Introduction

Visual Basic pour Applications, VBA, est la solution de programmation proposée avec les applications de la suite Office. La connaissance de VBA permet à l'utilisateur d'Excel de tirer pleinement profit du tableur de Microsoft en en développant les capacités et les fonctionnalités pour ses besoins spécifiques. Maîtriser Visual Basic pour Applications, c'est à coup sûr améliorer grandement sa productivité et celle de son entreprise.

L'intégration dans Excel de Visual Basic pour Applications, un *environnement de développement intégré* complet et professionnel, remonte à sa version 97. Depuis, Microsoft a confirmé sa volonté de faire de VBA un élément à part entière des applications Office et l'a progressivement proposé avec l'ensemble des applications de sa suite bureautique. Visual Basic pour Applications constitue aujourd'hui un langage et un environnement stables et pérennes. Office 2013 intègre la version 7.1 de Visual Basic (les versions XP, 2003 et 2007 d'Office intégraient Visual Basic 6.3, et la version 2010 intégrait la version 7).

Cet ouvrage traite de la programmation des versions 97 à 2013 d'Excel. Sauf exception signalée, les explications et les exemples proposés sont valides pour toutes versions d'Excel. En effet, d'une version à l'autre, il n'y a pas eu de révolution. Le modèle d'objets s'est affiné et les nouvelles fonctions d'Excel, apparues au cours des différentes versions du logiciel, peuvent également être manipulées *via* la programmation VBA. Cependant, le langage, la gestion des programmes, l'environnement et les outils au service du développeur – bref, tout ce que vous devez savoir pour programmer Excel et que cet ouvrage se propose de vous apprendre – restent inchangés d'une version à l'autre.

Donc, que vous utilisiez encore Excel 2003 ou que vous soyez passé à la version 2013, sachez que vous pourrez appliquer les connaissances que vous aurez acquises lors de la lecture de ce livre quand vous migrerez d'une version à l'autre d'Excel. Mieux, les programmes développés pour Excel 97 fonctionnent avec toutes les versions ultérieures du tableur et, dans la très grande majorité des cas, les programmes développés dans Excel 2013 devraient fonctionner avec les versions antérieures.

Dans cet ouvrage, vous découvrirez les différentes méthodes de création de projets VBA pour Excel, Visual Basic (le langage de programmation proprement dit) et les outils de développement et de gestion intégrés de Visual Basic pour Applications. Votre initiation à la programmation VBA se fera au moyen d'exemples de programmes détaillés et commentés.

Vous rencontrerez le terme projet tout au long de cet ouvrage. C'est ainsi que l'on nomme un ensemble de programmes développés avec Visual Basic pour Applications.

VBA 7 : 64 bits *vs* 32 bits

La suite Office est proposée en deux versions : 32 bits et 64 bits. Ces deux versions sont pour ainsi dire indifférenciables : seule la gestion de la mémoire varie d'une version à l'autre, autorisant la manipulation de fichiers nettement plus volumineux avec la version 64 bits. Cependant, Microsoft recommande l'installation de la version 32 bits, y compris sur un système d'exploitation 64 bits, notamment pour des raisons de compatibilité des compléments (comme les macros VBA) avec les versions précédentes. C'est d'aillleurs la version 32 bits qui est installée par défaut, et les utilisateurs souhaitant installer la version 64 bits doivent parcourir le CD afin d'exécuter le fichier d'installation correspondant.

Conséquence pour le développement de macros Excel : la version 7 de VBA est maintenant une version 64 bits, qui intègre le support d'un nouveau type de données permettant la manipulation des pointeurs (les pointeurs permettent la manipulation des API Windows). Cet ouvrage n'abordant pas la manipulation des API – notions réservées aux programmeurs chevronnés – cette nouveauté n'a aucune incidence sur la validité de ce que vous explique ce livre. Les concepts et techniques de programmation que vous apprendrez ici sont donc compatibles avec la version 64 bits d'Office comme avec les versions 32 bits d'Office, sans qu'il soit nécessaire d'adapter le code.

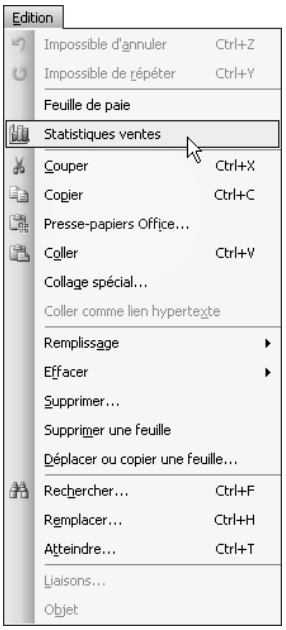
VBA, pour quoi faire ?

Excel offre des possibilités très étendues. Pourtant, quelle que soit la puissance des fonctions d'Excel, elles ne peuvent répondre à toutes les situations. La programmation VBA est la solution de personnalisation offerte par Excel, afin d'ajouter des caractéristiques, des fonctions et des commandes qui répondent précisément à vos besoins.

La programmation VBA peut être définie comme la *personnalisation d'un logiciel afin de s'assurer gain de temps, qualité des documents et simplification des tâches complexes ou fastidieuses.* Voici quelques exemples de ce que permettent les programmes VBA :

- **Combiner un nombre indéterminé de commandes.** Nous sommes souvent amenés à répéter ou à associer certaines commandes plutôt que d'autres et à ignorer certaines fonctionnalités en fonction de l'usage personnel que nous avons d'un logiciel. VBA permet d'associer un nombre illimité de commandes à une seule. Vous pouvez ainsi ouvrir simultanément plusieurs documents Excel stockés dans des dossiers ou sur des serveurs différents, y insérer des données spécifiques et leur appliquer des mises en forme adaptées, en exécutant une seule commande créée en VBA.
- **Ajouter de nouvelles commandes et de nouvelles fonctions à Excel.** VBA permet de créer de nouvelles commandes et d'ajouter des fonctions au tableur – par exemple une fonction personnalisée qui permet de calculer les taxes à retenir sur un salaire (ou, mieux, les primes à y ajouter), etc. Vous pouvez, en outre, attacher vos programmes VBA à des raccourcis clavier, à des icônes et à des commandes de menu afin d'en améliorer l'accessibilité.

Figure 0.1
VBA permet de personnaliser l'interface des applications Office en y ajoutant icônes et commandes de menus.



- **Automatiser des actions répétitives.** Nous sommes parfois amenés à répéter certaines opérations plusieurs fois sur un même document ou à réitérer des traitements spécifiques. Un programme VBA peut, par exemple, mettre en forme des cellules dans un classeur Excel, effectuer des séries de calculs, etc.
- **Modifier et améliorer les commandes d'une application.** Les commandes Excel ne sont pas toujours adaptées à nos besoins ou présentent parfois des limitations gênantes. Un programme VBA peut modifier, brider ou compléter les commandes d'une application. Vous pouvez ainsi intégrer dans un tableau le nom de l'utilisateur, le nombre de pages imprimées et l'imprimante utilisée chaque fois qu'une impression est lancée à partir d'Excel.
- **Faire interagir les différentes applications Office.** Un programme VBA peut exploiter des données issues de fichiers générés par d'autres programmes et interagir avec ceux-ci de façon transparente pour l'utilisateur. Vous pouvez ainsi créer une commande qui envoie automatiquement le classeur Excel ouvert en fichier joint dans un mail Outlook à des destinataires définis ou qui génère un rapport Word à partir de données Excel et l'imprime.
- **Créer des interfaces personnalisées.** Les programmes VBA peuvent ramener des tâches complexes à la simple information de champs dans des boîtes de dialogue personnalisées pour l'utilisateur final, simplifiant ainsi considérablement le travail de celui-ci, tout en vous assurant qu'aucun oubli ou fausse manipulation n'aura lieu.

Visual Basic pour Applications permet le développement de solutions adaptées à vos besoins. Les outils que vous apprendrez à manier vous permettront de développer des programmes simples, sans écrire la moindre ligne de code, comme des programmes complets intégrant une interface utilisateur adaptée.

La fonction d'un programme VBA peut être d'automatiser une tâche répétitive. Mais vous pouvez aussi créer très vite un petit programme VBA pour faire face à une nécessité immédiate ; par exemple, afin de généraliser un traitement exceptionnel à l'ensemble d'un document.

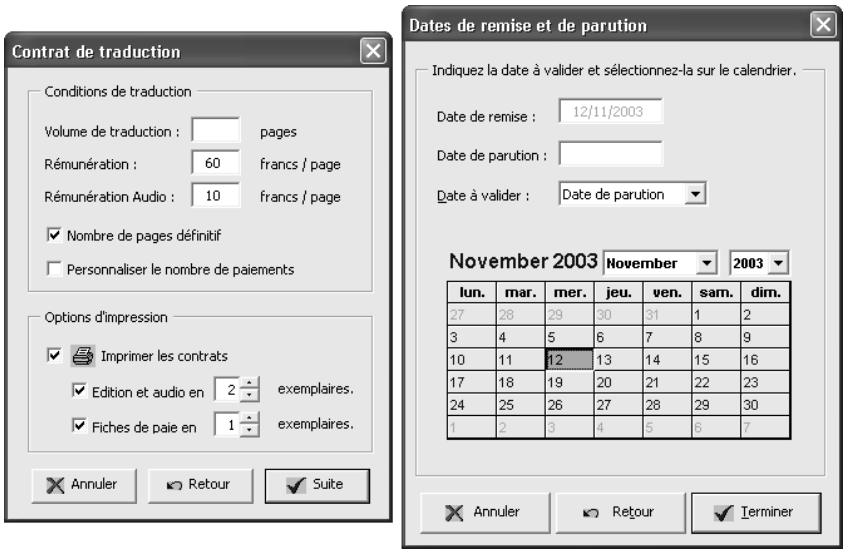


Figure 0.2
Visual Basic pour Applications vous permet de développer des interfaces utilisateur évoluées.

Des programmes

Les projets *VBA* sont des programmes ou *macros* écrits dans le langage Visual Basic. Si vous ne possédez aucune expérience préalable de programmation, ne vous inquiétez pas : cet ouvrage aborde le développement de projets VBA à travers l'enregistrement de macros. Lorsque vous l'activez, l'Enregistreur de macro mémorise chacune de vos actions. C'est votre programmeur personnel : vous utilisez simplement les commandes d'Excel et il se charge de traduire les actions exécutées en instructions Visual Basic. Il vous suffit ensuite d'exécuter la macro pour répéter l'ensemble des commandes enregistrées.

Le terme macro désigne le regroupement d'un ensemble de commandes en une seule. On parle parfois de macrocommandes pour désigner un programme qui se résume à l'exécution d'une série de commandes, sans égard pour le contexte. Des macros plus évoluées peuvent répéter des opérations en boucle, afficher des boîtes de dialogue qui autorisent une interaction avec l'utilisateur. Ces programmes se comporteront différemment en fonction des informations entrées ou de l'état du document sur lequel elles s'exécutent.

Le terme projet est plus large. Il désigne l'ensemble des éléments constituant vos programmes VBA. Il s'agit toujours de macros, mais à celles-ci peuvent s'ajouter des feuilles – qui constituent une interface utilisateur permettant de récolter des informations de tout type –, des modules de classe, et autres friandises que vous découvrirez tout au long de cet ouvrage.

L'enregistrement de macros constitue sans aucun doute le meilleur moyen de se familiariser avec la programmation en Visual Basic. Ainsi, sans connaître le langage – les instructions qui

le composent et la façon dont elles sont structurées –, vous pouvez créer des programmes VBA et en visualiser ensuite le code.

Une application hôte et des projets

Visual Basic pour Applications est un environnement de développement calqué sur Visual Basic, un outil de développement d'applications Windows. Les structures de contrôle du langage sont les mêmes et l'environnement proprement dit (Visual Basic Editor) est pour ainsi dire identique à celui de Visual Basic. Mais, contrairement à Visual Basic, Visual Basic pour Applications est conçu… *pour des applications*. Cela signifie que, tandis que les programmes Visual Basic sont autonomes, les programmes VBA ne peuvent être exécutés qu'à partir d'une application intégrant cet environnement de développement – Excel ou une autre application.

Lorsque vous développez un programme VBA, vous l'attachez à une application. Il s'agit de l'*application hôte* du programme. Plus précisément, vos programmes VBA sont attachés à un document (un fichier ou un modèle Word, une feuille de calcul Excel, une présentation PowerPoint…) spécifique à l'application hôte. L'ensemble des programmes VBA attachés à un document constitue un projet. Un projet regroupe des macros, mais peut également intégrer des interfaces utilisateur, des déclarations système, etc. Un projet constitue en fait la partie VBA d'un document. Si cet ouvrage ne traite que de la programmation pour Excel, sachez qu'un programme VBA peut être attaché à une autre application. Les concepts et les outils que vous découvrirez au long de cet ouvrage sont valides pour toutes les applications de la suite Office. Pour exécuter une macro VBA, vous devez avoir accès au document auquel elle est attachée. Vous pouvez choisir de rendre certaines macros disponibles à partir de n'importe quel document Excel ou en limiter l'accessibilité à un classeur Excel spécifique. La disponibilité des programmes VBA est abordée au Chapitre 2.

Un langage de programmation

Les projets VBA sont développés dans le langage de programmation Visual Basic. Vous découvrirez par la pratique la structure de ce langage et apprendrez rapidement à en discerner les composants et les relations qu'ils entretiennent. Comme nous l'avons dit précédemment, l'enregistrement de macros constitue une excellente initiation à Visual Basic. C'est sous cet angle que nous vous ferons découvrir ce langage.

Visual Basic est un langage de programmation *orienté objet*. Nous présenterons donc les concepts de la programmation orientée objet (POO). Vous apprendrez ce qu'est un objet, une propriété, une méthode ou un module de classe, etc. Vous verrez comment conjuguer ces éléments pour créer des applications Excel souples et puissantes. Visual Basic pour Applications constitue une bonne approche de la programmation pour le néophyte.

Visual Basic pour Applications intègre un grand nombre d'instructions. Cela permet de développer des macros susceptibles d'identifier très précisément l'état de l'application et des documents et reproduire l'exécution de la plupart des commandes disponibles dans l'application hôte.

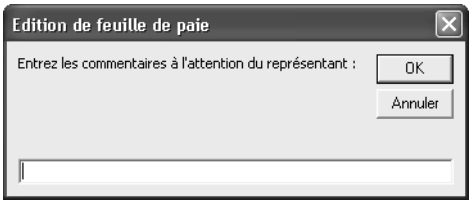
Vous verrez que certaines instructions sont spécifiques à Excel. C'est, par exemple, le cas des instructions permettant d'affecter une formule à une cellule. Vous n'utiliserez probablement qu'un nombre limité de ces instructions, en fonction de votre usage personnel d'Excel ou des

besoins de votre entreprise. Par ailleurs, certaines instructions spécifiques à Excel apparaîtront presque toujours dans vos macros. C'est, par exemple, le cas de la propriété `Range` qui renvoie un objet Excel tel qu'une cellule ou une plage de cellules.

D'autres instructions sont communes à l'ensemble des applications Office. C'est le cas de celles qui permettent de régler le comportement d'une macro : réaliser des opérations en boucle, induire des réactions face à certains paramètres, afficher des boîtes de dialogue simples (voir Figures 0.3 et 0.4) ou développer des interfaces utilisateur évoluées (voir Figure 0.1), etc. Ce sont ces instructions qui constituent véritablement ce qu'il est convenu d'appeler *le langage Visual Basic*. Vous aurez besoin d'y faire appel dès que vous voudrez créer un programme interactif, capable de se comporter différemment selon le contexte. La plupart de ces instructions ne peuvent être générées par enregistrement de macros, et doivent donc être éditées manuellement dans Visual Basic Editor.

Figure 0.3
La fonction VBA MsgBox permet d'afficher une boîte de dialogue.

Figure 0.4
Il existe une version VBA et une version Excel de la fonction InputBox.



Cet ouvrage ne se veut pas un dictionnaire du langage, mais un guide qui vous enseignera le développement de projets VBA de qualité. Vous apprendrez à enregistrer, modifier, exécuter et déboguer des macros, à créer des interfaces utilisateur ainsi qu'à gérer vos projets VBA. Vous découvrirez, à travers les nombreux exemples de projets VBA de cet ouvrage, un certain nombre d'instructions spécifiques à la *hiérarchie d'objets* d'Excel, qui vous familiariseront avec la logique de ce langage.

La hiérarchie d'objets d'une application, encore appelée modèle d'objets, est le rapport qu'entretiennent entre eux les différents objets d'une application. Ce concept ainsi que les notions spécifiques aux langages orientés objet seront développés au Chapitre 1, "Notions fondamentales de la programmation orientée objet".

En revanche, ce livre présente et illustre d'exemples commentés l'ensemble des structures de contrôle qui permettront de créer très simplement des macros évoluées. Nous vous fournirons les bases du langage Visual Basic. Elles suffisent pour créer une infinité de macros et répondre à vos besoins spécifiques.

Lorsque les principes du développement de projets VBA vous seront acquis et que vous créerez vos propres macros, il vous arrivera sûrement d'avoir besoin d'instructions que vous n'aurez pas rencontrées lors de la lecture de cet ouvrage ; vous pourrez alors utiliser l'Enregistreur de macro ou encore les rechercher dans l'aide de Visual Basic pour Applications ou dans l'Explorateur

d'objets – étudié au Chapitre 4. Vous verrez que l'aide de Visual Basic pour Applications fournit une référence complète du langage, facilement accessible et consultable.

Si vous n'avez aucune expérience de programmation, peut-être ce *Visual Basic* vous apparaît-il comme un langage barbare ou inaccessible. Ne vous inquiétez pas : le développement de projets VBA ne requiert ni expérience préalable de la programmation, ni connaissance globale du langage. Contentez-vous, au cours de votre lecture, d'utiliser les fonctions nécessaires aux exercices et que nous vous détaillerons. Cet ouvrage propose un apprentissage **progressif** et **concret** : vous développerez vos premiers projets VBA dès les premiers chapitres.

Un environnement de travail

Visual Basic pour Applications dispose d'un environnement de développement à part entière : Visual Basic Editor.

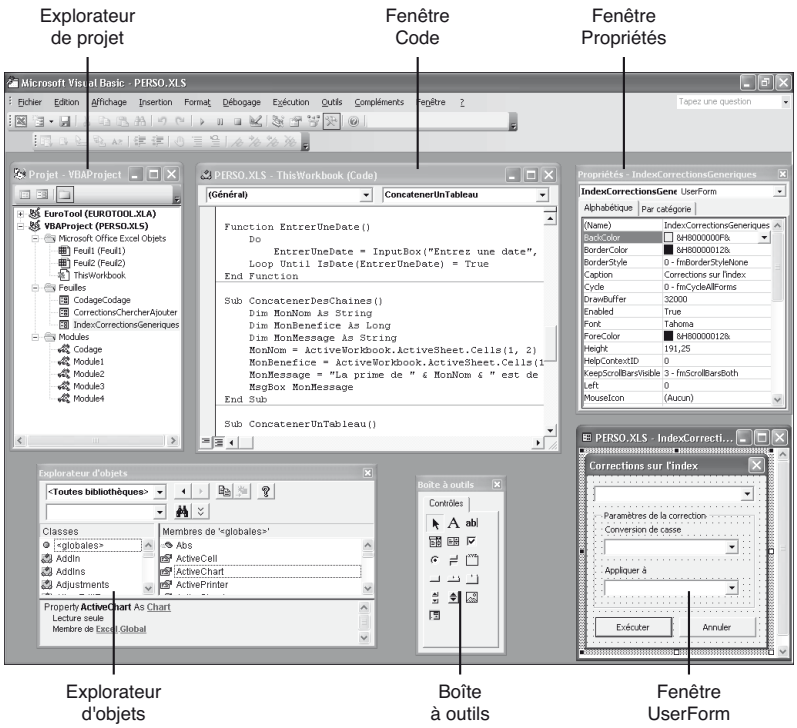


Figure 0.5
Visual Basic Editor est l'environnement de développement de Visual Basic pour Applications.

Visual Basic Editor est l'*environnement de développement intégré* des applications Office. Il permet de visualiser et de gérer les projets VBA, d'écrire, de modifier et de déboguer les macros existantes, de visualiser comment les commandes propres à une application Office sont traduites en langage Visual Basic, et inversement. C'est aussi un outil de débogage de vos projets VBA d'une grande efficacité. Visual Basic Editor propose nombre d'outils permettant de tester les macros et d'en étudier le comportement. Vous pouvez ainsi exécuter les commandes

de la macro pas à pas, en suivre le déroulement, insérer des commentaires dans le texte de la macro, etc. Enfin, Visual Basic Editor intègre des outils très intuitifs, dédiés au développement d'interfaces graphiques.

Vous apprendrez dans cet ouvrage à utiliser les nombreux outils de Visual Basic Editor à toutes les phases de développement d'un projet VBA.

Conventions typographiques

Afin d'en faciliter la lecture, nous avons adopté dans cet ouvrage un certain nombre de conventions typographiques. Lorsqu'un mot apparaît pour la première fois, il est composé en *italique*. Les programmes et les mots clés du langage Visual Basic apparaissent dans `une police à chasse fixe`. Lorsque, dans un programme, un mot signale une information attendue dans le code, celui-ci apparaît en *`italique`*.

Lorsqu'une ligne de code ne peut être inscrite sur une seule ligne de l'ouvrage, cette flèche (➥) en début de ligne indique que le texte est la poursuite de ligne précédente.

Par ailleurs, vous rencontrerez au long de cet ouvrage différents types de notes, signalées dans la marge par des pictogrammes.

Ces rubriques apportent un complément d'information en rapport avec le sujet traité. Leur lecture n'est pas indispensable. Mais elles peuvent vous aider à mieux cerner le sujet.

Vous trouverez sous ces rubriques la définition de termes techniques spécifiques à la programmation VBA.

Ces rubriques vous mettent en garde contre les risques inhérents à telle ou telle commande ou manipulation.

Il est parfois nécessaire de se rafraîchir la mémoire. Lorsqu'un sujet fait appel à des connaissances acquises plusieurs chapitres auparavant, cette rubrique vous les remémore brièvement.

Sous cette rubrique, vous trouverez des trucs pour aller plus vite et travailler plus efficacement.

Nous vous faisons ici part de notre expérience, en vous prodiguant des conseils qui vous aideront à développer des projets VBA de qualité.

Ces notes prodiguent des informations spécifiques aux versions antérieures à Office 2010.