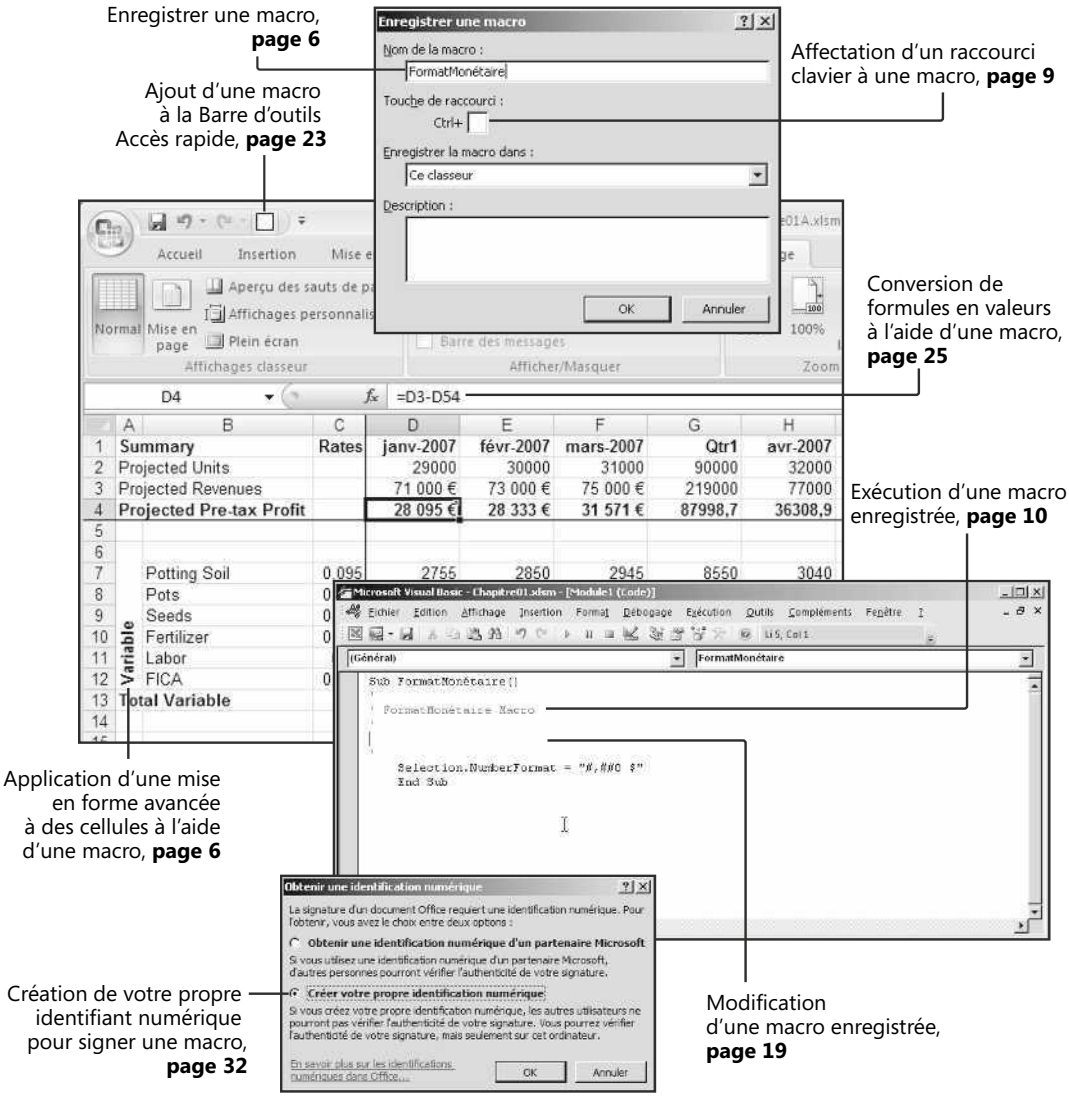
Coup d'œil sur le chapitre



1 Création d'une macro pour effectuer des tâches simples

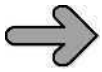
Ce chapitre aborde les points suivants :

- ✔ Enregistrer et exécuter une macro.
- ✔ Comprendre et modifier des macros simples enregistrées.
- ✔ Exécuter une macro à l'aide d'un raccourci clavier.
- ✔ Gestion de la sécurité des macros.

Il y a quelques semaines, j'ai égaré la télécommande de mon magnétoscope numérique. Un cauchemar. J'essayais de voir un épisode en temps différé parfaitement légalement de *Desperate Housewives*, sans y parvenir parce que les contrôles de navigation sont intégrés à la télécommande. Heureusement, en allant chercher un nouveau paquet de pop-corn dans un placard quelques jours plus tard, j'ai découvert ce que j'avais fait de la télécommande. Je puis avec bonheur regarder à nouveau des émissions en temps différé, sans parler du plaisir de pouvoir changer de chaîne sans devoir me lever.

VBA (*Visual Basic for Applications*) est la télécommande de Microsoft Office Excel 2007. Vous pouvez sans problème employer Excel sans jamais vous servir de VBA, mais la télécommande VBA facilite l'emploi d'Excel. Elle permet également de tirer profit de dispositifs auxquels ne donne pas accès l'interface utilisateur standard. Une fois familiarisé avec VBA, vous vous demanderez comment vous aviez pu vous en passer.

Important Avant d'entamer ce chapitre, vous devez télécharger les fichiers d'exercices depuis le site Dunod associé et les installer à leur emplacement par défaut. Reportez-vous pour plus d'information à « Emploi des fichiers d'exercices », dans l'introduction.



SERVEZ-VOUS du classeur *Budget.xlsx*. Ce fichier d'exercice se trouve dans le dossier *Documents\MSP\ExcelVBA07SBS*. Ce classeur contient une unique feuille de calcul nommée *Budget*. Cette feuille de calcul contient un budget mensuel prévisionnel pour l'année 2007.
VEILLEZ à enregistrer le classeur *Budget.xlsx* sous le nom *Chapitre01* dans un nouveau dossier nommé Travail. Vous placez le classeur dans un nouveau dossier de façon à être certain que celui-ci ne contient rien d'autre que les fichiers que vous y placez. Par la suite, cela vous permettra d'accorder votre confiance à tout classeur de macro de ce dossier.
OUVREZ le classeur *Chapitre01.xlsx*.

Différence entre VBA et une macro

Si VBA est la télécommande d'Excel, qu'est-ce donc qu'une **macro** ? Et quelle est la différence entre VBA et une macro ? Tout cela est très confus. Fondamentalement, une macro est un programme informatique qui donne des instructions automatisées à l'ordinateur. Les macros originales étaient une façon d'employer quelques caractères pour représenter un ensemble d'instructions. Elles étaient appelées *macros* car la sortie était bien plus imposante que l'entrée.

En fait, les premières macros de programmes de tableurs se bornaient à développer une courte chaîne de caractères en un long ensemble d'actions. Ce n'était que des raccourcis pour des commandes de l'interface utilisateur. Par exemple, si vous saisissez dans l'interface utilisateur *R* (pour « *Range* », plage), *N* (pour « *Name* », nom) et *C* (pour « *Create* », créer), vous saisissez dans la macro *RNC* pour automatiser le processus. Cette approche était intuitive, mais non dépourvue de faiblesses inhérentes. Les macros de séquence de touches étaient difficiles à lire et ne s'adaptaient guère à une interface utilisateur graphique. Quelle séquence de touches employer pour représenter le déplacement d'un rectangle à l'aide de la souris ? Elles rendaient également délicate l'amélioration de l'interface utilisateur, car toute modification apportée à la structure des menus entraînait l'échec des macros créées précédemment.

Pour résoudre ces problèmes, les premières versions d'Excel contenaient un nouveau type de langage de programmation, indépendant des noms des commandes de l'interface utilisateur. Par exemple, dans Microsoft Excel version 4 (Excel 2007 est la version 12), vous pouviez copier une plage d'au moins trois façons différentes : en appuyant sur CTRL+C, en cliquant sur le bouton Copier de la barre d'outils standard ou en cliquant sur Copier dans le menu Edition. Ces trois méthodes étaient représentées par la même instruction =COPY(). Ce nouveau langage de programmation n'était pas techniquement un langage de macro dans

la mesure où il ne développe pas quelques caractères en une séquence d'instructions. Techniquement, c'était un ensemble de fonctions, très similaires aux fonctions employées pour effectuer des tâches dans les cellules d'une feuille de calcul. Proclamer toutefois « vous pouvez automatiser des tâches simples en écrivant des programmes personnalisés fondés sur les fonctions spécialisées de feuille de calcul » pouvait paraître effrayant. L'équipe Excel a donc continué à faire référence aux programmes personnalisés sous le terme de macros. Le mot *macro* signifie désormais *un programme que les utilisateurs peuvent eux-mêmes écrire*.

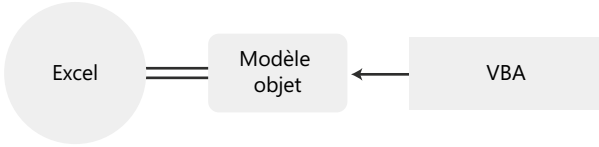
Les premières macros fondées sur des fonctions d'Excel représentaient une amélioration majeure sur les macros de séquence de touches, mais possédaient encore deux gros inconvénients. Tout d'abord, les macros Excel étaient très spécifiques à Excel : les fonctions ressemblaient trop à des formules de feuille de calcul pour pouvoir être adaptées à d'autres applications, comme Microsoft Office Word. Ensuite, le nombre de fonctions progressait avec chaque nouvelle version de Microsoft Office, sans qu'il n'existe de méthode efficace pour regrouper ou organiser les milliers de possibilités.

Pour éliminer la première contrainte (la spécificité des macros fondées sur des fonctions propres à Excel), Microsoft a présenté VBA (*Visual Basic for Applications*) dans la version 5 de Microsoft Excel.

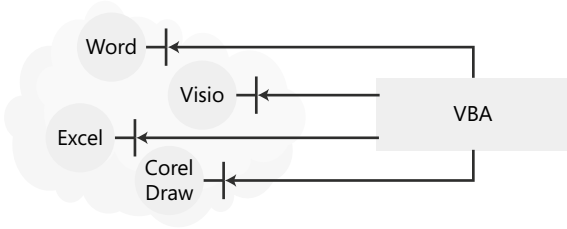
VBA agit comme un langage à but général indépendant de l'application. Soudainement, toute personne sachant travailler avec n'importe quelle version de Microsoft Visual Basic disposait d'une avance confortable en matière d'automatisation d'Excel et quiconque apprenait à écrire des macros Excel en VBA pouvait transférer ses connaissances vers les autres types de programmation Visual Basic. En outre, même si Excel était la première grande application à employer VBA, ce dernier n'était pas lié à Excel. Il fonctionnait tout aussi bien avec d'autres applications compatibles VBA, comme Word et Microsoft Office PowerPoint.

Pour résoudre la seconde contrainte des macros fondées sur les fonctions (l'existence de trop de commandes pour pouvoir les gérer efficacement), VBA fonctionne avec un *modèle objet*. Le terme *modèle objet* peut sembler intimidant, mais ce n'est qu'une façon logique d'organiser toutes les commandes dont dispose une application. Avec un modèle objet, chaque partie distincte d'une application (par exemple, un classeur, une plage ou un point sur un graphique) devient un *objet*, chaque objet disposant de sa propre liste de fonctions. Vous en apprendrez plus par la suite sur ce qu'est un objet et la façon dont les objets sont liés aux fonctions. Le point essentiel est que le modèle objet organise les millions de commandes possibles selon la façon dont est employée chaque commande. Par exemple, vous copiez et collez une plage de cellules, mais pas les points d'un graphique.

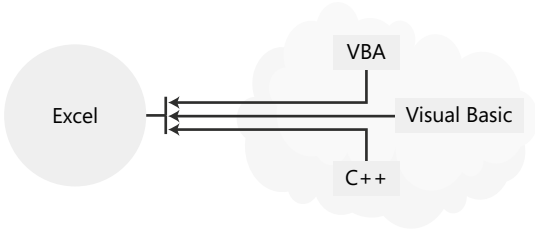
Grâce au modèle objet, VBA n'a besoin d'aucun accès particulier aux mécanismes internes d'Excel. Excel expose ses capacités au monde extérieur à l'aide du modèle objet et c'est à celui-ci que s'adresse VBA.



Cela signifie qu'une macro VBA Excel peut contrôler non seulement Excel, mais également toute application qui fournit un modèle objet. Toutes les applications Microsoft Office ainsi que de nombreuses autres applications Microsoft ou de tierce partie proposent les modèles objet adéquats.



Le VBA fourni avec Excel n'est pas le seul langage capable de communiquer avec le modèle objet. Tout langage qui prend en charge l'Automation peut contrôler Excel. Vous pouvez donc contrôler Excel bien sûr avec le VBA hébergé par Excel, mais également avec le VBA hébergé par Word, avec Microsoft Visual Basic version 6 ou même avec un langage comme C++. Avec une unique couche de traduction, vous pouvez également parler au modèle objet Excel depuis des applications Microsoft .NET écrites en C# ou en Microsoft Visual Basic .NET.



VBA et le .NET Framework

Si .NET ne signifie rien pour vous, sautez cette explication sans plus vous en préoccuper. Si vous connaissez bien .NET, vous pourriez vous demander en quoi les macros VBA sont apparentées aux programmes .NET.