>Créer ou supprimer une table</i>	706
Les requêtes pré-enregistrées	709
<i>Ajouter ou supprimer une requête</i>	710
<i>Modifier une requête</i>	711
Accéder aux données avec le langage SQL	712
Quelques règles syntaxiques du SQL	712
Exploiter les résultats d'une requête	714
<i>Autres exemples de requêtes d'interrogation</i>	721
Insérer un enregistrement dans une table	721
Modifier un enregistrement d'une table	723
Supprimer des enregistrements	723
Les requêtes paramétrées	724
Accéder aux données avec un ResultSet	725
Insérer un enregistrement dans une table	725
Modifier un enregistrement d'une table	726
Supprimer des enregistrements	727
Les différentes capacités des ResultSet	727
Accéder aux données avec un RowSet	728
Se connecter à la base de données	728
<i>Le RowSet établit la connexion</i>	729
<i>Se connecter à une base non enregistrée</i>	731
<i>Utiliser une connexion existante</i>	731
Explorer les résultats d'une requête	732
Insérer un enregistrement dans une table	734
Modifier ou supprimer un enregistrement d'une table	735
Tri et filtre supplémentaires	735
Utiliser les requêtes pré-enregistrées	737
Les événements du RowSet	737
Les transactions	740
Ouverture de la base	740
Gérer les transactions	742
Utilisation dans le contexte bureautique	744
Le publipostage	745
<i>Sortie imprimante</i>	747
<i>Sortie courriel</i>	748
Calc et les bases de données	749
<i>Importer des données dans Calc</i>	750
<i>Une requête dans une cellule</i>	752
<i>Importer des données depuis une cellule</i>	753
Importer les requêtes MS-Access	754
Conclusion	756

CHAPITRE 13

Les formulaires 757

Accéder aux contrôles d'un formulaire	758
Les sous-formulaires	762
Fonctionnalités de base des contrôles	762
Le bouton	762
<i>Le bouton picto ou bouton-image</i>	763
Les zones de liste non liées à une base de données	763
<i>La zone de liste simple</i>	763
<i>La zone de liste combinée</i>	764
<i>Autres possibilités des zones de liste</i>	765
La case à cocher	765
Le choix 1 parmi N	765
Les champs de saisie	766
<i>La zone de texte avec formatage</i>	767
Le compteur	768
Le contrôle Image ou contrôle Picto	768
Autres contrôles	769
Principes communs aux contrôles	769
<i>Imposer le focus sur un contrôle</i>	769
<i>Remettre le focus sur le document</i>	769
<i>Rendre un contrôle invisible</i>	770
<i>Gérer les événements d'un contrôle</i>	770
<i>Gestionnaire d'événements commun à plusieurs contrôles</i>	770
<i>Formulaire intelligent</i>	771
Les contrôles de formulaire dans Calc	771
Mémoriser une information cachée dans un contrôle	773
Contrôles et base de données	773
L'objet formulaire	774
La barre de navigation	774
Le contrôle Table	774
<i>Connaitre la sélection dans le contrôle Table</i>	776
<i>Déplacer la position courante dans le contrôle Table</i>	777
Modifier le contenu d'un champ du formulaire	778
Zone de liste et base de données	779
Les événements des formulaires	780
Ajouter des contrôles par programme	783
Les documents intégrés dans un document Base	788
Les macros dans un document Base	788
Ouvrir un formulaire d'un document Base	789

<i>Ouvrir depuis un autre document que Base</i>	789
<i>Ouvrir un formulaire intégré depuis un autre formulaire</i>	790
<i>Fermer un formulaire par un bouton</i>	792
<i>Tout fermer par un bouton</i>	793
<i>Obtenir les contrôles d'un formulaire intégré</i>	793
<i>Ouvrir un formulaire en mode conception</i>	794
Ouvrir un rapport d'un document Base	795
Conclusion	795

CHAPITRE 14

Techniques avancées pour le poste de travail 797

Les répertoires d'installation	797
La structure en couches d'OpenOffice.org 3	798
Le service PathSettings	798
Le service PathSubstitution	799
Répertoires des scripts	801
Répertoire d'installation d'une extension	801
Modifier la configuration d'OpenOffice.org	802
Comment est stockée la base de registre OpenOffice.org ?	803
<i>Configuration utilisateur</i>	803
<i>Configuration de l'installation</i>	804
La hiérarchie de la base de Registre	804
<i>Les données terminales</i>	805
<i>Explorer la base de registre</i>	806
<i>Lire une valeur de la base de registre</i>	806
<i>Modifier une valeur de la base de registre</i>	807
Modifier une configuration par fichier	807
Gérer les fichiers depuis l'API	807
Écrire un fichier binaire	807
Lire un fichier binaire	809
Lecture-écriture d'un fichier binaire	810
Écrire un fichier texte encodé	810
Lire un fichier texte encodé	812
Création et décompression d'un fichier Zip	813
Lancer l'application associée à un document	815
La palette des couleurs	816
Penser à l'utilisateur	818
Geler l'affichage du document	818
<i>Gels spécifiques à Calc</i>	819
Indiquer l'avancement du travail	819
Empêcher les actions de l'utilisateur	820

Les fenêtres dans OpenOffice.org	821
<i>Changer la position ou taille de la fenêtre</i>	822
<i>Document affiché avec plusieurs fenêtres</i>	823
<i>Ordre des fenêtres OpenOffice.org</i>	823
<i>Fenêtre visible ou invisible</i>	824
Afficher ou masquer une barre d'outils	824
Une fonction Format plus puissante	825
Différences avec la fonction Basic Format	827
Fermer complètement OpenOffice	827
Utiliser le Dispatcher	828
Copier-coller	830
Lancer une macro d'un autre document	831
Traitements spécifiques à MS-Windows	832
Accéder à la base de registres de MS-Windows	832
Utiliser l'API de MS-Windows	833
Manipuler les objets COM	834
Piloter OpenOffice.org par COM Automation	836
Envoyer un document par courrier électronique	838
Utiliser un serveur web	839
Intercepter un événement : le Listener	840
Modifier des macros par programmation	845
L'écriture dynamique de macros	845
Modifier les macros d'un autre document	846
Appeler un script écrit dans un autre langage	848
Conclusion	850

ANNEXES

Outils et ressources 851

ANNEXE A

Comprendre l'API d'OpenOffice.org 853

Qu'est-ce que l'API ?	853
Règles typographiques des noms dans l'API	855
L'API réelle et l'API selon OOoBasic	855
Les fonctions Basic dédiées à l'API	857
<i>GetProcessServiceManager</i>	858
<i>CreateUnoService</i>	858
<i>StarDesktop</i>	859
<i>ThisComponent</i>	859
<i>ThisDatabaseDocument</i>	859

<i>GetDefaultContext</i>	859
<i>CreateUnoStruct</i>	860
<i>CreateObject</i>	860
<i>CreateUnoValue</i>	861
<i>CreateUnoListener</i>	861
<i>CreateUnoDialog</i>	862
<i>IsUnoStruct</i>	862
<i>EqualUnoObjects</i>	862
<i>HasUnoInterfaces</i>	862
<i>ConvertToURL, ConvertFromURL</i>	862
La documentation de l'API et le Software Development Kit	863
Comment s'y retrouver ?	864
Xray	866
Object Inspector	869
MRI	869
Conclusion	869

ANNEXE B

Routines utilitaires..... 871

Tableaux de propriétés	871
Créer un tableau de propriétés	871
Accéder à une propriété par son nom	872
Coordonnées de cellules	873
Rechercher un objet par son nom	875
Redimensionner une image	876
Traduire un nom de style	878
Adresses URL de fichiers et répertoires	879
Trier un tableau de données en Basic	879
Rappel des routines utilitaires décrites dans le livre	880
Dialogue	880
Base de données, formulaires	880
Création et décompression d'un fichier Zip	881
Envoyer une commande au Dispatcher	881
Conclusion	881

ANNEXE C

Ressources disponibles sur l'Internet..... 883

Quelques conseils	883
Macros et extensions disponibles sur l'Internet	883
Règles de bonne conduite	884

Utiliser une liste de diffusion	884
Ressources en français	885
Le site fr.OpenOffice.org	885
Le forum de la communauté francophone	886
Les listes de diffusion francophones	886
Autres sites	887
Ressources en langue anglaise	887
Le forum de la communauté anglophone	887
Le forum OOoForum	887
Les listes de diffusion anglophones	887
Autres sites	888
<i>Sites de développement OpenOffice.org</i>	888
<i>L'entrepôt des Extensions</i>	889
Les sites pour télécharger OpenOffice.org	889
Où se trouvent les langpacks ?	890
BugZilla	890
Rechercher un rapport dans BugZilla	890
Rédiger un rapport	891
Partager la connaissance	892
Conclusion	892
 Index	 893

Avant-propos

Rappelons en guise de préambule qu'OpenOffice.org, suite bureautique libre et gratuite, est constituée des modules habituels de traitement de texte, de tableur, de présentation, ainsi que de modules de dessin et d'édition de formules mathématiques. Tournant aussi bien sous Windows et Linux que sous Mac OS X, elle peut être utilisée en lieu et place de la suite Microsoft Office.

À NOTER OpenOffice.org et ses dérivés

OpenOffice.org est une base à partir de laquelle diverses variantes ont été dérivées. Sun Microsystems avait commercialisé StarOffice. Depuis, Sun a été absorbé par Oracle, qui commercialise Oracle Open Office, version non libre dont le nom crée une ambiguïté... Parmi les versions libres et gratuites, certaines sont adaptées à des distributions Linux, et NeoOffice est plus orienté Mac. Plus récemment, LibreOffice, qui se veut plus indépendant et plus évolutif, est porté sur tous les systèmes d'exploitation. Soulignons que LibreOffice est loin de constituer un simple produit dérivé. Bien au contraire, il s'agit là d'une séparation majeure sous l'égide de l'OpenDocument Foundation, et nombreux sont les contributeurs qui prennent position pour cette nouvelle initiative. Nous vous recommandons donc de suivre son évolution. Tous ces produits dérivés d'OpenOffice.org témoignent d'une dynamique d'évolution fonctionnelle tout en assurant la pérennité des documents bureautiques produits manuellement ou par programmation. Le contenu de ce livre s'applique généralement à toutes ces variantes et nous en resterons à la dénomination générique OpenOffice.org.

À qui s'adresse ce livre ?

Vous êtes un utilisateur de la suite OpenOffice.org 2 ou 3 (ou un dérivé comme LibreOffice) et vous connaissez bien ses nombreuses possibilités. Cependant, dans certains cas, vous souhaitez simplifier des manipulations répétitives. Le langage de macros Basic d'OpenOffice.org, intégré dans la suite, peut répondre à votre besoin. Il est conçu pour être simple d'emploi, tout en étant puissant.

Avec ce livre, vous apprendrez à programmer la suite OpenOffice.org, par exemple pour ajouter une fonctionnalité personnelle déclenchée par un bouton sur une barre d'outils ou par un raccourci. Vous pourrez même automatiser des traitements, par exemple effectuer des modifications dans toute une série de documents Writer, ou lire une base de données pour en extraire des informations et les insérer dans un document Writer.

Ce livre est-il accessible à un débutant en programmation ? Les connaissances de base de la programmation sont exposées dans les premiers chapitres. Nous avons aussi inclus des conseils et des bonnes pratiques, qui vous éviteront bien des déboires. À chaque étape, nous avons choisi des exemples volontairement simples, chacun focalisé sur le point à expliquer. Évidemment, le chemin sera plus ardu et plus long si vous en êtes à vos tout débuts : avancez très progressivement, en écrivant de nombreux petits programmes pour vous approprier les concepts de base. Ne craignez pas les erreurs, elles sont source de connaissances. Progressivement, vos programmes s'enrichiront et deviendront toujours plus utiles, à votre grande satisfaction.

Si vous avez l'expérience de la programmation, la première partie vous semblera facile. Méfiez-vous cependant des analogies avec d'autres langages, car chacun a ses particularités. Si vous connaissez la programmation orientée objet, vous comprendrez plus facilement les principes de l'API OpenOffice.org, qui sera utilisée à partir de la troisième partie ; mais ce n'est pas indispensable.

Si vous êtes dans un service informatique chargé d'automatiser des processus utilisant la suite OpenOffice.org ou StarOffice, ou de migrer des applications écrites pour la suite MS-Office, cet ouvrage vous économisera de très nombreuses heures de recherche et de tâtonnements et accélérera la phase d'apprentissage. Comme la mémoire humaine a ses limites, vous souhaiterez garder ce livre à portée de main. Il est cependant probable que vous rencontrerez des besoins non décrits ici, mais la base de connaissances acquises vous facilitera l'étude de l'API OpenOffice.org. Même si votre projet n'emploie pas Basic, celui-ci peut vous aider à trouver plus rapidement comment utiliser telle ou telle fonctionnalité et en déduire le codage équivalent dans votre langage.

PRÉCAUTION

Pour tirer parti de ce livre, il est recommandé de bien connaître les possibilités qu'OpenOffice.org offre au niveau de son interface utilisateur, dans les domaines sur lesquels vous souhaitez intervenir par macro. En effet, autant éviter un développement si quelques manipulations résolvent votre problème ou le simplifient. Savez-vous bien utiliser les styles de paragraphe, de caractère, de page ? les modèles de document ? la recherche générique ? le copier-coller avec ou sans formatage ? l'utilisation des bases de données ? Savez-vous ce qu'est un signet ? N'hésitez pas à lire ces excellents livres donnant toutes les astuces pour être productif sous OpenOffice.org.

📖 *OpenOffice.org 2.2 efficace* de Sophie Gautier, Christian Hardy, Frédéric Labbé, Michel Pinquier, éditions Eyrolles 2007.

📖 *OpenOffice.org 3.2 efficace* de Sophie Gautier, Gilles Bignebat, Christian Hardy et Michel Pinquier, éditions Eyrolles 2010.

Contenu de l'ouvrage

Vous trouverez une description complète et précise du Basic OpenOffice.org, rédigée pour être compréhensible pour un débutant en programmation, tout en apportant des informations indispensables au programmeur expérimenté.

Vous apprendrez comment utiliser facilement l'interface de programmation d'application (API) pour lire, écrire, modifier les documents OpenOffice.org, accéder aux bases de données, et dialoguer avec l'utilisateur. L'API d'OpenOffice.org est extrêmement riche et parfois complexe. Il nous a fallu privilégier les sujets les plus courants et les vérifier chacun par des macros. Si vous ne trouvez pas la réponse à une question dans cet ouvrage, c'est peut-être qu'il s'agit d'un cas rare et particulièrement difficile à réaliser.

L'API est une interface indépendante du langage de programmation. À cet égard, les descriptions des fonctionnalités sont valides pour divers environnements : les langages de script intégrés à OpenOffice.org (Basic, Python, JavaScript, BeanShell, cités au chapitre 1), le pilotage par un langage externe via COM (abordé au chapitre 14), voire même dans des composants développés en Java ou C++. Un développeur d'applications pourra facilement transposer dans un autre langage les exemples Basic donnés dans cet ouvrage. L'annexe A décrit les principes de l'API, son utilisation par le Basic OpenOffice.org, et comment aller plus loin avec Xray et la documentation de l'API et les outils d'introspection (Xray et similaires).

Notre souci a été d'être clair et progressif, sans trop entrer dans des considérations théoriques. Nous avons pour cela créé des centaines d'exemples de macros Basic, avec des documents spécialement configurés pour chaque essai. Tous les exemples ont été testés, et pour la plupart retestés, sur les versions récentes. Ils sont mis à votre disposition en téléchargement libre sur le site www.editions-eyrolles.com – cela vous épargnera l'effort de frappe et les erreurs de ressaisie. Ces exemples peuvent servir de modèles pour créer rapidement de nouvelles macros, en quelques copier-coller.

Prenez votre temps en lisant les explications et les exemples : chaque phrase est importante. Nulle prétention littéraire pour ce texte technique. N'hésitez pas à relire des passages que vous pensez connaître.

La source de documentation étant presque exclusivement en anglais, nous avons choisi d'aider le lecteur peu familier de cette langue en traduisant les termes importants et en utilisant des noms de variables en français. À l'usage, le fait d'employer des noms français facilite beaucoup l'assimilation, même si on lit couramment l'anglo-américain.

Ce livre n'est pas une simple traduction de documents anglais, ni une collection d'astuces récoltées sur les forums. Il est une synthèse de connaissances et présente de

façon directement utile beaucoup d'informations peu connues, mal documentées ou non documentées. Nous signalons notamment des anomalies de fonctionnement, des limitations, ou des erreurs de documentation afin de vous éviter les difficultés que nous avons rencontrées.

Nous mettons aussi à votre disposition en téléchargement un grand nombre de routines utilitaires pour simplifier de nombreux codages. Leur liste est donnée à l'annexe B.

L'interface utilisateur des versions successives présente inévitablement de petites modifications par rapport à notre version de travail. De plus, la forme des icônes peut varier d'une distribution à l'autre, selon le système d'exploitation, ou la configuration choisie par l'utilisateur. Cela ne devrait pas vous empêcher de retrouver l'équivalent de ce qui est imprimé dans ce livre.

La **première partie** montre l'utilité de la programmation OpenOffice.org, et présente les langages de script et les diverses manières de déclencher un script. Vous verrez comment utiliser l'enregistreur de macro, et pourquoi il est finalement assez limité. Contrairement à la concurrence, OpenOffice.org supporte plusieurs langages de script ; nous en faisons une comparaison. Le but de la programmation OpenOffice.org est souvent d'ajouter des fonctionnalités. Les extensions sont un moyen de les diffuser facilement, nous en montrerons les possibilités ainsi que les outils pour les produire.

Dans la **deuxième partie**, nous décrivons le langage OOoBasic. Vous faites connaissance avec l'environnement de développement intégré, et vous l'utilisez pour écrire et exécuter votre première macro. Vous apprenez les diverses manières d'exécuter une macro. Même si vous connaissez déjà un Basic (par exemple Visual Basic™ qui lui est proche), parcourez les chapitres de cette partie. En cas de problème d'exécution, relisez-la, elle contient bien des détails importants et souvent non décrits dans la documentation officielle.

À partir de la **troisième partie**, vous apprenez à écrire ou modifier des documents OpenOffice.org : Writer, Calc, Draw, etc. Vous aurez besoin d'utiliser l'API, cœur de la programmation OpenOffice.org dans quelque langage que ce soit. Mais nous éviterons les exposés théoriques rébarbatifs pour nous concentrer sur les solutions à des besoins réels. OpenOffice.org réutilise des concepts généraux dans chaque type de documents, mais avec des variations propres à chacun. Nous avons regroupé les principes communs dans le chapitre « Les documents OpenOffice.org », puis décrit les aspects spécifiques aux documents Writer, Calc et Draw/Impress dans les chapitres suivants. Dans chacun de ces chapitres, il n'est nul besoin d'effectuer une lecture complète : après avoir acquis les notions de base, utilisez ensuite le livre comme une référence, et n'approfondissez que les sujets qui vous sont utiles.

La **quatrième partie** va au-delà des manipulations de documents pour vous permettre de construire des applications élaborées. Vous y apprendrez à afficher des dialogues tout à fait semblables à ceux des applications classiques, à utiliser une base de données et des formulaires élaborés. Des aspects plus transversaux sont ensuite traités : gestion de la configuration, gestion de fichiers depuis l'API, utilisation du Dispatcher, interaction avec le monde MS-Windows, et diverses méthodes bien utiles.

Les **annexes** présentent de nombreuses informations complémentaires. Nous expliquons ce qu'est l'API et comment en obtenir des informations pour aller encore plus loin. Nous présentons ensuite une liste de routines utilitaires dont certaines ont été utilisées pour simplifier nos exemples. Nous signalons enfin les ressources Internet incontournables pour qui souhaite se tenir à jour : il s'agit de forums où chercher assistance, ou de sites fournissant des exemples de macros, des documents explicatifs et des outils.

ASPECTS JURIDIQUES

Les descriptions, les exemples et les divers logiciels de cet ouvrage et des fichiers disponibles en téléchargement sont fournis comme potentiellement utiles, mais *sans aucune garantie*, ni explicite ni implicite, y compris les garanties de commercialisation ou d'adaptation dans un but spécifique. Les exemples sont fournis dans un but d'explication et leurs principes sont librement réutilisables.

Quelques routines utilitaires sont soumises à la licence LGPL (comme indiqué en commentaire dans le codage, voir le Zip téléchargeable). Cette licence, peu contraignante, est décrite sur le site <http://www.gnu.org/copyleft/lesser.html>. Une traduction non officielle de la licence LGPL est disponible sur le site http://www.linux-france.org/article/these/licence/lgpl/lgpl_monoblock.html.

Changements par rapport à la précédente édition

Avec cette nouvelle édition, totalement revue et réorganisée, nous avons ajouté des notions qui n'avaient pas été décrites, tenu compte des modifications et ajouts apportés par les versions successives d'OpenOffice.org 2 et 3, amélioré de nombreux codages, mis à jour les références d'adresses Internet, et signalé des outils apparus depuis la version précédente de notre livre.

Pour alléger la lecture, nous avons supprimé ce qui était spécifique de l'ancienne version 1 d'OpenOffice.org, et les limitations propres à la version initiale 2.0. Sauf indication contraire dans le texte, ce qui est décrit est aussi valable pour les dernières versions 2 (versions 2.3.1 et plus récentes). Les copies d'écran sont faites avec la version 3.

Principaux ajouts

Chapitre 1 : les extensions.

Chapitre 2 : information sur la compatibilité VBA.

Chapitre 3 : type `Byte`, sous-type `Decimal`, type `Collection`, type défini par l'utilisateur.

Chapitre 4 : boucle `For Each`.

Chapitre 5 : opérateur `Like`, fonctions `DateDiff`, `DateAdd`, `DatePart`, `Format`.

Chapitre 7 : description complète des options d'export PDF, valeurs d'encodage de caractères pour import et export CSV et texte, propriétés de document définies par l'utilisateur.

Chapitre 8 : surlignement, statistiques du document, tableaux irréguliers en version 3, actualiser un document `Writer`.

Chapitre 9 : fusion de cellules, filtrage, fonction matricielle, liens vers un autre classeur, exemples de *Listener*.

Chapitre 10 : méthodes et propriétés pour gérer un diaporama en cours.

Chapitre 11 : ajout dynamique des contrôles de dialogue, dialogues et messages multilingues.

Chapitre 12 : valeurs d'encodage de caractères pour une base plate, connexion à une base non enregistrée, publipostage plus détaillé.

Chapitre 13 : description systématique des contrôles de formulaire, ajout dynamique des contrôles de formulaire, macros dans un document `Base`, ouvrir et fermer un formulaire depuis un autre formulaire de fichier `Base`.

Chapitre 14 : lire et modifier la configuration `OpenOffice.org`, écrire et lire un fichier texte encodé, formatage par l'API, exemples d'utilisation du `Dispatcher`.

Documents disponibles en téléchargement

Sur le site web des éditions Eyrolles, www.editions-eyrolles.com, vous trouverez la fiche de ce livre, où vous pourrez télécharger gratuitement un fichier Zip. Ce fichier se décompacte dans un répertoire `MacrosLivre` comprenant autant de sous-répertoires que de chapitres donnant des exemples de macros. Les macros d'un chapitre se trouvent dans des documents `OpenOffice.org`, ainsi que les fichiers associés éventuels. La référence du fichier contenant la macro est indiquée en première ligne de chaque macro reproduite dans cet ouvrage.

Avant exécution, il vous faudra copier les fichiers nécessaires dans un répertoire de travail de votre ordinateur, là où, justement, leur exécution est autorisée.

Remerciements

Nous remercions tous les intervenants français et étrangers dans les divers forums OpenOffice.org, soit qu'ils aient apporté leur pierre à la compréhension du produit, soit qu'ils nous aient incités à approfondir tel ou tel domaine grâce à la grande diversité de leurs questions.

Nous remercions tous ceux qui ont bien voulu publier leurs macros, car leur lecture est une source d'information inestimable autant pour les débutants que pour les programmeurs confirmés.

Enfin, merci à Sun, racheté par Oracle, pour avoir laissé en Open Source la suite OpenOffice.org, donnant ainsi à quiconque la possibilité d'étudier la structure d'un logiciel moderne et de très grande envergure. Nous espérons que cet esprit sera conservé.

Merci enfin aux développeurs d'OpenOffice.org pour leur souci constant de rendre automatisable et accessible par l'API tous leurs développements internes.

PREMIÈRE PARTIE

Introduction à la programmation OpenOffice.org

Cette première partie vous montre l'utilité de la programmation OpenOffice.org à travers quelques exemples. Si différents langages de script peuvent être utilisés pour les obtenir, Basic est le plus accessible d'entre eux.

Vous apprendrez comment sécuriser l'utilisation des scripts ou macros, et les diverses manières de déclencher un script. Vous ferez connaissance avec l'enregistreur de macros Basic, puis avec les interfaces utilisateur propres à chaque langage de script supporté (à l'exception de Basic, qui sera le sujet de la partie suivante).

1

Les scripts dans OpenOffice.org

Le terme macro évoque plutôt le langage Basic, qui sera d'ailleurs notre principal langage de programmation dans ce livre. Mais il n'est pas le seul, comme nous allons le voir. On devrait maintenant employer le terme plus général de script, mais les habitudes ont la vie dure, et le terme macro est employé partout dans les interfaces utilisateur.

OpenOffice.org offre différents langages de programmation (langages de script), contrairement à la concurrence. Sa structure permet même d'en rajouter d'autres. Nous serons amenés à signaler quelques concepts avancés, qui seront plus clairs après lecture du reste du livre.

De l'automatisation d'OOo à l'application d'entreprise

Avant de vous lancer dans l'aventure, vous vous demandez peut-être ce qu'on peut bien réaliser d'intéressant avec OOoBasic et l'API d'OpenOffice.org. Eh bien, tout est fonction du besoin. Une « bonne » macro est une macro qui satisfait un besoin, qu'il soit récurrent ou ponctuel. Il n'est pas nécessaire de bâtir un environnement applicatif complet (même si cela est tout à fait possible) et quelques lignes suffisent parfois à rendre des services inestimables au quotidien.

Les macros d'OpenOffice.org permettent d'adapter le logiciel à un besoin spécifique, avec cet avantage indéniable que dans le cas des macros OOoBasic, tout est déjà intégré et prêt à l'utilisation. OOoBasic offre un cadre d'exécution commun pour élaborer des additifs logiciels. Une macro « arrivant » sur un poste est certaine de retrouver ce cadre de travail, et ce quelle que soit la plate-forme utilisée.

Des macros pour les utilisateurs d'OpenOffice.org

Les utilisations des macros sont multiples. On peut bien sûr intervenir directement sur un document en cours pour reproduire une tâche répétitive ou fastidieuse, mais aussi fédérer plusieurs documents pour des traitements transversaux. Bien des utilisateurs ont leurs propres macros, non publiées, qui leur font gagner du temps dans leurs activités quotidiennes, depuis l'application d'un style de caractère en cliquant sur une simple icône jusqu'à la mise en forme de plusieurs documents à la fois.

Une macro d'intérêt général peut être distribuée sous la forme d'une extension. Une extension est un fichier reconnu par OpenOffice.org qui permet de lui ajouter facilement une nouvelle fonctionnalité.

Des applications à part entière pour l'entreprise

Un nombre croissant d'entreprises et d'administrations développent des applications internes basées sur OOoBasic et l'API d'OpenOffice.org. Des outils internes à l'API peuvent notamment permettre d'envisager une utilisation à travers un réseau voire Internet. Là encore, de nombreuses fonctionnalités sont présentes en interne.

Par exemple, si un important fonds documentaire est disponible dans un certain format et qu'il devient nécessaire d'en effectuer une migration pour obtenir une version PDF des documents, les fonctionnalités d'import/export le permettent.

Si des données sont éparpillées dans plusieurs sources et qu'il devient nécessaire de les fédérer voire d'en construire des graphes à intervalles donnés, l'accessibilité à l'API de Calc va pouvoir répondre au besoin.

Si un mailing requiert des interventions particulières ou s'il devient nécessaire de récupérer des données dans des documents contenant des champs utilisateurs afin de les consolider, là encore, l'API et les macros peuvent être utilisées.

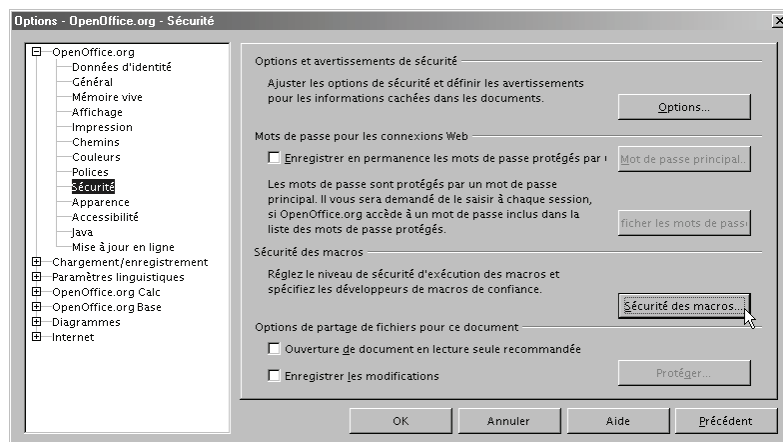
Enfin, les macros peuvent servir à faire de petits scripts simples complètement déconnectés du contexte bureautique, comme des « moulinettes » sur des fichiers texte.

Les macros et la sécurité

La puissance des macros comporte un revers : des individus peuvent écrire des documents anodins contenant des programmes conçus dans un but malveillant. Les utilisateurs de MS-Outlook, MS-Word et MS-Excel en savent quelque chose. En réalité, il est heureusement rare de récupérer de tels documents, mais beaucoup plus courant qu'un correspondant de bonne foi vous envoie un document avec une macro de son cru, et que celle-ci provoque des dégâts dans votre PC ou dans votre configuration OpenOffice.org. Ainsi, d'une manière générale, un document destiné à être diffusé devra être lisible sans macro.

Dans OpenOffice.org, l'utilisateur définit les conditions d'exécution des macros à partir du menu Outils>Options>OpenOffice.org>Sécurité. La figure 1-1 reproduit ce panneau.

Figure 1-1
Entrée vers le panneau
de sécurité des macros

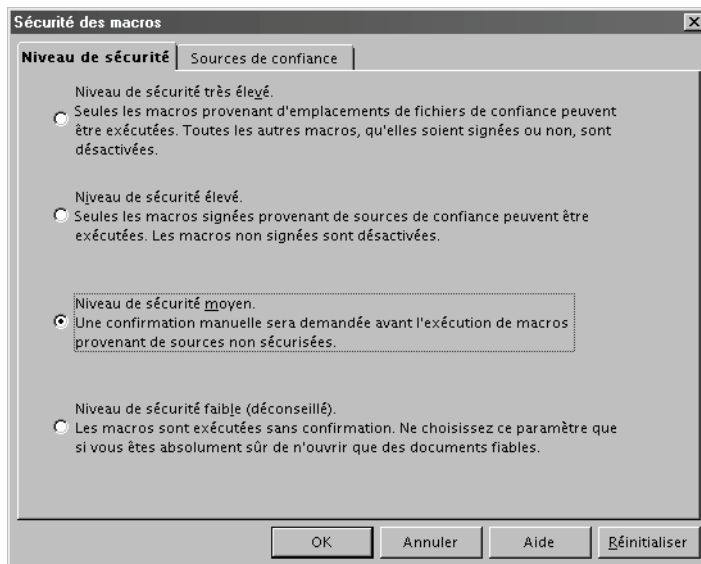


En fait, tout se passe dans le panneau qui apparaît en cliquant le bouton **Sécurité des macros**, qui concerne tous les scripts, pas seulement Basic. Ce panneau comporte deux onglets, le premier est reproduit à la figure 1-2. Les textes explicatifs de chaque niveau ne sont pas tous corrects, nous allons voir exactement ce qu'il en est.

Les différents niveaux de sécurité

Ces niveaux de sécurité ne concernent que les macros contenues dans un document. Les macros intégrées dans votre exemplaire d'OpenOffice.org sont supposées inoffensives (il s'agit des macros hébergées dans les zones Mes macros et Macros OpenOffice.org, et des extensions installées).

Figure 1-2
Onglet Niveau de sécurité
des macros



Niveau faible

Ce niveau de sécurité autorise toute macro, quelle que soit l'origine du document. Ne l'utilisez que si vous aimez vivre dangereusement.

Niveau moyen

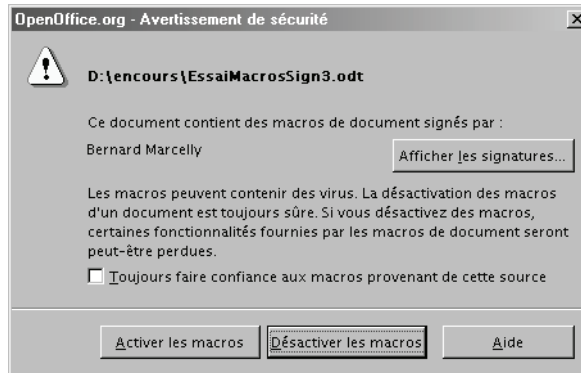
Le niveau de sécurité moyen vous avertit si le document contenant la macro ne se trouve dans aucun de vos répertoires de confiance (voir l'onglet Sources de confiance). Une bonne sécurité consiste à déclarer quelques répertoires de confiance, ceux où vous savez que certains documents nécessitent des macros. Quand vous récupérez un document inconnu, placez-le dans un répertoire ordinaire. Si le document contient des macros, OpenOffice.org vous en avertit (figure 1-3 pour des macros signées ou figure 1-4 pour des macros non signées) en vous donnant plusieurs possibilités d'action :

- Fermer la fenêtre (case X) désactive l'exécution des macros.
- Dans le cas de la figure 1-3, si vous cochez la case *Toujours faire confiance aux macros provenant de cette source*, les macros seront activées, le certificat sera ajouté dans la liste des certificats de confiance (figure 1-7) et la question ne vous sera plus posée pour les documents avec des macros portant la même signature, quel que soit leur emplacement. Ne cochez la case qu'après avoir cliqué sur le bouton *Afficher les signatures*. En effet, le certificat peut être invalide, ou le niveau de confiance faible s'il est délivré gratuitement par Internet.

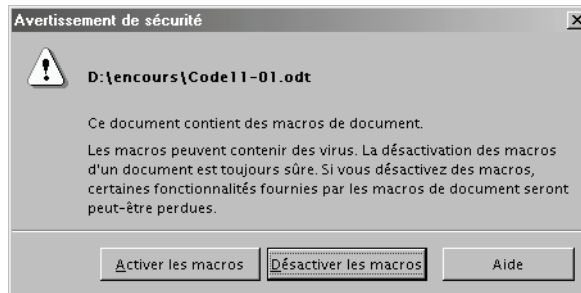
- Dans le cas de la figure 1-4, ou si vous ne cochez pas la case de la figure 1-3, vous pouvez activer ou non l'exécution des macros pour cette fois-ci. Le fait de les désactiver n'empêche absolument pas d'ouvrir le document, ni de visualiser les instructions avec l'éditeur de macros (que nous verrons au chapitre 2).

Figure 1-3

Niveau moyen,
avertissement de macro signée

**Figure 1-4**

Niveau moyen,
avertissement de macro



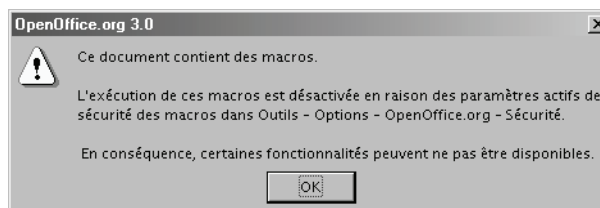
Niveau élevé

Ce niveau autorise l'exécution des macros dont le document se trouve dans un des répertoires de confiance. Si le document se trouve en dehors de ces répertoires, OpenOffice.org considère deux cas :

- Soit les macros ne sont pas signées : elles sont systématiquement désactivées et OpenOffice.org affiche le message d'avertissement de la figure 1-5.

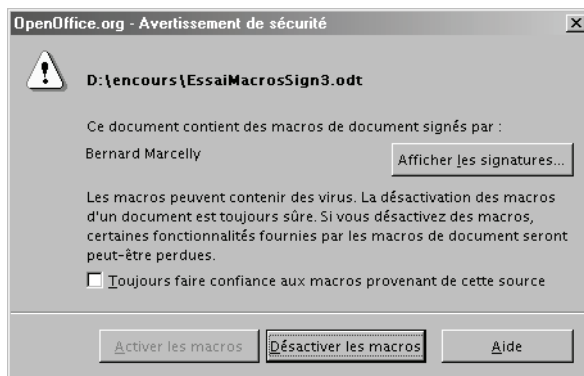
Figure 1-5

Niveau élevé, avertissement
de macro



- Soit les macros sont signées, vous obtenez alors le message de la figure 1-6.

Figure 1-6
Niveau élevé, avertissement
de macro signée



Si vous cochez la case **Toujours faire confiance aux macros provenant de cette source**, vous accédez au bouton **Activer les macros**. Après ouverture du document, le certificat sera ajouté dans la liste des certificats de confiance (figure 1-7) et la question ne vous sera plus posée pour les documents avec des macros portant la même signature, quel que soit leur emplacement. Ne cochez la case qu'après avoir cliqué sur le bouton **Afficher les signatures**. En effet, le certificat peut être invalide, ou le niveau de confiance faible s'il est délivré gratuitement par Internet. Si vous ne cochez pas la case, l'exécution de macros du document est désactivée.

Niveau très élevé

La sécurité se base exclusivement sur les répertoires de confiance. Les macros de documents situés en dehors de ces répertoires sont systématiquement désactivées et OpenOffice.org affiche le message d'avertissement de la figure 1-5.

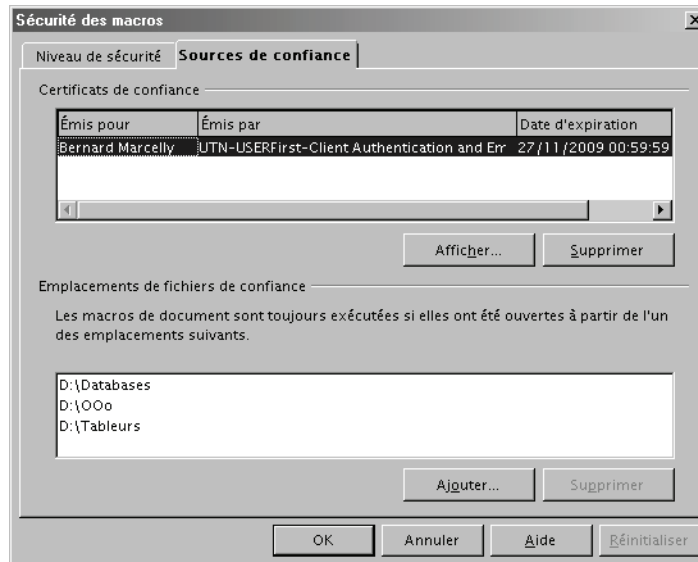
Les sources de confiance

L'onglet **Sources de confiance** est reproduit à la figure 1-7.

La zone du haut liste les certificats de confiance qui représentent des signatures acceptables pour les macros. Lorsque, sur le message des figures 1-4 ou 1-6, vous cochez **Toujours faire confiance aux macros de cette source**, le certificat correspondant est automatiquement ajouté dans la liste des certificats de confiance.

La zone du bas liste les répertoires de confiance. Pour chacun, la confiance s'étend à toute l'arborescence de sous-répertoires qu'il contient. Évitez de mettre la racine d'un disque principal car tout le disque serait alors considéré comme de confiance, ce qui n'aurait plus de signification.

Figure 1-7
Onglet Sources de confiance



Les signatures numériques

La création ou l'importation de certificats permettant de valider une signature numérique nécessite d'autres logiciels comme Firefox et autres navigateurs Internet. Nous ne détaillerons pas ici les procédures.

Se documenter

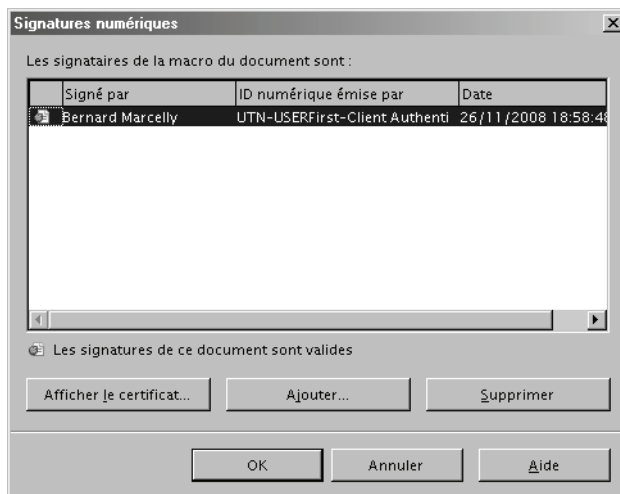
Dans l'aide (F1), cherchez dans l'index Signature et Utilisation des signatures numériques.

Un document OpenOffice.org peut être certifié numériquement par un ou plusieurs certificats. Notez qu'avec la version 3 d'OpenOffice.org les documents à signer doivent être sauvegardés au format ODF 1.2. On certifie un document avec le panneau obtenu par **Fichier>Signatures numériques**. Une fois certifié OpenOffice.org calcule une signature numérique sur le document. Cette signature de document est indépendante de la signature éventuelle des macros du document. De manière similaire, les macros d'un document peuvent être signées avec le panneau de la figure 1-8, obtenu avec le menu **Outils>Macro>Signature numérique**.

Le processus est le suivant :

- 1 Sauvez votre document, sans le fermer.
- 2 Ouvrez le panneau de la figure 1-8 et ajoutez un ou plusieurs certificats.
- 3 Fermez le document *sans* le sauver !

Figure 1-8
Signatures numériques
des macros



Le document peut être copié ou déplacé, il gardera ses signatures.

Plus tard, après toute modification du document, répétez exactement le processus, car la sauvegarde supprime les signatures pour éviter qu'un tiers ne modifie les macros en gardant une apparence de sécurité.

L'enregistreur de macros

L'enregistreur de macros enregistre les séquences de manipulations de l'utilisateur sous forme d'une macro, ce qui permet ensuite de reproduire à volonté la même séquence. La méthode est très simple et ne nécessite pas de connaissances de programmation.

Comment enregistrer une macro ?

L'enregistrement est déclenché en cliquant sur Outils>Macros>Enregistrer une macro. À partir de cet instant, toutes les actions sur le document OpenOffice.org contenu dans la même fenêtre sont enregistrées. Vous remarquerez une petite fenêtre en avant-plan : elle vous permet de terminer l'enregistrement.

Ayant cliqué sur cette fenêtre de terminaison de macro, un autre panneau apparaît. Choisissez dans quelle bibliothèque et quel module vous souhaitez sauvegarder votre macro. Nous expliquerons plus loin ces termes et ce panneau. Essentiellement, vous avez le choix entre un module d'une bibliothèque disponible en permanence dans

OpenOffice.org et une bibliothèque propre au document en cours. En général, spécialement pour un débutant, il suffit de choisir la bibliothèque Standard du document en cours et de cliquer le bouton Enregistrer (enregistrer la macro). Un nouveau panneau apparaît, qui vous demande de choisir le nom du module, par exemple `Module1`. Maintenant, votre macro est écrite dans le module et elle a pour nom `Macro1` ou un nom similaire. Vous pouvez changer ce nom dans l'éditeur de macros qui s'affiche.

Notre document exemple `Code-02-01.odt` contient dans sa bibliothèque Standard une séquence réalisée avec l'enregistreur de macros : aller à la fin du document, écrire un texte et terminer le paragraphe. Voici le codage obtenu (les lignes blanches sont omises).

```
sub BonjourEnregistreur
rem -----
rem define variables
dim document as object
dim dispatcher as object
rem -----
rem get access to the document
document = ThisComponent.CurrentController.Frame
dispatcher = createUnoService("com.sun.star.frame.DispatchHelper")
rem -----
dispatcher.executeDispatch(document, ".uno:GoToEndOfDoc", "", 0, Array())
rem -----
dim args2(0) as new com.sun.star.beans.PropertyValue
args2(0).Name = "Text"
args2(0).Value = "L'enregistreur de macros vous salue !"
dispatcher.executeDispatch(document, ".uno:InsertText", "", 0, args2())
rem -----
dispatcher.executeDispatch(document, ".uno:InsertPara", "", 0, Array())
end sub
```

Le document exemple contient d'autres scripts réalisant exactement la même chose en Basic, JavaScript, BeanShell, Python. En comparant les codages, vous constaterez que l'enregistreur de macro produit un codage très différent des autres : en effet il n'utilise pas les fonctions de l'API, mais seulement un mécanisme appelé *dispatch*.

Un outil limité

L'enregistreur de macros souffre de limitations assez sévères :

- Il n'est disponible que sous Writer et Calc.
- Il ne sait que « mimer » des actions de l'utilisateur, et encore, pas toutes.
- Il utilise des commandes peu documentées (les « slot ID »).
- Il ne permet pas d'écrire des macros interactives.

- Il produit un codage Basic non optimisé qui est assez difficile à lire, sans rapport avec un « vrai » codage Basic OpenOffice.org.

Par ailleurs, mais c'est le principe d'un tel enregistreur, il ne peut produire que du codage linéaire (c'est-à-dire qu'il est incapable de faire par exemple une boucle pour répéter une action sur une liste d'objets ou de choisir entre plusieurs alternatives).

Pour avoir plus de possibilités, il faut écrire soi-même les instructions de la macro, ce qui nécessite de connaître un langage de programmation et l'API OpenOffice.org. C'est la voie qui est développée dans ce livre.

Il est cependant des cas où l'enregistreur de macros nous sera utile : lorsque l'API ne permet pas certaines manipulations que peut réaliser l'enregistreur, ou seulement au prix de développements complexes. Il est alors possible de résoudre la difficulté en combinant un codage Basic avec les instructions créées par l'enregistreur.

Les différents langages de script

Depuis la version 2, OpenOffice.org intègre plusieurs langages de script, et pas seulement Basic. Chaque langage a ses avantages et ses défauts, et un programmeur expérimenté préférera celui qui est le plus adapté à son projet, ou même à une partie du projet. Nous appellerons macro ou script tout programme réalisé avec un de ces langages.

Certains détails de cette section sont destinés aux lecteurs ayant acquis une bonne connaissance de la programmation avec OpenOffice.org.

Basic OpenOffice.org

Basic sera notre langage de développement dans ce livre, mais nous ne l'aborderons qu'avec le chapitre 2. Pour vous donner un avant-goût, voici une petite macro `BonjourBasic`, que vous trouverez dans la bibliothèque `Library1` du document exemple `Code-02-01.odt`. Elle réalise l'équivalent de l'exemple de l'enregistreur de macros.

```
Sub BonjourBasic
Dim monDocument As Object, monTexte As Object, monCursur As Object
monDocument = ThisComponent
monTexte = monDocument.Text
monCursur = monTexte.createTextCursor
monCursur.gotoEnd(false)
monTexte.insertString(monCursur, "Basic et l'API vous saluent !", false)
monTexte.insertControlCharacter(monCursur, _
com.sun.star.text.ControlCharacter.PARAGRAPH_BREAK, false)
End Sub
```

Particularités des autres langages de script

Les langages autres que Basic sont gérés par le Scripting Framework, qui fait l'objet d'un chapitre complet dans le *Developer's Guide* (documentation en anglais) disponible en ligne à l'adresse suivante :

http://wiki.services.openoffice.org/wiki/Documentation/DevGuide/OpenOffice.org_Developers_Guide

Une variable prédéfinie `XSCRIPTCONTEXT` est disponible dans un script. Cet objet expose trois méthodes :

- `getDocument()` renvoie l'objet document en cours.
- `getDesktop()` renvoie l'objet application OpenOffice (équivalent du Basic `StarDesktop`).
- `getComponentContext()` renvoie le contexte, nécessaire pour appeler certaines méthodes.

Les dialogues, que nous verrons au chapitre 11, peuvent être appelés par un script quelconque, et les événements de dialogue peuvent aussi être traités par un script.

Il est plus facile de développer un script non Basic dans *Mes macros*, quitte à le transférer ensuite dans un document avec adaptation éventuelle.

Java compilé

OpenOffice.org peut exécuter des scripts en Java compilé (fichiers `.jar`). Mais il n'existe pas de panneau Macros correspondant, et leur installation devra être faite manuellement par un programmeur confirmé ou par le biais d'extensions.

JavaScript

Langage bien connu des créateurs de sites web, JavaScript est utilisé ici comme langage de programmation indépendant de toute page web. Chaque macro JavaScript est contenue dans un fichier portant l'extension `.js`, stocké dans un répertoire bibliothèque. Le panneau Macros JavaScript de la figure 1-9 est obtenu par le menu *Outils>Macros>Gérer les macros>JavaScript*. Ici la macro se trouve dans la bibliothèque `Library3` du document `Code01-01.odt`.

Vous pouvez créer une nouvelle bibliothèque ou une nouvelle macro dans celle sélectionnée. Le bouton *Éditer* affiche le contenu de la macro dans la fenêtre de l'éditeur OpenSource Rhino (voir figure 1-10). Il offre des possibilités d'évaluation de variables, de pas-à-pas et de point d'arrêt ; mais il n'y a ni coloration syntaxique ni aide en ligne sur les instructions. Il souffre actuellement de défauts rédhibitoires (Issue 70176, Issue 70215) qui conduisent à le déconseiller et préférer un éditeur séparé pour modifier le fichier.

Figure 1–9
Panneau Macros JavaScript

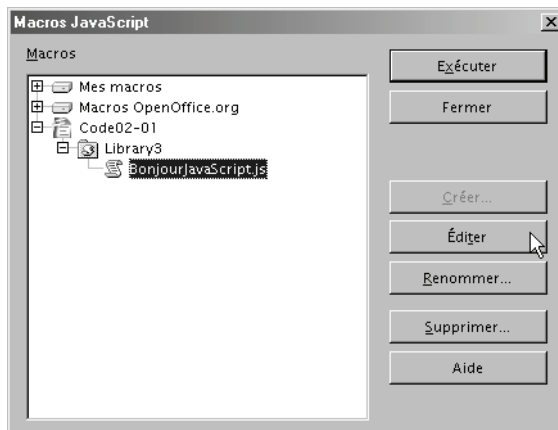
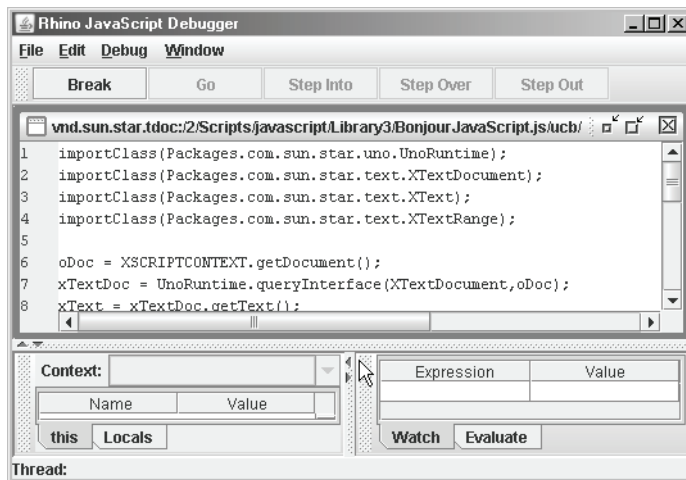


Figure 1–10
Éditeur de JavaScript



Quand vous créez une nouvelle macro JavaScript, OpenOffice.org écrit automatiquement un codage de type HelloWorld. S'il peut servir d'exemple sous Writer, il est inutilisable sous Calc, Draw, etc.

Voici le codage du script `BonjourJavaScript.js`. Comme pour Java, il est nécessaire d'obtenir explicitement chaque interface dont on utilise une méthode. Par contre il n'y a pas de typage de données.

```
importClass(Packages.com.sun.star.uno.UnoRuntime);
importClass(Packages.com.sun.star.text.XTextDocument);
importClass(Packages.com.sun.star.text.XText);
importClass(Packages.com.sun.star.text.XTextRange);
```

```
oDoc = XSCRIPTCONTEXT.getDocument();
xTextDoc = UnoRuntime.queryInterface(XTextDocument,oDoc);
xText = xTextDoc.getText();
xTCursor = xText.createTextCursor();
xTCursor.gotoEnd(false);
xText.insertString( xTCursor, "JavaScript vous salue ! " , false);
xText.insertControlCharacter(xTCursor,
Packages.com.sun.star.text.ControlCharacter.PARAGRAPH_BREAK, false);
```

Les arguments passés à une macro JavaScript sont récupérés dans une variable globale prédéfinie `ARGUMENTS` qui est un tableau de valeurs de type `Object`. Par exemple, ici on récupère l'objet événement transmis par le déclenchement d'un bouton de formulaire :

```
evt = ARGUMENTS[0];
```

Pour aller plus loin

Site web de Rhino, en anglais :

► <http://www.mozilla.org/rhino/>

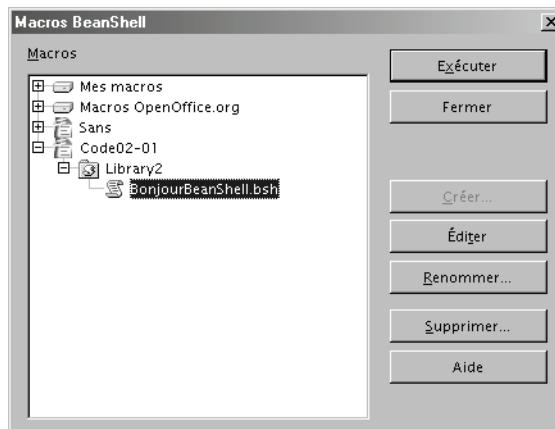
Un site web français sur JavaScript :

► <http://www.toutjavascript.com/>

BeanShell

BeanShell est une sorte de Java interprété et plus simple au niveau des déclarations. Chaque macro BeanShell est contenue dans un fichier portant l'extension `.bsh`, stocké dans un répertoire bibliothèque. Le panneau Macros BeanShell de la figure 1-11 est obtenu par le menu `Outils>Macros>Gérer les macros>BeanShell`. Ici la macro se trouve dans la bibliothèque `Library2` du document `Code01-01.odt`.

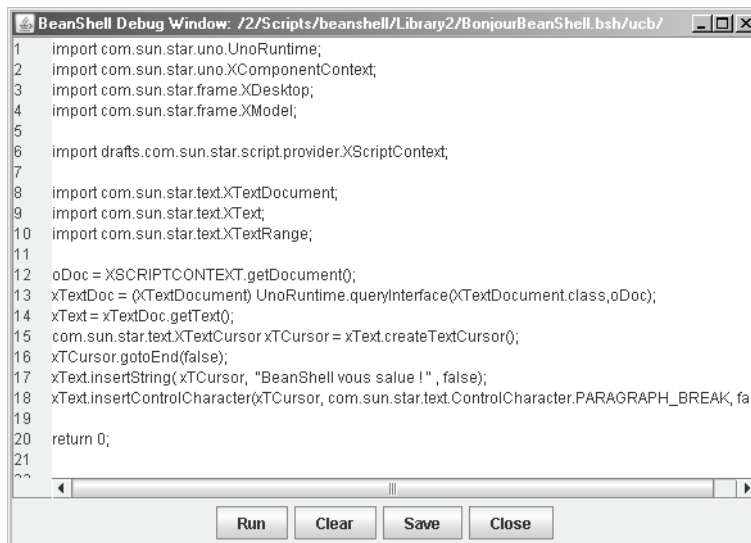
Figure 1-11
Panneau Macros BeanShell



Vous pouvez créer une nouvelle bibliothèque ou une nouvelle macro dans celle sélectionnée. Le bouton Éditer affiche le contenu de la macro dans la fenêtre de l'éditeur BeanShell (voir figure 1-12). C'est un éditeur rustique qui n'offre ni police à espacement fixe, ni coloration syntaxique, et aucun outil de mise au point.

Figure 1-12

Éditeur de BeanShell



Comme avec JavaScript, lorsque vous créez une nouvelle macro BeanShell, OpenOffice.org présente automatiquement un codage type « HelloWorld ». Il peut servir d'exemple sous Writer, mais il est inutilisable sous Calc, Draw, etc.

Voici le codage du script `BonjourBeanShell.bsh`. Comme pour Java, il est nécessaire d'obtenir explicitement chaque interface dont on utilise une méthode. Le principal intérêt de BeanShell réside dans sa capacité à utiliser des codages Java tout en profitant d'un langage plus souple pour les types de données.

```

import com.sun.star.uno.UnoRuntime;
import com.sun.star.uno.XComponentContext;
import com.sun.star.frame.XDesktop;
import com.sun.star.frame.XModel;

import drafts.com.sun.star.script.provider.XScriptContext;

import com.sun.star.text.XTextDocument;
import com.sun.star.text.XText;
import com.sun.star.text.XTextRange;

```

```
oDoc = XSCRIPTCONTEXT.getDocument();
xTextDoc = (XTextDocument) UnoRuntime.queryInterface(
    XTextDocument.class, oDoc);
xText = xTextDoc.getText();
com.sun.star.text.XTextCursor xTCursor = xText.createTextCursor();
xTCursor.gotoEnd(false);
xText.insertString( xTCursor, "BeanShell vous salue ! " , false);
xText.insertControlCharacter(xTCursor,
    com.sun.star.text.ControlCharacter.PARAGRAPH_BREAK, false);

return 0;
```

Les arguments passés à une macro BeanShell sont récupérés dans une variable globale prédéfinie `ARGUMENTS`, qui est un tableau de valeurs de type `Object`. Par exemple, ici on récupère l'objet événement transmis par le déclenchement d'un bouton de formulaire :

```
evt = (ActionEvent) ARGUMENTS[0];
```

Plus d'informations

► <http://www.beanshell.org/>

Python

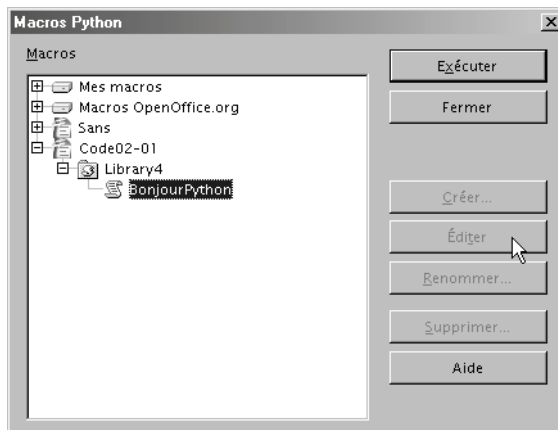
Python est un langage Open Source, très puissant et original. Les points remarquables, par rapport à Basic, sont les suivants :

- L'indentation obligatoire facilite la relecture.
- Les variables ne sont pas déclarées mais leur usage est contrôlé. Elles peuvent changer de type dynamiquement.
- Les algorithmes sont plus simples grâce aux fonctions puissantes incluses dans Python et ses modules principaux.
- La gestion des erreurs est celle des langages modernes.
- La programmation objet facilite la conception de programmes complexes.

Python permet la création de composants UNO, contrairement à Basic. Un composant UNO ajoute un service API qui peut être utilisé par tout autre langage.

Chaque macro Python est une fonction déclarée dans un fichier ayant l'extension `.py`, stocké dans le sous-répertoire `Scripts/python/` ou un sous-répertoire de celui-ci. Le panneau Macros Python de la figure 1-13 est obtenu par le menu `Outils>Macros>Gérer les macros>Python`. Ici la macro se trouve dans la bibliothèque `Library3` du document `Code01-01.odt`. Un fichier source peut comporter plusieurs fonctions appelables.

Figure 1-13
Panneau Macros Python



Vous remarquerez sur la figure 1-13 que les boutons Créer, Éditer, Renommer, Supprimer sont inactifs ; ce qui ne laisse en fait que la possibilité d'exécuter un codage existant. OpenOffice.org n'offre pas encore d'éditeur pour Python. Le développeur de macros Python doit donc utiliser un éditeur externe pour modifier son fichier et travaillera dans Mes macros. Il n'y a pas d'outil de mise au point, et les messages d'erreur apportent malheureusement peu d'informations utiles.

Les programmeurs Python noteront qu'il est nécessaire d'utiliser l'interpréteur Python intégré à OpenOffice.org, et que l'IDLE n'est pas non plus utilisable pour exécuter un script dans OpenOffice.org. Ces limitations sont dues au manque de développeurs. Souhaitons que quelques développeurs ou des entreprises généreuses proposent leur aide à la communauté OpenOffice.org afin de mieux intégrer ce langage. Dans l'état actuel, les avantages de Python dans OpenOffice sont plutôt réservés aux programmeurs expérimentés sachant de surcroît lire l'anglais.

Voici le codage du script BonjourPython. L'utilisation de l'API OpenOffice.org est très semblable à Basic.

Faites attention à ne pas oublier deux particularités de Python : respecter la casse (majuscules/minuscules) pour les noms de variables, et mettre des parenthèses vides dans tout appel d'une méthode sans argument.

```
from com.sun.star.text.ControlCharacter import PARAGRAPH_BREAK

def BonjourPython( ):
    """Ecrit un texte dans le document Writer"""
    monDocument = XSCRIPTCONTEXT.getDocument()
    monTexte = monDocument.Text
    monCurseur = monTexte.createTextCursor()
    monCurseur.gotoEnd(False)
```

```
monTexte.insertString(monCurseur, "Python vous salue !", False)
monTexte.insertControlCharacter(monCurseur, PARAGRAPH_BREAK, False)
return None
```

Pour plus d'informations

Deux pages web de référence, en anglais, à lire attentivement pour développer des macros en Python :

- http://wiki.services.openoffice.org/wiki/Python_as_a_macro_language
- <http://udk.openoffice.org/python/python-bridge.html/>

Le site web Python, en anglais :

- <http://www.python.org/>

L'Association francophone Python :

- <http://www.afpy.org/>

Exécuter une macro depuis OpenOffice.org

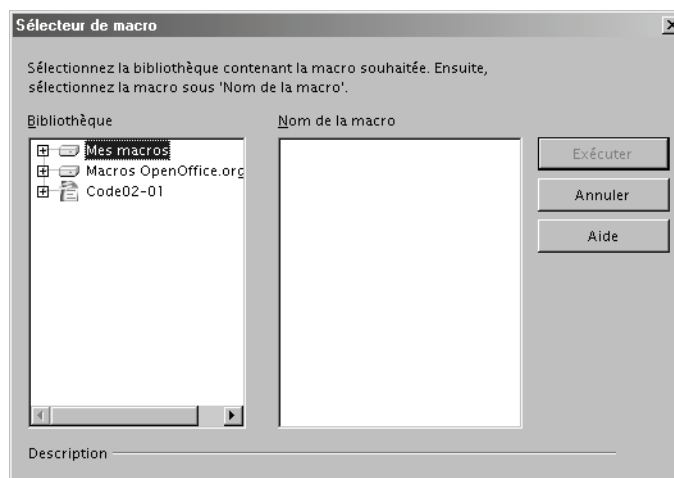
Une macro s'intègre dans les mécanismes de votre exemplaire d'OpenOffice.org, ajoutant ainsi une fonctionnalité, soit dans le contexte d'un type de document, soit pour tous les documents. Vous pouvez alors la déclencher de différentes façons.

Exécuter une macro depuis le menu Outils

Avec le menu Outils>Macros>Exécuter la macro vous obtenez le panneau reproduit à la figure 1-14.

Figure 1-14

Panneau du
Sélecteur de macro

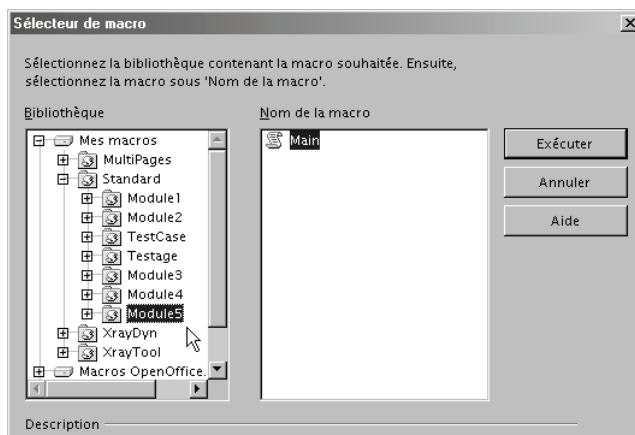


Remarquez qu'il existe trois arbres contenant chacun des bibliothèques de macros :

- Mes macros correspond aux macros que vous avez ajoutées et qui sont disponibles pour toute application OpenOffice.org et depuis tout document.
- Macros OpenOffice.org contient des bibliothèques de macros fournies par OpenOffice.org. Dans une installation réseau, l'administrateur peut y ajouter des bibliothèques qui seront alors disponibles pour tous les utilisateurs OpenOffice.org.
- Code02-01 est le nom d'un document que nous avons ouvert. Un document peut aussi contenir des bibliothèques de macros.

Développez l'arborescence jusqu'à trouver le module contenant votre macro, sélectionnez-la, et cliquez sur le bouton Exécuter. Dans la figure 1-15 nous avons choisi une macro Basic contenue dans Mes macros.

Figure 1-15
Sélection d'une macro Basic



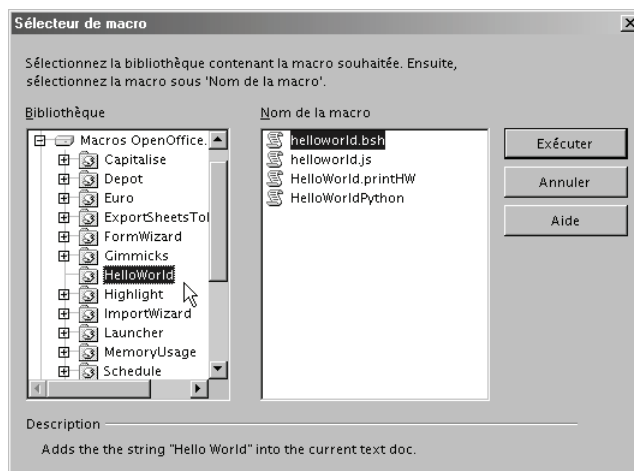
Ce même panneau permet de choisir une macro écrite dans un autre langage que Basic. Il en existe quelques-unes fournies avec l'installation, dans la branche Macros OpenOffice.org. La figure 1-16 vous en montre plusieurs :

- `helloWorld.bsh` est une macro BeanShell ;
- `helloWorld.js` est une macro JavaScript ;
- `HelloWorld.printHW` est une macro en Java compilé ;
- `HelloWorldPython` est une macro en Python.

Vous remarquerez que, sauf pour les macros Basic, il est possible d'afficher un commentaire descriptif de la macro. Le document exemple Code01-01.odt, disponible dans le Zip téléchargeable sur le site des éditions Eyrolles, contient lui aussi des macros écrites dans ces langages.

Vous retrouverez la même structure arborescente dans les autres manières de déclencher une macro par l'interface utilisateur.

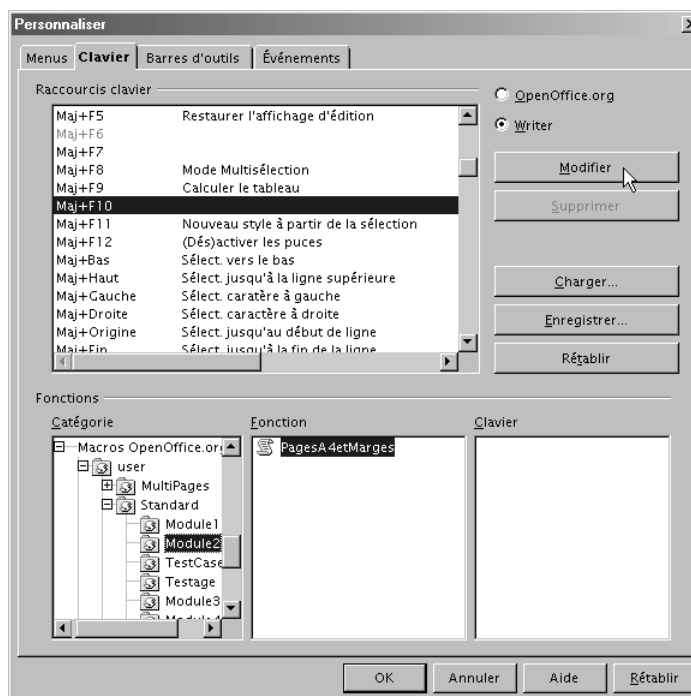
Figure 1-16
Macros écrites
en divers langages



Exécuter une macro depuis un raccourci clavier

Avec le menu Outils>Personnaliser vous obtenez le panneau Personnaliser, qui comporte un onglet Clavier. Cet onglet est représenté sur la figure 1-17.

Figure 1-17
Ajouter un raccourci clavier



Pour affecter une macro à un raccourci clavier afin de pouvoir ensuite l'exécuter d'une simple combinaison de touches, il faut commencer par déterminer si la macro sera disponible pour une application particulière (ici tous les documents Writer), ou pour toute application OpenOffice.org (Calc, Draw, Writer, etc). Selon ce choix, le cadre Raccourcis clavier présente la liste des raccourcis possibles, en indiquant ceux déjà utilisés. Choisissez un raccourci inutilisé ou supprimez un raccourci déjà attribué.

Passez ensuite à la minuscule fenêtre Catégorie. Il faut utiliser les ascenseurs pour l'explorer. Dans la partie inférieure sont regroupées les macros dans la branche Macros OpenOffice.org. Cliquez sur les signes + et vous trouvez :

- user : correspond à l'arborescence Mes macros vue avec le menu Outils.
- share : correspond à l'arborescence Macros OpenOffice.org du menu Outils.
- un nom de document : regroupe les macros éventuellement présentes dans un document actuellement ouvert. Attention, il faut jamais utiliser une macro de document ! En effet, le raccourci sera valable pour tout document.

En cliquant successivement pour développer l'arborescence, vous finissez par faire apparaître dans la fenêtre Fonctions une liste de noms de macros. Une fois que le bon raccourci et la bonne commande sont tous deux sélectionnés dans les panneaux du haut et du bas, cliquez sur le bouton Modifier. La touche ou combinaison de touches s'affiche dans la fenêtre Clavier et le nom de la macro s'inscrit en face du raccourci. Cliquez sur OK pour refermer la boîte de dialogue.

Dans votre document, saisissez un paragraphe, puis testez la macro en l'invoquant par son raccourci. À l'appel du raccourci clavier, la macro s'exécute.

Exécuter une macro avec un bouton de barre d'outils

Vous pouvez également lancer votre macro en cliquant sur un nouveau bouton dans une des barres d'outils.

Cliquez sur la flèche descendante tout au bout à droite sur la barre d'outils. Dans le menu contextuel, choisissez Personnaliser la barre d'outils. Vous obtenez le panneau de la figure 1-18.

Choisissez d'enregistrer dans OpenOffice.org ou dans une des applications (ici Writer). Cliquez sur la position de votre futur bouton, puis cliquez sur Ajouter. Vous obtenez le panneau de la figure 1-19.

Le choix de la macro s'effectue comme pour un raccourci clavier. Vous aurez sans doute envie d'affecter une icône à ce nouveau bouton. Dans le panneau de la figure 1-18 cliquez sur le bouton Modifier puis choisissez Changement d'icône dans le menu.

Figure 1–18
Personnaliser
une barre d'outils

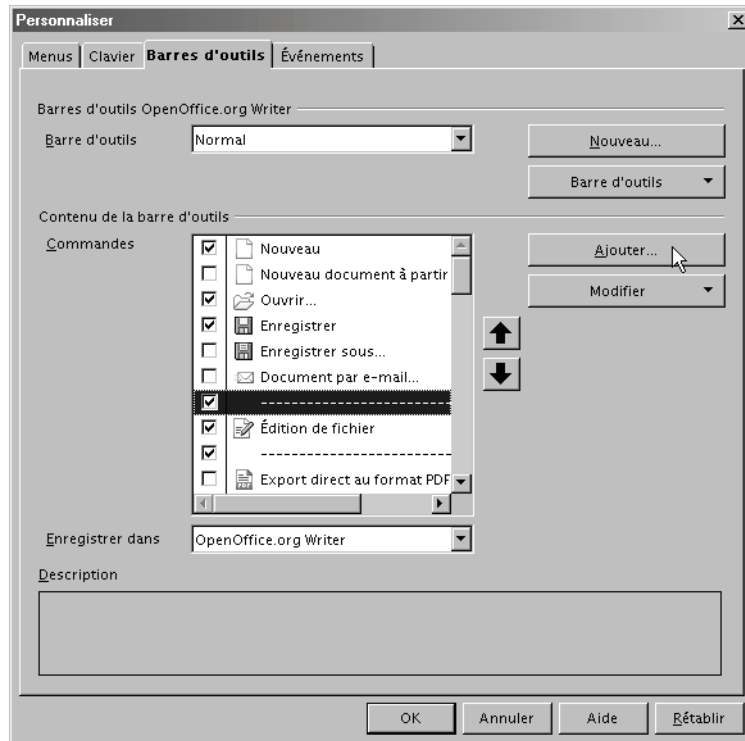
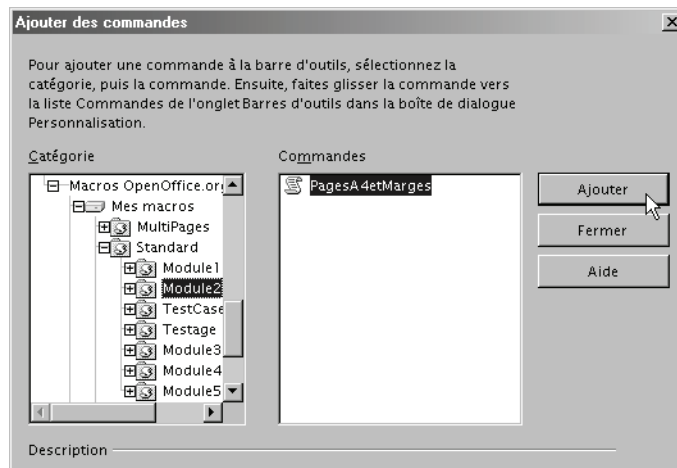


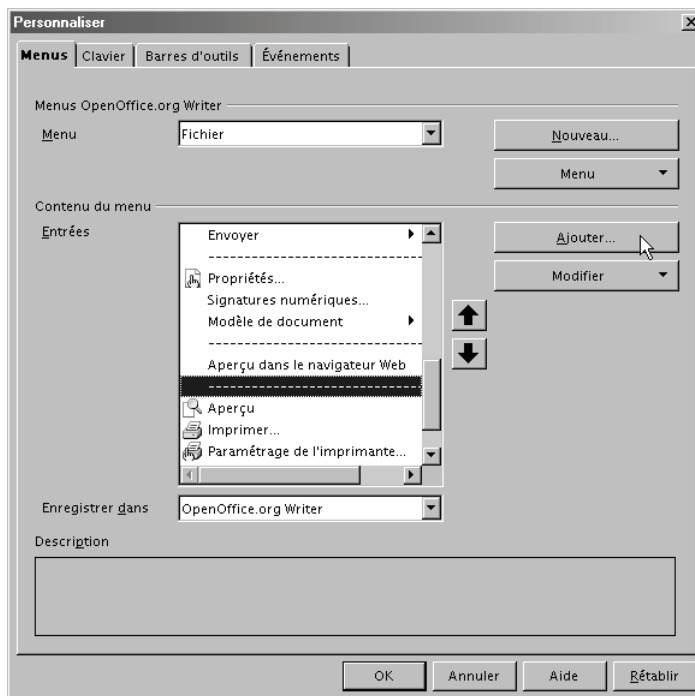
Figure 1–19
Ajouter une macro
sur une barre d'outils



Exécuter une macro par une entrée de menu

OpenOffice.org vous permet de modifier les éléments dans une liste de menu. Il est ainsi possible d'ajouter un élément qui déclenchera une macro. Avec le menu Outils>Personnaliser vous obtenez le panneau Personnaliser, qui comporte un onglet Menus, représenté sur la figure 1-20.

Figure 1-20
Personnaliser
une entrée de menu



Choisissez d'enregistrer l'entrée de menu dans l'application (ici, Writer) ou dans un document ouvert. Le reste de la procédure est identique à celle présentée à la section précédente.

Exécuter une macro depuis une extension

Une extension peut apporter une nouvelle barre d'outils, ou modifier une barre d'outils existante, ou encore ajouter des entrées de menu ; tout dépend du concepteur de l'extension. Une fois installée, les macros de l'extension peuvent être déclenchées avec les nouveaux boutons ou les nouvelles entrées de menu.

Exécuter une macro depuis un document

Un simple bouton de formulaire déposé sur une page de document Writer, Calc ou Draw permet de déclencher une macro. Les contrôles de formulaire utilisent très fréquemment des macros, comme nous le verrons au chapitre 13. Une forme dessinée dans Writer, Calc, Draw, Impress peut déclencher une macro lorsque l'utilisateur clique dessus.

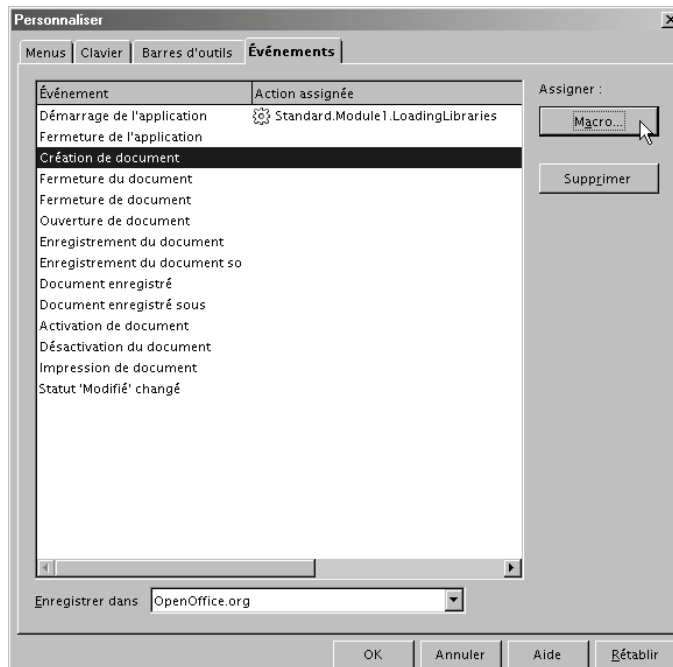
Sur une page de document Writer, vous pouvez insérer un champ permettant de déclencher une macro par hyperlien. Pour cela, dans le menu Insertion>Champs>Autres, cliquez sur l'onglet Fonctions et dans le cadre Type de champ, cliquez sur Exécuter la macro. Choisissez votre macro, puis tapez le texte du lien dans la zone Annotation, et insérez.

Dans Calc, il est possible de créer (avec une macro) votre fonction de calcul pour l'utiliser dans une formule de cellule. Nous verrons au chapitre 9 comment procéder. Notez également que le contrôle de validité d'une cellule peut appeler une macro en cas d'entrée non valide.

Exécuter une macro sur un événement

Pour certains usages, il est parfois intéressant de déclencher automatiquement une macro sur un événement tel que l'ouverture ou la fermeture d'un document. Pour cela, nous utilisons l'onglet Événements du panneau Personnaliser (voir la figure 1-21).

Figure 1-21
Déclencher une macro
sur un événement



Déterminez si l'affectation doit concerner OpenOffice.org ou le document ouvert. Choisissez l'événement. Cliquez sur le bouton Macro. Le panneau Sélecteur de macro apparaît, la suite est identique aux cas précédents. La figure montre qu'une macro est déjà affectée à l'événement Démarrage de l'application.

Bien d'autres événements peuvent faire l'objet d'un traitement par macro, par exemple un clic de souris, une touche enfoncée, etc.

Exécuter une macro en ligne de commande

Il est parfaitement possible de lancer OpenOffice.org pour simplement exécuter une macro, sans passer par l'interface utilisateur. La syntaxe diffère selon le système d'exploitation. Nous décrirons surtout l'appel d'une macro Basic sous MS-Windows, puis nous verrons comment exécuter des macros d'autres langages. Enfin, nous signalerons les particularités propres à Linux.

Cette section nécessite des connaissances de base sur les programmes en ligne de commande (MS-DOS ou Unix Shell). Nous supposons en outre que vous avez une connaissance minimale du Basic OpenOffice.org, sinon nous vous conseillons de vous reporter auparavant au chapitre 2.

Sous Windows

Lancer une macro Basic résidente

On appelle « résidente » une macro qui se trouve dans une des bibliothèques de *Mes macros* ou *Macros OpenOffice.org*, et qui est donc indépendante de tout document. Pour l'exécuter, il faut indiquer successivement les éléments suivants dans la ligne de commande :

- 1 Le chemin d'accès et le nom de l'exécutable d'OpenOffice.org (*soffice.exe*). Notons que ce chemin dépend du système d'exploitation, de la version et des conditions d'installation d'OpenOffice.org.
- 2 Le paramètre optionnel `-headless` si on souhaite empêcher l'affichage de tout message à destination de l'utilisateur.
- 3 Une séquence de caractères qui se présente sous la forme du mot `macro` suivi du nom de la bibliothèque, du nom du module, du nom de la macro, et de ses arguments :

```
macro:///maLib.monModule.maMacro(Arg1, Arg2,...)
```

Ressources

Dans le Zip téléchargeable, vous trouverez dans le dossier des macros de ce chapitre, les fichiers correspondants à nos exemples.

Dans Mes Macros, créez la bibliothèque `maBibli1` et le module `monModule`. Dans ce dernier, écrivez la macro suivante, ou faites un copier-coller de la même macro qui se trouve dans le fichier `Code01-02.odt` :

```
' écrit un fichier texte comportant n fois le caractère c
Sub bavard(c As String, n As Integer)
Dim f As Integer
f = FreeFile

open "C:\essaiMacro.txt" for Output As f
Write #f, Time & " bavard a dit : " & String(n, c)
close #f

msgbox "bavard a terminé"
end sub
```

Cette macro écrit l'heure courante et un texte dépendant de la valeur de ses arguments. Adaptez éventuellement le chemin du fichier résultat `essaiMacro.txt` à votre propre configuration.

Réalisez ce fichier batch (fichier `LanceBavard1.bat`) contenant ceci (l'instruction principale doit être écrite en une seule ligne).

```
rem lancement de la macro
"C:\Program Files\OpenOffice.org 3\program\soffice.exe"
➤ "macro:///maBibli1.monModule.bavard(A, 20)"
pause
```

Modifiez si besoin le chemin d'accès à `soffice`, il dépend de la version OpenOffice.org utilisée. Mettez ce fichier batch dans un répertoire où OpenOffice.org autorise les exécutions de macros.

Arrêtez l'application OpenOffice.org, y compris le lanceur. Exécutez le fichier batch. La fenêtre MS-DOS va s'ouvrir, puis le logo de démarrage d'OpenOffice.org va s'afficher et, après un certain temps, le message « bavard a terminé » apparaîtra dans un petit panneau. Cliquez sur OK, puis terminez l'exécution du batch en appuyant sur une touche. Vérifiez que le fichier `essaiMacro.txt` est bien écrit.

Réalisez un deuxième fichier batch, (LanceBavard2.bat) contenant cette variante :

```
rem lancement de la macro
"C:\Program Files\OpenOffice.org 3\program\soffice.exe"
  ➤ -headless "macro:///maBibli1.monModule.bavard(B, 15)"
pause
```

Arrêtez l'application OpenOffice.org, y compris le lanceur. Exécutez le fichier batch. La fenêtre MS-DOS va s'ouvrir et, après un certain temps, affichera l'attente. Aucun autre message n'est apparu. Terminez l'exécution du batch en appuyant sur une touche. Vérifiez que le fichier `essaiMacro.txt` est bien écrit avec cette deuxième commande.

Lorsque l'argument `-headless` est employé, les éventuelles boîtes de dialogue de OOo, comme la confirmation de lancement des macros, reçoivent la réponse par défaut.

Ouvrez un document OpenOffice.org quelconque et relancez le même fichier batch `LanceBavard2.bat`. Cette fois-ci, le message « bavard a terminé » s'affiche, malgré l'option `-headless`.

Arguments d'appels de la macro

Comme vous l'avez constaté, le premier argument est considéré comme une chaîne d'un seul caractère ; le deuxième est aussi une chaîne de caractères, mais Basic la convertit en une valeur numérique valide pour le paramètre `n` de la macro.

Une chaîne de caractères est transmise telle quelle, sans ajouter de guillemets, sous Windows XP. Sous d'autres versions de MS-Windows, il peut être nécessaire d'ajouter des guillemets. Il n'est pas possible de transmettre une chaîne de caractères comportant un guillemet ou une virgule, et les lettres accentuées sont modifiées.

On peut transmettre à la macro les arguments d'appel du fichier batch, par exemple :

```
"macro:///maBibli1.monModule.bavard(%1, %2)"
```

Lancer une macro Basic contenue dans un document

Si le batch appelle une macro contenue dans un document OpenOffice.org la méthode diffère sur plusieurs points :

- On doit ajouter le chemin d'accès au document s'il n'est pas déjà chargé.
- L'argument `macro` doit comporter le nom court, sans extension, du document, sous cette forme (attention au nombre de caractères /) : `macro://monDoc/maLib.monModule.maMacro(Arg1, Arg2,...)`
- Le document ne se fermera pas automatiquement.

Sur le dernier point, il est possible d'écrire à la fin de la macro une instruction fermant le document, voir le chapitre 7.

Réalisez ce fichier batch (LanceBavard3.bat) :

```
rem lancement de la macro
"C:\Program Files\OpenOffice.org 3\program\soffice.exe"
  ➤ "Code01-02.odt" "macro://Code01-02/Standard.monModule.bavard(C,20)"
pause
```

Mettez ce batch ainsi que le fichier Code01-02.odt (disponible dans le Zip téléchargeable) dans un même répertoire, qui doit être le répertoire courant pour la ligne de commande.

Arrêtez l'application OpenOffice.org, y compris le lanceur. Exécutez le fichier batch. La fenêtre MS-DOS va s'ouvrir, puis le document va s'afficher et, après un certain temps, le message « bavard a terminé » apparaîtra dans un petit panneau. Cliquez sur OK. Le document reste ouvert. L'exécution du batch ne reprend qu'à la fermeture d'OpenOffice.org. Vérifiez que le fichier `essaiMacro.txt` est bien écrit. L'option `-headless` donnerait le même résultat.

Si le document contenant la macro est déjà chargé, le fichier batch est un peu simplifié (LanceBavard4.bat) :

```
rem lancement de la macro
"C:\Program Files\OpenOffice.org 3\program\soffice.exe"
  ➤ "macro://Code01-02/Standard.monModule.bavard(D,20)"
pause
```

Chargez au préalable le document Code01-02.odt. Exécutez le fichier batch. La fenêtre DOS va s'ouvrir et, après un certain temps, le message « bavard a terminé » apparaîtra dans un petit panneau. Cliquez sur OK. Le batch reprend son exécution (et pause). Le fichier reste ouvert.

Lancer un script autre que Basic

Pour lancer un script d'un langage quelconque, il faut employer le Scripting Framework, qui a deux limitations quand on l'utilise depuis un batch :

- On ne peut lancer qu'un script « résident » (situé dans Mes macros ou dans Macros OpenOffice.org). Il est impossible de lancer un script d'un document.
- On ne peut pas transmettre d'argument au script.

L'argument du programme `soffice.exe` est une chaîne de caractères dont la structure générale est :

```
vnd.sun.star.script:alpha?language=beta&location=gamma
```

Respectez les majuscules et minuscules pour tous les caractères. Nous employons les mots `alpha`, `bêta`, `gamma` pour désigner des termes que nous allons expliciter.

- Le mot `bêta` est à remplacer par le nom du langage de script, exemples : `Java`, `JavaScript`, `BeanShell`, `Python`.
- Le mot `gamma` est à remplacer par le terme `user` si le script se trouve dans `Mes macros`, ou par le terme `share` si le script se trouve dans `Macros OpenOffice.org`.
- Le mot `alpha` est à remplacer par une séquence dépendant du langage.

Pour un script `JavaScript` ou `BeanShell` créé avec l'interface utilisateur, `alpha` se compose du nom de bibliothèque, d'un point, et du nom du fichier avec son extension.

```
HelloWorld.helloworld.bsh
```

Pour un script `Python`, `alpha` se compose du chemin vers le fichier `python`, en adressage relatif par rapport au répertoire `python/`, ensuite un caractère `$`, et enfin le nom de la fonction à appeler.

```
tata/exemple.py$maFonction
```

Quand un script est appelé par commande, il reçoit un argument. Cet argument n'a aucune utilité, mais il faut en tenir compte dans la déclaration de la fonction appelée (`Python` ou macro `Java`). Cet argument parasite n'a pas de conséquence sur les macros `JavaScript` ou `BeanShell`, car elles ne sont pas déclarées comme des fonctions.

Voici un exemple de ligne batch (`HelloJavaScript.bat`) appelant le script `JavaScript HelloWorld` qui se trouve dans `Macros OpenOffice.org`. Ouvrez un nouveau document `Writer` avant de lancer le batch car ce script écrit un texte dans le document `Writer` courant.

```
"C:\Program Files\OpenOffice.org 3\program\soffice.exe"  
➤ "vnd.sun.star.script:HelloWorld.helloworld.js?  
➤ language=JavaScript&location=share"
```

Basic peut lui aussi être lancé via le `Scripting Framework`, mais cela n'a pas d'intérêt, et implique une syntaxe encore différente.

Pour terminer sur ce point, signalons enfin que l'appel d'un script d'une extension installée utilise un argument plus complexe.

Sous Linux

Le principe est le même que sous MS-Windows. Les chemins suivront évidemment la syntaxe Unix, les arguments d'un shell seront obtenus par \$1 etc.

Il faut ajouter dans le fichier shell une redirection vers un terminal X. Faites attention aux guillemets.

```
export DISPLAY=":0.0"  
/usr/bin/soffice -headless "macro:///maBibli1.monModule.bavard("$1,  
15")"
```

Les extensions

De quoi s'agit-il ?

Le concept d'extension remplace et améliore considérablement le concept d'add-on des versions anciennes d'OpenOffice.org tout en restant compatible avec lui. Une extension est un fichier qui permet d'étendre les fonctionnalités d'OpenOffice.org (par exemple l'export au format LaTeX). Le nom d'un fichier extension a maintenant pour extension .oxt, mais les anciennes versions utilisent .zip ou .uno.pkg.

Vous pouvez créer des extensions vous-même, ou profiter de celles qui sont disponibles sur un site web. Soyez prudent dans ce dernier cas : une extension peut potentiellement, comme toute macro, provoquer des dégâts, intentionnels ou non. En installant une extension, vous acceptez implicitement son exécution, quel que soit le niveau de sécurité que vous avez choisi. La communauté OpenOffice.org a créé un site pour regrouper les extensions disponibles, ce qui facilite leur évaluation et réduit les risques :

<http://extensions.services.openoffice.org/>

Vous y trouverez de très nombreuses extensions téléchargeables, la plupart gratuites, certaines payantes. Bien entendu, la qualité et l'utilité des extensions est très variable.

Une extension peut se limiter à installer un fichier de configuration. C'est le cas des dictionnaires écrits pour OpenOffice.org.

Une extension est souvent un moyen pour diffuser facilement une bibliothèque de scripts sur différents postes de travail. Elle peut simplement contenir une seule macro Basic, ou un ensemble de macros, mais ce n'est pas limitatif : tous les langages de script que nous avons vus peuvent être utilisés, ainsi que des bibliothèques de code objet compilé. Si nécessaire, une extension peut comporter des parties codées avec différents langages de programmation et plusieurs bibliothèques.

Une extension peut apporter un composant UNO, créé en Java, C++ ou Python. Ce composant peut être un nouveau service qui enrichit l'API, ou un add-in pour Calc, ou encore un add-in pour diagrammes.

En pratique, les extensions élaborées nécessitent de modifier l'interface utilisateur afin de les rendre plus conviviales. Aussi, une extension peut :

- ajouter une barre d'outils avec plusieurs boutons ayant des icônes spécifiques ;
- ajouter ou supprimer des boutons dans des barres d'outils existantes ;
- ajouter une entrée ou un sous-menu dans l'entrée Outils>Add-ons ;
- ajouter ou supprimer de nouvelles entrées ou même une arborescence de sous-menus dans les menus principaux (Fichier, Édition, Affichage, etc.) ;
- ajouter des pages qui seront intégrées au système d'aide OpenOffice.org ; la page correspondant au contexte étant appelée comme pour une fonctionnalité de l'application.

Ces ajouts et modifications peuvent être propres à chaque application d'OpenOffice.org concernée (Writer, Calc, Writer-HTML, etc). Par exemple, un bouton n'apparaît que pour Calc, un autre pour Writer, Calc, Draw et Impress.

Ajoutons que, pour permettre une diffusion internationale, tout le système d'extension est multilingue : il est possible de préparer tous les textes (menus, pages d'aide, etc.) pour plusieurs langues. S'ils sont disponibles, les textes apparaîtront automatiquement dans la langue de l'interface utilisateur.

D'autres mécanismes facilitent la gestion d'extensions d'envergure professionnelle :

- un identifiant unique, spécifique à l'extension (pour éviter les conflits de noms entre auteurs d'extensions) ;
- un numéro de version ;
- un nom « commercial » ;
- une icône symbolisant l'extension dans le gestionnaire des extensions ;
- un texte descriptif affiché par le gestionnaire des extensions ;
- un texte de la licence du produit, affiché à l'installation ;
- un contrôle d'adéquation entre l'extension et le poste utilisé (version d'OpenOffice.org, plate-forme matérielle) ;
- l'ajout de pages d'options spécifiques dans le menu Outils>Options ;
- un système de recherche de mise à jour vers une version plus récente de l'extension ;
- des liens vers le site de l'éditeur du logiciel ;
- un lien vers un site affichant les notes de livraison.

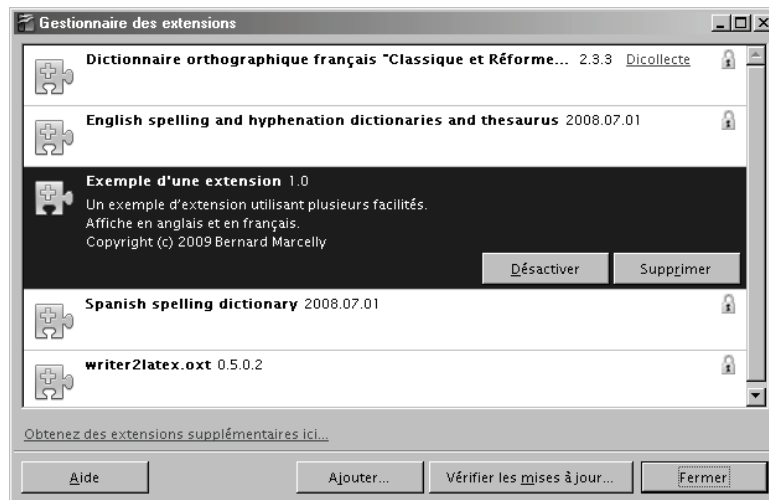
Installer une extension

Quand une extension est installée sur un seul poste, les scripts sont visibles dans la section Mes Macros. Dans une installation OpenOffice en réseau, le responsable réseau a la possibilité d'installer une extension en mode partagé, accessible depuis tous les postes. Les scripts de l'extension apparaissent alors dans Macros OpenOffice.org. Sur un système multi-utilisateur comme Windows XP Home, depuis OpenOffice.org version 3.0 vous pouvez installer l'extension pour un utilisateur ou pour tous.

Installation depuis OpenOffice.org

Le gestionnaire des extensions permet d'ajouter ou de supprimer très facilement des extensions. Vous y accédez par le menu Outils>Gestionnaire des extensions. Le panneau obtenu (figure 1-22) liste les extensions déjà installées. Celles comportant un petit cadenas sont installées pour tous les utilisateurs.

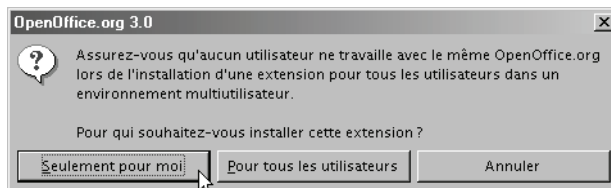
Figure 1-22
Le gestionnaire
des extensions



Il suffit de cliquer sur le bouton Ajouter et de choisir le fichier extension. Le panneau qui apparaît (figure 1-23) vous donne le choix de l'installer pour un seul utilisateur ou pour tous les utilisateurs. Le même panneau sert à supprimer une extension ou à voir s'il existe des mises à jour pour les extensions installées.

À partir de la version 3.0, il est possible d'installer une extension en la lançant depuis un gestionnaire de fichiers, par exemple avec un double-clic.

Figure 1-23
Installer une extension



Installation depuis la ligne de commande

Le responsable informatique qui préfère l'installation en ligne de commande, lancera la commande `unopkg` qui se trouve dans le sous-répertoire `program/` du répertoire d'installation d'OpenOffice.org. Dans ces exemples des principaux modes de lancement nous n'avons pas écrit le chemin complet du fichier `monExtension.oxt`.

```
*** ajouter une extension pour un utilisateur
unopkg add monExtension.oxt
*** ajouter une extension pour tous les utilisateurs
unopkg add --shared monExtension.oxt
*** supprimer une extension à partir de son identifiant unique
unopkg remove xxx.yyyy.zzz.monExtension
unopkg remove --shared xxx.yyyy.zzz.monExtension
*** lancer unopkg en mode interactif
unopkg gui
*** interactif, en désignant le fichier (OOo 3.0 minimum)
unopkg gui monExtension.oxt
*** lister les options de unopkg
unopkg -h
```

Attention, pendant l'installation pour tous les utilisateurs le responsable informatique doit s'assurer que personne d'autre n'utilise OpenOffice.org.

Concevoir une extension

Une extension se présente sous la forme d'une archive Zip qui regroupe différents fichiers. Réaliser une extension comportant toutes les possibilités que nous avons décrites représente parfois un effort plus important que le codage du script, en particulier pour une extension multilingue avec pages d'aide. Une telle extension comporte souvent des dizaines de fichiers. La plupart d'entre eux sont des fichiers XML dont les contenus sont codifiés selon des syntaxes précises.

Certains développeurs créent une extension en se basant sur une extension existante. Pour cela, il faut :

- 1 dézipper l'archive ;
- 2 modifier les fichiers XML manuellement avec un éditeur ;

- 3 ajouter et remplacer les fichiers propres à la nouvelle extension (bibliothèque Basic, fichiers images, etc) ;
- 4 et enfin zipper le dossier.

POUR LES EXPERTS Documentation officielle

Les extensions sont décrites techniquement dans le *Developer's Guide*, section *Extensions*. Ces pages, en anglais, sont disponibles à cette adresse :

► <http://wiki.services.openoffice.org/wiki/Documentation/DevGuide/Extensions/Extensions>

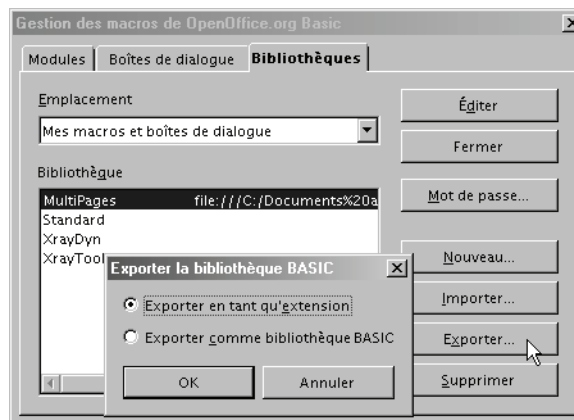
En pratique créer ainsi une extension est pénible et long si on veut profiter des possibilités existantes. Si vous ne connaissez pas bien la syntaxe des différents sous-fichiers de l'extension, l'échec est assuré. Nous présentons ici trois manières plus efficaces de créer une extension.

Exporter une extension minimale

Un codage Basic réalisé dans une bibliothèque Basic autre que Standard est exportable en tant qu'extension simple, en cliquant sur un bouton dans le gestionnaire de macros Basic, comme sur la figure 1-24. Une fois l'export effectué, supprimez la bibliothèque Basic existante, puis installez l'extension, ce qui rétablira la bibliothèque.

Figure 1-24

Créer une extension à partir d'une bibliothèque Basic



Ainsi, il est facile d'ajouter ou de supprimer une bibliothèque de macros pour différents utilisateurs. Ceux-ci pourront les employer dans leurs codages ou configurer manuellement leur poste pour les appeler d'un clic sur un bouton de barre d'outils, ou par un raccourci clavier. En contrepartie on n'utilise aucune des possibilités avancées décrites plus haut.

L'outil BasicAddonBuilder

L'outil Open Source BasicAddonBuilder a été élaboré par Paolo Mantovani. Il sert à créer une extension à partir d'une bibliothèque de macros Basic, et à ajouter une barre d'outils et des menus. Un Assistant très bien conçu facilite sa mise en œuvre (voir les figures 1-25 et 1-26).

Figure 1-25
BasicAddonBuilder, étape 1

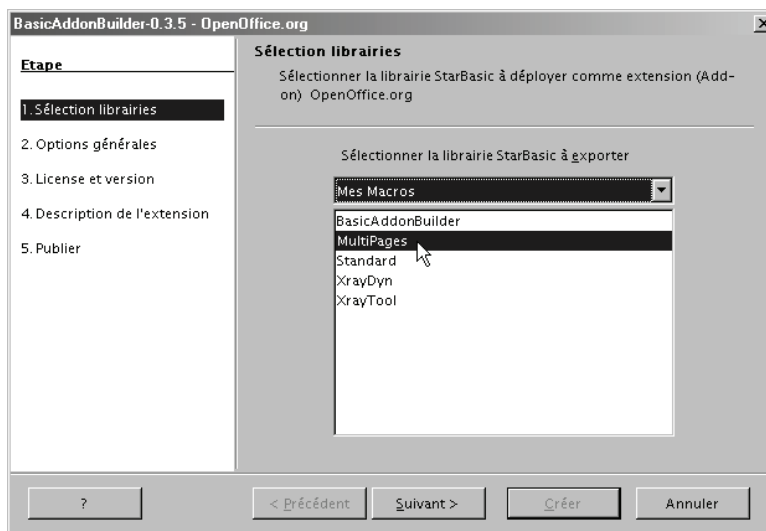
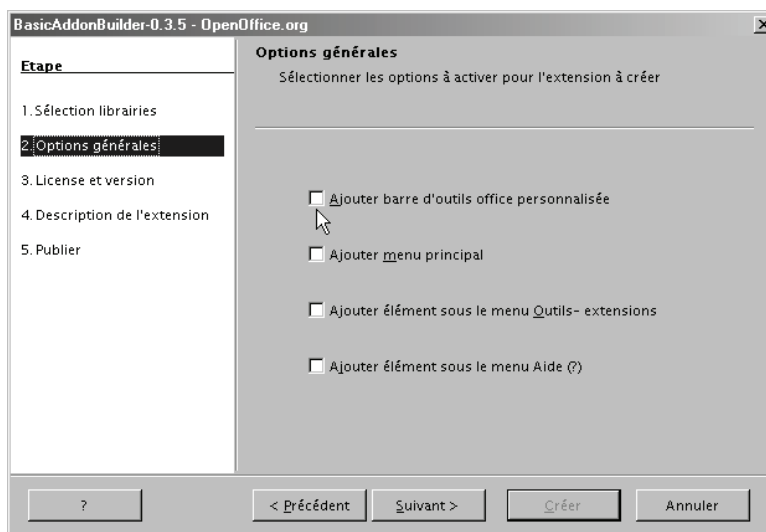


Figure 1-26
BasicAddonBuilder, étape 2



Pour la petite histoire, BasicAddonBuilder est une extension, réalisée en plusieurs langues, dont le français. Cet outil est idéal pour une extension comportant des macros Basic, une barre d'outils et quelques menus. L'inconvénient est que, dans la version actuelle, si vous voulez modifier quelque chose vous devez ré-exécuter toutes les étapes de l'Assistant sans vous tromper, ce qui peut devenir pénible.

Pour télécharger BasicAddonBuilder, rendez-vous à la page :

http://wiki.services.openoffice.org/wiki/Extensions_Packager

L'outil Extension Compiler

Un des auteurs de ce livre a créé l'outil Open Source Extension Compiler, seulement disponible en anglais. Il permet de réaliser tous types d'extensions, pas seulement Basic, et supporte tous les mécanismes connus, ce qui implique en contrepartie un effort de compréhension plus important. Pour le télécharger, rendez-vous à l'adresse suivante :

http://wiki.services.openoffice.org/wiki/Extensions_Packager

L'outil se présente sous la forme d'un modèle de document Writer (extension .ott). En l'ouvrant par un double-clic, vous obtenez un nouveau document que vous allez personnaliser. Commencez par le sauver sous le nom de votre future extension, et dans un répertoire dédié. Dans ce répertoire (ou dans des sous-répertoires), vous devrez mettre toutes les bibliothèques que votre extension apporte (Basic, Python, etc), les fichiers images, et tous autres fichiers spécifiques qu'elle utilise. Ainsi, tous les éléments nécessaires à la création de l'extension se trouvent dans le répertoire dédié.

Ce document est à la fois un outil réalisé en Basic, la documentation de l'outil, et le support sur lequel vous décrivez votre extension. Vous allez écrire dans ce document :

- Une succession linéaire de macro-instructions (en fait des appels de sous-programmes Basic) dans une macro déjà préparée ; le document décrit la syntaxe d'utilisation de chaque macro-instruction.
- Si vous le souhaitez, un ou plusieurs textes destinés à être affichés par l'extension (descriptions, licences, pages d'aide), en une ou plusieurs langues pour chacun.

Voici à titre d'exemple le début des instructions Basic nécessaires pour compiler une extension multilingue.

```
beginDescription("org.bmarcelly.ExtExample", "1.0", "user")
  beginDependencies
    setOoDependency("2.4", "OpenOffice.org 2.4.0 minimal")
  endDependencies
  setExtensionDescription("en, fr")
  setLicense("en, fr", "", "user", True)
  setDisplayName("en", "Example of an extension")
  setDisplayName("fr", "Exemple d'une extension")
```

```

    setHelp("en, fr")
endDescription

beginAnnexes
    useLibrary("Basic", "basTest4/")
endAnnexes

beginAddonUI
    beginAddonMenu
        beginMenu
            beginTitles("Writer")
                setTitle("Calling scripts", "en")
                setTitle("Appel de scripts", "fr")
            endTitles
            beginMenuItems
                beginCommand
                    beginTitles
                        setTitle("Basic : display time", "en")
                        setTitle("Basic : afficher l'heure", "fr")
                    endTitles
                    setURL("Basic", "basTest4", "Module1", "WhatTimeIsIt")
                endCommand
                beginCommand
                    beginTitles
                        setTitle("Basic : Hello", "en")
                        setTitle("Basic : Hello", "fr")
                    endTitles
                    setURL("Basic", "basTest4", "Module1", "HelloBas")
                    setImage("Images/greenPage")
                endCommand
                addSeparator
            endMenuItems
        endMenu
    endAddonMenu
end
rem --- etc... ---

```

La compilation s'effectue en lançant l'exécution du programme Basic. Tous les fichiers de configuration nécessaires sont créés, l'ensemble est zippé, et le fichier extension est obtenu après quelques secondes. S'il détecte une erreur grave, le compilateur fournit un message diagnostic et stoppe. Il suffit alors de corriger la ligne en cause et de relancer la compilation.

Le compilateur signale par un message d'avertissement les incompatibilités potentielles entre les options choisies et le numéro de version minimal d'OpenOffice.org accepté. En effet, comme les fonctionnalités liées aux extensions ont été introduites au fil des versions successives d'OpenOffice.org, il faut éviter qu'un utilisateur n'installe une extension sur une version incompatible.

Si vous décidez par la suite d'améliorer votre extension, il vous suffira de reprendre le document, d'y apporter les modifications nécessaires et de le recompiler pour obtenir la nouvelle version de l'extension.

Conclusion

Après ce tour d'horizon des scripts dans OpenOffice.org, il est temps d'aborder Basic, le langage le plus couramment utilisé, ainsi que son environnement de développement intégré. Ceci nous permettra de voir le contenu d'une macro, de l'éditer et de l'exécuter.

DEUXIÈME PARTIE

Le langage OOoBasic

Il existe de nombreuses variantes du langage Basic. Ils se ressemblent, mais chacun a ses particularités. L'application OpenOffice.org intègre un langage Basic spécifique, ainsi qu'un environnement de développement pour créer et mettre au point vos programmes en OOoBasic.

Même si vous connaissez déjà un autre Basic, par exemple Visual Basic™ qui lui est proche, prenez le temps de parcourir les chapitres de cette partie. En cas de problème d'exécution, relisez-la, car elle contient bien des détails importants et pas toujours indiqués dans la documentation officielle.

À l'intention des néophytes, nous indiquons aussi quelques règles de bonne programmation, non pas pour vous enlever le plaisir de programmer, mais au contraire pour vous éviter échecs et découragements.

Introduction au Basic

Le Basic OpenOffice.org est le langage que nous utilisons dans les exemples de ce livre. Après l'avoir situé parmi ses confrères, nous verrons l'environnement de développement intégré (EDI en français ou IDE en anglais), outil essentiel pour écrire et mettre au point aisément les codages en Basic. Nous commencerons par nous familiariser avec ses différents menus et comprendre leur utilité pour développer des programmes Basic. Nous aborderons alors les règles de syntaxe de ce langage, ainsi que quelques bonnes pratiques de programmation.

Premier aperçu du langage OpenOffice.org Basic

Le langage Basic de la suite OpenOffice.org est parfois appelé en abrégé OOoBasic afin de le distinguer des autres langages Basic, et nous reprendrons ce terme. Absorbée par Oracle, la société Sun préférait le terme StarOffice Basic ou StarBasic, car elle diffusait la suite StarOffice™ qui était basée sur OpenOffice.org.

OOoBasic, langage de script d'OpenOffice.org

Son nom rappelle un des plus anciens langages, Basic, conçu à l'origine pour des débutants en programmation disposant de moyens informatiques parfois très limités. En réalité OOoBasic n'a plus qu'un lointain rapport avec son ancêtre. Bien qu'il reste d'un abord simple, OOoBasic est un langage de programmation moderne permettant de développer des applications sophistiquées. C'est un langage de programmation

modulaire : fini les numéros de lignes, les GOTO et la programmation « spaghetti » ; les macros sont regroupées en modules et en bibliothèques réutilisables.

OOoBasic est conçu pour écrire, lire et modifier des documents OpenOffice.org en utilisant l'API (interface de programmation d'application) propre à OpenOffice.org. Il est entièrement intégré à OpenOffice.org, et ce dernier comporte un environnement de développement intégré (EDI) servant à écrire les macros et à les mettre au point. L'exécution du codage est immédiate, sans nécessiter de compilation explicite. Ceci en fait une base très pratique (et gratuite) pour s'initier à la programmation ou pour vérifier rapidement un codage utilisant une fonction de l'API.

OOoBasic n'est pas un langage de programmation autonome : il a besoin d'OpenOffice.org pour s'exécuter. C'est un langage orienté vers l'automatisation de besoins de bureautique, il ne remplace pas les langages généralistes comme C++, Java, Delphi, Python et autres. Les langages généralistes offrent des structures de données plus élaborées et généralement des bibliothèques de fonctions utilitaires dans divers domaines. Ils seront donc plutôt employés dans un cadre applicatif plus global que les documents OpenOffice.org. Il est bien plus facile de recourir à l'API par le biais d'OOoBasic (ou Python, Delphi) qu'avec Java ou C++. En contrepartie, seuls ces derniers permettent d'intervenir au cœur de l'API pour y ajouter ou modifier des fonctionnalités. Il serait donc faux de croire que « qui peut le plus peut le moins » : une très bonne connaissance de l'informatique, du concept UNO et de l'anglais écrit sont nécessaires pour cela.

Les macros OOoBasic peuvent être intégrées soit dans un document, soit dans OpenOffice.org. Elles sont transportables d'un ordinateur à un autre, et même distribuables sur un réseau de machines. Elles sont exécutables sur tous les systèmes d'exploitation reconnus par OpenOffice.org, ce qui constitue un atout important pour les échanges de documents.

Il existe divers moyens pour déclencher une macro : manuellement, en cliquant sur un bouton d'une barre d'outils, en cliquant sur un bouton dans un document, en utilisant une entrée de menu, avec un raccourci clavier, lors de divers événements comme le chargement ou la fermeture du document, ou automatiquement de manière invisible.

Les manipulations de documents OpenOffice.org se font normalement à travers l'API, quel que soit le langage utilisé. OOoBasic n'y fait pas exception, et malgré son apparence simplicité, il est capable d'utiliser pratiquement toutes les fonctionnalités offertes par l'API OpenOffice.org. Ainsi, les concepts décrits dans cet ouvrage sont largement réutilisables avec un autre langage.

OOoBasic et VBA

Lorsqu'un utilisateur ou une entreprise décide d'utiliser OpenOffice.org à la place de la suite MS-Office, la conversion des documents Word et Excel existants est effectuée généralement correctement ; en revanche, les macros Visual Basic for Application (VBA™) dans ces documents ne sont pas exécutables telles quelles avec OOoBasic. La raison est qu'il n'y a pas d'équivalence simple entre les deux langages car les réalisations logicielles sous-jacentes sont différentes bien que les possibilités soient comparables. Plus précisément, VBA intègre à la fois un langage de programmation Basic et des fonctions d'accès à l'API MS-Office présentées sous la forme de pseudo-instructions.

Les instructions du langage OOoBasic lui-même sont très similaires et souvent identiques à celles du langage Visual Basic. Il existe toutefois de nombreuses différences de détails, qui peuvent nécessiter une modification de l'algorithme. Si vous connaissez déjà VBA, ne faites pas de supposition sur OOoBasic : lisez attentivement la description du Basic dans ce livre, vous découvrirez des nombreuses différences, souvent subtiles, mais dont vous devez tenir compte.

L'équivalent des fonctions d'accès à l'API MS-Office est l'ensemble des services de l'API d'OpenOffice.org, accessibles depuis différents langages de programmation. Contrairement à l'éditeur de code de VBA, l'éditeur de OOoBasic ne propose pas l'auto-complétion, c'est-à-dire la proposition en cours de frappe des propriétés et méthodes des objets de l'API. L'outil Xray, décrit dans l'annexe A, permet de les afficher, mais seulement en ajoutant une instruction dans la macro à exécuter. Les objets de l'API OpenOffice.org sont souvent très différents, et seule une bonne connaissance des deux principes de programmation permettra d'écrire un algorithme équivalent, lorsque c'est possible. L'effort intellectuel de conversion pouvant être assez important, il est souhaitable de se demander si le document est encore utile, et dans l'affirmative, de déterminer quel est le but à atteindre avec des macros, plutôt que tenter une conversion instruction par instruction ou macro par macro.

Le Basic avec un goût de VBA

Avant d'aborder la description du Basic OpenOffice, il est nécessaire de préciser de quoi nous allons parler. En effet, au fil des évolutions d'OpenOffice.org, différentes variantes sont apparues dans le but de permettre une meilleure compatibilité avec VBA de Microsoft.

Dans les premières versions d'OpenOffice.org 2, Sun Microsystems avait introduit deux modificateurs permettant une meilleure compatibilité avec VBA : la déclaration `Option Compatible`, et l'instruction `CompatibilityMode`.

À la même époque, Novell a démarré un projet ambitieux visant à exécuter directement des macros VBA existantes. Les résultats de ce projet sont intégrés dans les développe-

ments OpenOffice.org à partir de la version 2.4.1 ainsi que dans LibreOffice. Les modificateurs de Sun sont remplacés par une seule déclaration `Option VBASupport` en tête du module, avant toute déclaration de macro ou de donnée.

Option VBASupport 1

Cette option a quatre effets :

- Dans le module considéré, l'analyseur syntaxique est légèrement modifié pour traiter la syntaxe VBA. Par exemple, l'appel d'une routine avec plusieurs arguments ne doit pas les englober de parenthèses.
- Dans le module considéré, certains objets VBA sont reconnus, comme `ActiveSheet`, `ThisWorkbook`, etc.
- Certaines instructions VBA deviennent disponibles (par exemple `InStrRev`) dans ce module.
- Pour une meilleure compatibilité, le comportement de Basic change pour certaines instructions (`Mid`, `Dir`, `RmDir`, par exemple). Ceci peut causer des comportements bizarres si votre code utilise à la fois des modules avec `VBASupport` et des modules `OOoBasic`.

En fonction des options du menu `Outils>Options>Chargement/Enregistrement>Propriétés VBA`, les macros d'un document MS-Office sont soit ignorées, soit mises en commentaires dans des modules de macros. À partir de la version 3.0, une option permet de rendre exécutables les macros d'un document Excel, en utilisant le mécanisme de compatibilité développé par Novell. Il en sera bientôt de même pour les documents Word.

En pratique, il existe de très nombreuses différences mineures entre `OOoBasic` et VBA qui empêchent l'exécution directe de macros VBA. Elles ne sont pas toujours documentées par Microsoft, mais elles sont utilisées. Par exemple, le mot-clé `Name` peut être employé comme nom de variable dans VBA, mais pas dans `OOoBasic` qui le considère comme un mot réservé. Novell est ainsi conduit à multiplier les « petites » améliorations du mode de compatibilité à mesure qu'il découvre des problèmes en analysant des macros VBA. Ces modifications sont introduites au fil des versions mineures d'OpenOffice.org.

Il ne faut pas s'illusionner sur cette option de compatibilité : les implémentations de MS-Office et de VBA d'une part, et de `OOoBasic` et de l'API OpenOffice d'autre part, diffèrent dans leur principe et dans leur logique. Une parfaite compatibilité est impossible. Le seul intérêt de cette option est de permettre d'exécuter avec peu ou pas de modifications des macros VBA simples, et de réduire ainsi le travail de portage, ce qui est déjà très appréciable. Cependant, un programmeur devra effectuer manuellement la conversion des macros complexes ou trop éloignées des principes

OpenOffice. Il faudra alors repenser les macros et les récrire en ayant une parfaite connaissance de l'API OpenOffice.

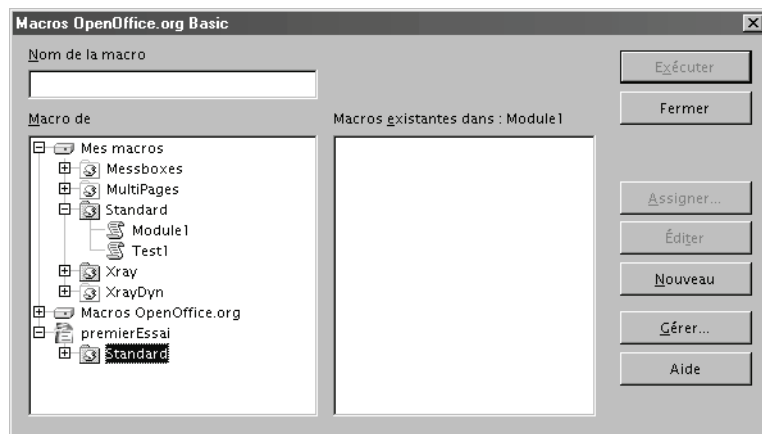
C'est pourquoi nous n'utiliserons pas l'option de compatibilité VBA. Ce livre ne décrit que le Basic spécifiquement OpenOffice.org, encore appelé OOoBasic, et nous l'utiliserons pour montrer les possibilités de l'API propre à OpenOffice.org.

Premiers pas dans l'environnement de développement Basic

Anatomie de la fenêtre Macros Basic

Créez un nouveau document Writer, sauvegardez-le et gardez-le ouvert. Ouvrez le menu Outils>Macros>Gérer les macros>OpenOffice.org Basic. Vous obtenez le panneau reproduit à la figure 2-1.

Figure 2-1
Le premier panneau
Macros Basic



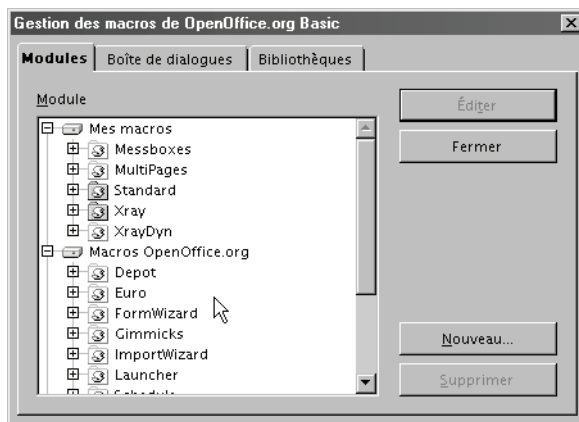
L'aspect exact de l'arborescence au-dessous de Macro de dépend des macros qui sont disponibles dans votre environnement OpenOffice.org. Vous aurez toujours une racine Mes macros et une racine Macros OpenOffice.org. Ces deux racines contiennent des macros disponibles tant qu'OpenOffice.org est chargé sur votre ordinateur. Il est important de savoir qu'elles correspondent à un seul conteneur de bibliothèques, appelé soffice dans les versions précédentes. Nous utiliserons ce terme à différents endroits de ce livre, car il est encore utilisé dans la documentation technique.

Le panneau affiche aussi une racine pour chaque document ouvert : chacune est un conteneur de bibliothèques de macros. Vous remarquerez un élément mis en valeur dans la figure, Standard, qui est une branche du document premierEssai.odt ; cette

bibliothèque par défaut est toujours présente, même vide. De même, il existe une bibliothèque nommée Standard dans Mes macros.

Cliquez sur le bouton Gérer.... Vous obtenez un autre panneau, Gestion des macros de OpenOffice.org Basic qui comporte trois onglets : Modules, Boîtes de dialogue, et Bibliothèques (voir la figure 2-2). Restez sur l'onglet Modules.

Figure 2-2
Onglet Modules :
les bibliothèques
de codage Basic



En utilisant l'ascenseur et en cliquant sur les +, explorez l'arborescence. Dans notre exemple, nous avons développé le premier niveau de Macros OpenOffice.org. Nous avons ainsi affiché la liste de ses bibliothèques de codage Basic.

Examinez encore l'arborescence de la figure 2-2 : la bibliothèque Euro est de couleur blanche pour signifier qu'elle n'est pas chargée actuellement. Pour exécuter une macro d'une bibliothèque, celle-ci doit être chargée.

Double-cliquez sur l'icône de la bibliothèque Euro. Une arborescence apparaît, comme sur la figure 2-3, et l'icône devient jaune. Vous venez de charger la bibliothèque Euro. Remarquez qu'une fois chargée, il n'est pas possible de la décharger. En fait, il faudrait fermer complètement OpenOffice.org pour cela, ou fermer le document si la bibliothèque appartient à un document.

L'arborescence de la bibliothèque Euro nous montre des noms de modules (on peut effectivement choisir des noms plus parlants que Module1, Module2, etc.). Le codage Basic est mémorisé dans ces modules.

Cliquez maintenant sur l'onglet Boîtes de dialogue. Cette page n'affiche que les modules de dialogue existant dans chaque bibliothèque (voir figure 2-4). Chaque module contient une boîte de dialogue.

Les dialogues affichent des informations et posent des questions à l'utilisateur pour obtenir des réponses de manière semblable aux panneaux d'OpenOffice.org. Nous

Figure 2-3
Les modules
de la bibliothèque Euro

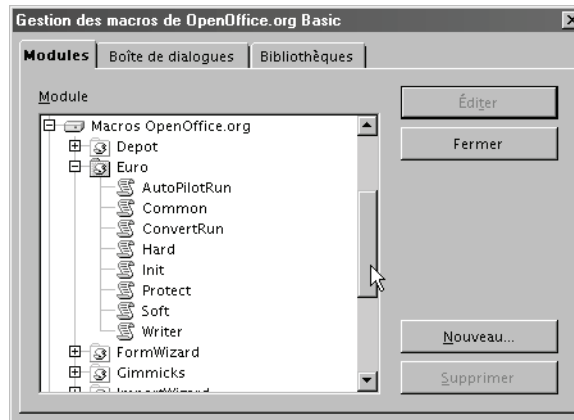
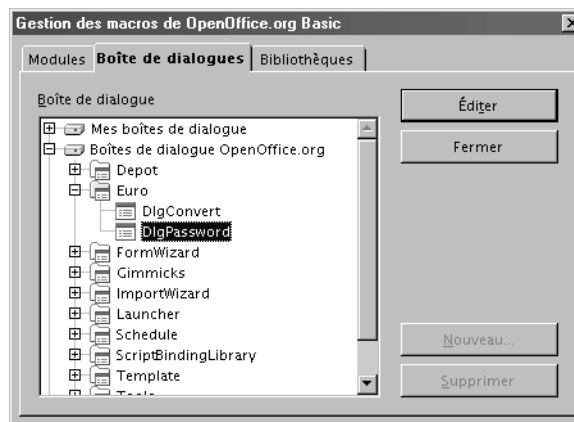


Figure 2-4
Onglet Dialogue :
les bibliothèques de dialogue



expliquerons comment créer des boîtes de dialogue au chapitre 11. Il est important de savoir que les dialogues ne sont pas réservés au langage Basic, ils peuvent être utilisés depuis d'autres langages de script. Cependant, pour des raisons d'implémentation, il existe encore une bibliothèque de modules Basic, même vide, pour chaque bibliothèque de dialogue. L'EDI de Basic affiche les modules de codage et de dialogue. Notez les noms de racines : Mes boîtes de dialogue et Boîtes de dialogue OpenOffice.org correspondent aux dialogues disponibles tant qu'OpenOffice.org est chargé sur votre ordinateur. Un document peut lui aussi contenir des dialogues.

Passons maintenant à l'onglet Bibliothèques du panneau Gestion de macros. Il est reproduit à la figure 2-5.

Ce panneau affiche initialement les bibliothèques de Mes macros et boîtes de dialogue. La liste déroulante au-dessus permet de choisir les bibliothèques communes Macros