Programando Excel VBA

Para leigos

As teclas de atalho do Excel permitem que você realize certas tarefas usando apenas o teclado. A ideia é que você aumente sua eficiência ao limitar o número de vezes que suas mãos precisam ir e voltar do teclado para o mouse. Desenvolver o hábito de usar essas teclas de atalho pode ajudá-lo a trabalhar com mais eficácia ao usar o Visual Basic Editor.

TECLAS DE ATALHO PADRÕES DO VISUAL BASIC EDITOR

Ao trabalhar com o Visual Basic Editor, transite pelas janelas usando atalhos do teclado em vez de pegar no mouse. Esses atalhos possibilitarão a navegação pela interface do Visual Basic Editor.

O que Pressionar	O que Acontece
Alt+F11	Alterna entre as janelas do VBE e do Excel.
Shift+F10	Exibe o menu de atalho da janela ativa (replica o clique com o botão direito do mouse).
Ctrl+R	Abre a janela Projeto (ou Project Explorer).
F4	Abre a janela Propriedades.
F2	Abre o Pesquisador de Objetos (ou Object Browser).
F1	Abre a Ajuda do VBA.
F7	Ativa a janela abrir módulo.

TECLAS DE ATALHO PARA TRABALHAR NA JANELA DE CÓDIGO DO VISUAL BASIC EDITOR

Em certo ponto, você pode se ver trabalhando com muitas macros ao mesmo tempo. Pode ser chato tentar navegar entre e dentro de procedimentos de macro clicando com o mouse. Estes atalhos de teclado possibilitam pular para um procedimento-alvo, navegar pelos módulos e até encontrar o ponto inicial de variáveis.

O que Pressionar	O que Acontece
Ctrl+seta para baixo	Seleciona o próximo procedimento.
Ctrl+seta para cima	Seleciona o procedimento anterior.
Ctrl+Page Down	Desde uma tela.
Ctrl+Page Up	Sobe uma tela.
Shift+F2	Vai para a função ou variável selecionada.
Ctrl+Shift+F2	Vai para a última posição.
Ctrl+Home	Vai para o começo do módulo.
Ctrl+End	Vai para o final do módulo.

Ctrl+seta para a direita	Move-se uma palavra para a direita.
Ctrl+seta para a esquerda	Move-se uma palavra para a esquerda.
End	Move-se para o fim da linha.
Home	Move-se para o começo da linha.
Tab	Recua a linha atual.
Shift+Tab	Remove a indentação da linha atual.
Ctrl+J	Lista as propriedades e os métodos do objeto selecionado.

TECLAS DE ATALHO PARA DEPURAR O CÓDIGO NO VISUAL BASIC EDITOR

Depurar seu código é uma parte importante de trabalhar com macros do Excel. Embora haja maneiras de usar os recursos de depuração por meio das opções do menu do Visual Basic Editor, você pode achar estes atalhos de teclado muito mais eficazes para depurar seu código.

O que Pressionar	O que Acontece
F5	Executa o procedimento atual ou continua depois de pausar.
Ctrl+Break	Interrompe o procedimento executado no momento.
F8	Vai para o modo de depuração e executa uma linha de cada vez.
Ctrl+F8	Executa o código até encontrar o cursor.
Shift+F8	Examina a linha atual enquanto está no modo de depuração.
F9	Ativa/desativa um ponto de interrupção para a linha selecionada atual.
Ctrl+Shift+F9	Limpa todos os pontos de interrupção.
Alt+D+L	Compila o projeto Visual Basic atual.

TECLAS DE ATALHO PARA NAVEGAR NA JANELA PROJETO DO VISUAL BASIC EDITOR

Quer navegar pelos seus projetos Visual Basic sem colocar a mão no mouse? Tente usar estes atalhos de teclado para se mover entre projetos e módulos:

O que Pressionar	O que Acontece
Seta para cima	Sobe a lista do projeto um item de cada vez.
Seta para baixo	Desce a lista do projeto um item de cada vez.
Home	Move-se para o primeiro arquivo na lista do projeto.
End	Move-se para o último arquivo na lista do projeto.
Seta para a direita	Maximiza a pasta selecionada.
Seta para a esquerda	Minimiza a pasta selecionada.
F7	Abre a janela de código do arquivo selecionado.

Programando Excel® VBA

Para leigos

Programando Excel® VBA

Para leigos

Tradução da 5ª Edição

Michael Alexander
e John Walkenbach

ALTA BOOKS
EDITORA
Rio de Janeiro, 2019

Sobre o Autor

Michael Alexander é Microsoft Certified Application Developer (MCAD) e autor de vários livros sobre análise avançada de negócios com Microsoft Access e Microsoft Excel. Tem mais de 20 anos de experiência em consultoria e desenvolvimento de soluções Microsoft Office. Mike foi nomeado Microsoft MVP por suas contribuições contínuas à comunidade Excel. Encontre ele em `www.datapigtechnologies.com` [conteúdo em inglês].

Sumário Resumido

Introdução

Saudações, futuro programador de Excel...

Sem dúvida, você tem suas razões para escolher um livro sobre programação VBA. Talvez tenha conseguido um novo emprego (parabéns). Talvez esteja tentando automatizar algumas das tarefas repetitivas de processamento de dados que precisa realizar. Talvez, no fundo, você seja só um nerd. Independentemente da razão, muito obrigado por escolher este livro.

Aqui você encontrará tudo o que precisa para começar a trabalhar com o VBA rapidamente. Mesmo que não tenha a mínima ideia do que seja programar, este livro pode ajudar. Diferentemente da maioria dos livros de programação, este tem várias informações que foram projetadas para incluir apenas o que você precisa saber para aumentar seu conjunto de habilidades em programação VBA.

Sobre Este Livro

Vá a qualquer grande livraria (física ou online) e encontrará muitos livros sobre Excel. Uma rápida olhada pode ajudá-lo a decidir se este é realmente o certo para você. Este livro

- Foi desenvolvido para usuários intermediários a avançados de Excel que pretendem se adaptar rapidamente à programação de Visual Basic para Aplicações (VBA).
- Não requer experiência anterior com programação.
- Cobre os comandos mais comumente usados.
- É adequado para Excel 2013, Excel 2016 ou Excel 2019.
- Poderá fazê-lo rir de vez em quando.

Se estiver usando o Excel 2003, precisará de um livro muito mais antigo (e de um abraço). Se estiver usando o Excel 2007 ou 2010 *pode* não ter problemas, mas algumas coisas mudaram. Você talvez se dê melhor com a edição anterior.

A propósito, esta não é uma obra de introdução ao Excel. Se você estiver procurando um livro de Excel geral, verifique quaisquer outros publicados pela Alta Books, no site `www.altabooks.com.br`.

Estes livros também estão disponíveis em edições para versões anteriores do Excel.

Note que o título deste livro não é *O Guia Completo de Programação VBA em Excel Para Leigos*. Eu não abordo todos os aspectos de programação em Excel — e, novamente, é provável que você não queira saber *tudo* sobre esse assunto.

Se ler esta obra e se descobrir faminto por um material de programação de Excel mais abrangente, experimente o *Microsoft Excel 2019 Power Programming with VBA* [sem tradução no Brasil]. E sim, edições para versões anteriores do Excel também estão disponíveis.

Seção Obrigatória das Convenções Tipográficas

Todos os livros de computação possuem uma seção como esta (eu acho que existe alguma lei federal exigindo isso). Leia ou simplesmente pule esta parte.

Algumas vezes, eu me refiro a combinações de teclas, o que significa que você mantém uma tecla pressionada enquanto pressiona outra. Por exemplo, Ctrl+Z significa que mantém a tecla Ctrl pressionada enquanto pressiona Z.

Nos comandos de menu, uso um caractere distinto para separar itens do menu ou da Faixa de Opções. Por exemplo, você usa o seguinte comando para criar um intervalo nomeado em uma planilha:

Fórmulas ➪ Nomes Definidos ➪ Definir Nome

Fórmulas é a guia no topo da Faixa de Opções, Nomes Definidos é o grupo da Faixa de Opções e Definir Nome é o comando em si.

O Visual Basic Editor ainda usa menus e barras de ferramentas à moda antiga. Então Ferramentas ➪ Opções significa escolher o menu Ferramentas e selecionar o item Opções do menu.

A programação em Excel envolve desenvolver *código*, isto é, as instruções que o Excel segue. Todo código neste livro é apresentado em fonte monoespaçada, assim:

```
Range("A1:A12").Select
```

Algumas linhas longas de código não cabem nas margens deste livro. Nesses casos, uso a sequência de caracteres padrão VBA de continuação de linha: um espaço seguido por um caractere de sublinhado. Eis um exemplo:

```
Selection.PasteSpecial Paste:=xlValues, _
Operation:=xlNone, SkipBlanks:=False, _
Transpose:=False
```

Quando inserir esse código, você poderá digitá-lo como está ou colocá-lo em uma linha única (retirando os espaços e os sublinhados).

Verifique Suas Configurações de Segurança

Vivemos em um mundo cruel. Parece que há sempre um canalha tentando se aproveitar de você ou causando algum tipo de problema. O mundo da computação não é diferente. Você provavelmente conhece os vírus de computador, que podem causar coisas desagradáveis em seu sistema. Mas, sabia que eles também podem estar em um arquivo Excel? É verdade. É relativamente fácil escrever um vírus de computador usando VBA. Um usuário desavisado pode abrir um arquivo Excel e espalhar o vírus para outros arquivos Excel e outros sistemas.

Com o passar dos anos, a Microsoft ficou cada vez mais preocupada com problemas de segurança. Isso é bom, mas também significa que os usuários do Excel precisam entender como as coisas funcionam. Você pode verificar as configurações de segurança do Excel clicando no comando Arquivo ➪ Opções ➪ Central de Confiabilidade ➪ Configurações da Central de Confiabilidade. Existem muitas opções lá dentro, e corre o boato que nunca mais se ouviu falar das pessoas que abriram tal caixa de diálogo.

Se clicar na guia Configurações de Macro (à esquerda da caixa de diálogo Central de Confiabilidade), as suas opções serão as seguintes:

- **Desabilitar todas as macros sem notificação.** As macros não funcionarão, independentemente do que você faça.
- **Desabilitar todas as macros com notificação.** Quando abrir um arquivo com macros, você verá uma Barra de Mensagem aberta com uma opção para clicar e habilitar as macros, ou (se a janela do Visual Basic Editor estiver aberta) receberá uma mensagem perguntando se quer habilitar as macros.
- **Desabilitar todas as macros, exceto as digitalmente assinadas.** Apenas macros com uma assinatura digital podem rodar (porém, até mesmo para as assinaturas que não foram marcadas como confiáveis, você receberá o aviso de segurança).

- **Habilitar todas as macros.** Todas as macros rodam sem avisos. Essa opção não é recomendada pois códigos possivelmente perigosos podem ser executados.

Imagine este cenário: você passa uma semana escrevendo um programa VBA incrível que revolucionará a sua empresa. Você o testa cuidadosamente e depois o envia ao seu chefe. Ele o chama ao seu escritório e reclama que a sua macro não faz absolutamente nada. O que está acontecendo? Possivelmente, as configurações de segurança do Excel de seu chefe não permitem a execução de macros. Ou talvez ele tenha decidido aceitar a sugestão padrão da Microsoft e desativar as macros ao abrir o arquivo.

A questão toda? Só porque uma pasta de trabalho do Excel contém uma macro, isso não garante que a macro será executada. Tudo depende da configuração de segurança e se o usuário decide ativar ou desativar macros para aquele arquivo.

Para trabalhar com este livro, será preciso habilitar as macros para os arquivos com os quais você trabalha. Meu conselho é usar o segundo nível de segurança. Então, quando abrir o arquivo que criou, poderá simplesmente habilitar as macros. Se abrir um arquivo de alguém que não conhece, deve desabilitar as macros e verificar o código VBA para ter certeza de que não possui nada destrutivo ou malicioso. Geralmente é muito fácil identificar um código VBA suspeito.

Outra opção é designar uma pasta confiável. Escolha Arquivo ➪ Opções ➪ Central de Confiabilidade ➪ Configurações da Central de Confiabilidade. Selecione a opção Locais Confiáveis e designe uma pasta específica para ser um local confiável. Armazene suas pastas de trabalho confiáveis lá e o Excel não o incomodará com a habilitação de macros. Por exemplo, se fizer o download dos arquivos de amostra deste livro, poderá colocá-los em um local confiável.

Penso que...

Normalmente, pessoas que escrevem livros têm em mente um leitor-alvo. Os pontos a seguir descrevem mais ou menos o meu leitor-alvo hipotético:

- Você tem acesso a um PC no trabalho, e provavelmente em casa. E esses computadores estão conectados à internet.
- Usa Excel 2013, Excel 2016 ou Excel 2019.
- Vem usando computadores há muitos anos.
- Usa o Excel com frequência no trabalho e considera ser mais capaz no uso da ferramenta do que o público geral.

- Precisa que o Excel faça algumas coisas que atualmente não consegue que ele faça.
- Tem pouca ou nenhuma experiência em programação.
- Compreende que o sistema de Ajuda do Excel pode realmente ser útil. Encare os fatos, este livro não cobre tudo. Se puder se entender com o sistema de Ajuda, você conseguirá achar as peças que faltam.
- Precisa concluir algum trabalho e não tem muita tolerância com livros grossos e chatos sobre computação.

Ícones Usados Neste Livro

DICA

Não pule as informações marcadas com este ícone. Ele identifica um atalho que pode poupar muito do seu tempo (e talvez até permita que saia do trabalho em um horário razoável).

Este ícone também é usado para que você saiba que o código discutido está disponível na web. Faça o download dele para evitar muita digitação.

LEMBRE-SE

Este ícone diz quando você precisa armazenar informações nas profundezas do seu cérebro para uso posterior.

PAPO DE ESPECIALISTA

Este ícone sinaliza o material que pode ser considerado técnico. Talvez você o ache interessante, mas, se estiver com pressa, pode pulá-lo.

CUIDADO

Leia tudo o que estiver marcado com este ícone. Caso contrário, poderá perder seus dados, explodir seu computador, causar uma fusão nuclear — ou talvez até arruinar todo o seu dia.

Arquivos de Amostra Online

Este livro tem arquivos de amostra online que podem ser baixados. Para fazer o download dos arquivos, visite

```
www.altabooks.com.br
```

e procure pelo título do livro.

Com os arquivos de exemplos, você poupará muita digitação. Melhor ainda, será possível brincar com eles e experimentar diversas alterações. Na verdade, a melhor maneira de dominar VBA é experimentando.

De Lá para Cá, Daqui para Lá

Este livro contém tudo o que você precisa para aprender programação VBA em um nível médio-avançado. Ele começa com o básico da gravação de macros e vai desenvolvendo capítulo por capítulo.

Se você desconhece completamente as macros de Excel, comece pela Parte 1 para se atualizar sobre os fundamentos da gravação de macros. Se já tem experiência com isso, mas quer entender melhor o VBA por trás delas, vá para a Parte 2, na qual obterá um entendimento conciso de como funciona o VBA, além da base de que precisa para implementar seu próprio código.

Por fim, se já estiver familiarizado com os conceitos de programação e só quiser dar uma passada rápida por algumas técnicas mais avançadas, como criar funções e add-ins personalizados, sinta-se à vontade para pular diretamente para a Parte 4. Há também uma Folha de Cola cheia de atalhos úteis. Visite o site `www.altabooks.com.br` e procure pelo título do livro para encontrá-la.

1

Começando com a Programação Excel VBA

NESTA PARTE . . .

Conheça o Visual Basic para Aplicações.

Veja exemplos de algumas das coisas que você pode fazer com VBA.

Trabalhe com uma seção real de programação Excel.

Descubra como o Excel lida com a segurança macro.

NESTE CAPÍTULO

- Obtendo uma visão geral conceitual do VBA
- Descobrindo o que se pode fazer com o VBA
- Descobrindo vantagens e desvantagens de usar o VBA
- Entendendo tudo sobre o que é o VBA
- Mantendo-se compatível ao Excel

Capítulo **1**

O que É VBA?

Se você está ansioso para pular na programação VBA, espere um pouco. Este capítulo é totalmente desprovido de qualquer material de treinamento prático. No entanto, ele contém algumas informações essenciais de apoio que o ajudam a se tornar um programador de Excel. Em outras palavras, este capítulo prepara o caminho para tudo que vem pela frente e dá a você uma ideia de como a programação de Excel se ajusta no esquema geral do universo. Não é tão chato quanto imagina, então tente resistir ao desejo de pular para o Capítulo 2.

Tudo Bem, Então o que É VBA?

VBA, que significa *Visual Basic for Applications* (Visual Basic para Aplicações), é uma linguagem de programação desenvolvida pela Microsoft — você sabe, a empresa que tenta fazê-lo comprar uma nova versão do Windows de tempos em tempos. Excel, juntamente a outros membros do Microsoft Office, inclui a linguagem VBA (sem custos extras). Resumindo, VBA é uma ferramenta que as pessoas usam para desenvolver programas que controlam o Excel.

Imagine um robô inteligente que sabe tudo sobre o Excel. Esse robô pode ler instruções e também operar o Excel com muita rapidez e precisão. Quando quer

ALGUMAS PALAVRAS SOBRE A TERMINOLOGIA

PAPO DE ESPECIALISTA

A terminologia de programação em Excel pode ser um pouco confusa. Por exemplo, VBA é uma linguagem de programação, mas também serve como uma linguagem de macro. Nesse contexto, macro é um conjunto de instruções que o Excel realiza para imitar as teclas pressionadas e as ações do mouse. Como você chama algo escrito em VBA e executado em Excel? É uma *macro* ou um *programa*? Normalmente, o sistema de Ajuda do Excel se refere aos procedimentos VBA como *macros*, portanto essa é a terminologia usada neste livro. Mas também pode-se chamar isso de *programa*.

Você verá o termo *automatizar* neste livro. Significa que uma série de etapas são completadas automaticamente. Por exemplo, se você escrever uma macro que acrescenta cor a algumas células, imprime a planilha e depois remove a cor, essas três etapas foram *automatizadas*.

A propósito, *macro* não é um acrônimo de **M**essy **A**nd **C**onfusing **R**epeated **O**peration (Operação Repetida Confusa e Desordenada). Ela vem da palavra grega *makros*, que significa grande — e também descreve o seu contracheque depois de se tornar um programador especialista em macro.

que o robô faça algo no Excel, você escreve um conjunto de instruções para robôs usando códigos especiais. Depois diz ao robô para seguir suas instruções enquanto se senta e toma uma limonada. Isso é VBA — uma linguagem de código para os robôs. Note, entretanto, que o Excel não vem com um robô e nem com limonada.

O que Você Pode Fazer com o VBA?

Você provavelmente está ciente de que os usuários do Excel usam o programa para milhares de diferentes tarefas. Seguem alguns exemplos:

- Análise de dados científicos
- Preparação de orçamentos e previsões financeiras
- Criação de faturas e outros formulários
- Desenvolvimento de gráficos de dados
- Manutenção de listagens de assuntos como nomes de clientes, notas de alunos ou ideias para presentes (um lindo bolo de frutas seria ótimo)
- Etc. etc. etc.

Os exemplos são muitos, mas acredito que você já entendeu. O que quero dizer é que o Excel é usado para uma grande variedade de tarefas, e qualquer um que ler este livro tem diferentes necessidades e expectativas em relação a ele. Uma coisa que todos os leitores têm em comum é *a necessidade de automatizar algum aspecto do Excel*. Isso, meu caro leitor, é o que VBA faz.

Por exemplo, seria possível criar um programa VBA para importar alguns números e depois formatar e imprimir o seu relatório de vendas do final do mês. Depois de desenvolver e testar o programa, você pode executar a macro com um único comando, levando o Excel a executar automaticamente muitos procedimentos demorados. Em vez de aprender uma cansativa sequência de comandos, você pode clicar um botão acessar o Facebook para matar o tempo enquanto a macro faz o trabalho.

As próximas seções descrevem em poucas palavras alguns usos comuns para as macros VBA. Uma ou duas farão você ficar esperto.

Inserindo um monte de texto

Se precisa inserir frequentemente o nome da empresa, endereço e número de telefone em suas planilhas de trabalho, é possível criar uma macro para digitar essas informações. Você pode estender esse conceito o quanto quiser. Por exemplo, pode desenvolver uma macro que digita automaticamente uma lista de todos os vendedores que trabalham para sua empresa.

Automatizando as tarefas executadas com frequência

Suponha que você seja o gerente de vendas e precisa preparar o relatório de vendas do fim do mês para satisfazer seu chefe. Se for uma tarefa direta, pode desenvolver um programa VBA para executá-la. O seu chefe ficará impressionado com a qualidade ótima e consistente dos seus relatórios e você poderá ser promovido a um novo cargo para o qual está altamente desqualificado.

Automatizando operações repetitivas

Se você precisa executar a mesma ação em, digamos, 12 diferentes pastas de trabalho do Excel, pode gravar uma macro enquanto realiza a tarefa na primeira e, então, deixar que a macro repita a sua ação nas outras pastas de trabalho. O melhor disso é que o Excel nunca reclama que está entediado. O gravador de macros do Excel é similar à gravação de uma ação ao vivo com um gravador de vídeo. Mas ele não requer uma câmera e a bateria nunca precisa ser recarregada.

Criando um comando personalizado

Você geralmente usa a mesma sequência de comandos no Excel? Se sim, poupe alguns segundos desenvolvendo uma macro que combina esses comandos em um único comando personalizado que pode ser executado pressionando uma única tecla, ou com um só clique de botão. Provavelmente, você não poupará *tanto* tempo, mas certamente será mais preciso. E o cara da divisória ao lado ficará realmente impressionado.

Criando um botão personalizado

Você pode personalizar a barra de ferramentas de Acesso Rápido com seus próprios botões, que executam macros escritas por você. Funcionários de escritório costumam ficar muito impressionados com botões que fazem mágica. E, se realmente quiser impressionar seus colegas, pode até adicionar novos botões à Faixa de Opções.

Desenvolvendo novas funções de planilha

Embora o Excel inclua centenas de funções integradas (tais como SOMA e MÉDIA), é possível criar funções da planilha de trabalho *personalizadas*, que podem simplificar muito as suas fórmulas. É surpreendente ver como isso é fácil (explore como fazer isso no Capítulo 20). E, melhor ainda, a caixa de diálogo Inserir Função exibe as suas funções personalizadas, fazendo com que elas pareçam integradas. Coisa muito boa.

Criando add-ins personalizados para o Excel

Você provavelmente está familiarizado com os add-ins do Excel. Por exemplo, o Analysis ToolPak (Pacote de Ferramentas de Análise) é um add-in popular. É possível usar o VBA para desenvolver os seus próprios add-ins especiais.

Vantagens e Desvantagens do VBA

Esta seção descreve as coisas boas do VBA, e seu lado sombrio também.

Vantagens do VBA

É possível automatizar quase tudo o que é feito no Excel, basta escrever instruções para ele executar. Automatizar uma tarefa usando o VBA tem muitas vantagens:

- O Excel sempre executa as tarefas exatamente do mesmo jeito (na maioria dos casos, a consistência é uma coisa boa).
- O Excel executa a tarefa muito mais depressa do que você pode fazer manualmente (a menos, é claro, que você seja o Clark Kent).
- Se você for um bom programador de macros, o Excel sempre executará as tarefas sem erros (o que não pode ser dito sobre você, não importa o quanto é cuidadoso).
- Se as coisas forem configuradas corretamente, qualquer um sem conhecimento em Excel poderá realizar a tarefa executando a macro.
- É possível fazer coisas em Excel que seriam impossíveis de outra maneira — o que pode torná-lo uma pessoa muito popular no escritório.
- Para as tarefas longas e demoradas, você não precisa ficar sentado e entediado diante do computador. O Excel faz o trabalho e você fica à toa.

Desvantagens do VBA

É justo que eu use o mesmo tempo para as desvantagens (ou *possíveis* desvantagens) do VBA:

- Você precisa saber como escrever programas em VBA (mas foi para isso que comprou este livro, certo?). Felizmente, não é tão complicado quanto você poderia esperar.
- Outras pessoas que precisem usar os seus programas VBA devem ter suas próprias cópias do Excel. Seria muito bom se um botão fosse pressionado e o aplicativo Excel/VBA se convertesse em um programa autônomo, mas isso não é possível (e provavelmente nunca será).
- Às vezes, as coisas dão errado. Em outras palavras, não é possível supor cegamente que o seu programa VBA sempre funcionará da maneira correta em todas as circunstâncias. Bem-vindo ao mundo da depuração e, se outros estiverem usando as suas macros, do suporte técnico.
- VBA é um alvo em movimento. Como se sabe, a Microsoft atualiza continuamente o Excel. Mesmo que a Microsoft se esforce para que haja compatibilidade entre as versões, você pode descobrir que o código VBA que escreveu não funciona adequadamente com as versões mais antigas ou com uma versão futura do Excel.

VBA Resumido

Só para que saiba onde está se metendo, veja um resumo rápido sobre VBA.

- **As ações são executadas em VBA escrevendo (ou gravando) código em um módulo VBA.** Você vê e edita os módulos VBA usando o Visual Basic Editor (VBE).
- **Um módulo VBA consiste de procedimentos Sub (secundários).** Um procedimento Sub nada tem a ver com submarinos ou sanduíches saborosos. Trata-se de um grupo de código de computador que realiza alguma ação em ou com objetos (a ser discutido em breve). O exemplo a seguir mostra um procedimento Sub simples, chamado UmMaisUm. Esse programa incrível, quando executado, exibe o resultado de 1 mais 1:

```
Sub UmMaisUm()
    Soma = 1 + 1
    MsgBox "A resposta é " & Soma
End Sub
```

  Um procedimento Sub que não é executado adequadamente é chamado de substandard (um padrão secundário).

- **Um módulo VBA também pode ter procedimentos Function.** Um procedimento Function retorna um único valor. É possível chamá-lo a partir de outro procedimento VBA ou até usá-lo como uma função em uma fórmula de planilha. A seguir está um exemplo de um procedimento Function (chamado de SomarDois). Essa Function aceita dois números (chamados de argumentos) e retorna a soma desses valores:

```
Function SomarDois(arg1, arg2)
    SomarDois = arg1 + arg2
End Function
```

  Um procedimento Function que não funciona corretamente é dito disfuncional.

- **O VBA manipula objetos.** O Excel oferece dezenas de objetos que podem ser manipulados. Exemplos de objetos incluem uma pasta de trabalho, uma planilha, um intervalo de células, um gráfico e uma forma. Há muitos outros objetos à sua disposição, e você pode manipulá-los usando um código VBA.
- **Os objetos são organizados em uma hierarquia.** Os objetos podem agir como *contêineres* para outros objetos. O Excel está no topo da hierarquia. O próprio Excel é um objeto chamado Application (Aplicação). O objeto Application contém outros objetos, tais como os objetos Pasta de Trabalho e Add-Ins. O objeto Pasta de Trabalho pode conter outros, como os objetos Planilhas e Gráficos. Um objeto Planilha pode conter objetos, como Range e

PivotTable. O termo *modelo de objeto* refere-se à organização desses objetos (mais detalhes sobre modelo de objeto podem ser encontrados no Capítulo 4).

- **Objetos do mesmo tipo formam uma coleção.** Por exemplo, a coleção Worksheets consiste em todas as planilhas em uma pasta de trabalho específica. A coleção Charts consiste em todos os objetos Gráficos em uma pasta de trabalho. As próprias coleções são objetos.
- **Você se refere a um objeto especificando sua posição na hierarquia de objetos, usando um ponto como separador.** Por exemplo, é possível fazer referência à pasta de trabalho Pasta1.xlsx como

```
Application.Workbooks("Pasta1.xlsx")
```

Isso se refere à pasta de trabalho `Pasta1.xlsx` na coleção Workbooks. Essa coleção está contida no objeto Application (ou seja, no Excel). Levando isso a outro nível, você pode se referir a Plan1 em `Pasta1.xlsx` como

```
Application.Workbooks("Pasta1.xlsx").Worksheets("Plan1")
```

Ainda é possível levar isso a outro nível e fazer referência a uma célula específica (nesse caso, a célula A1):

```
Application.Workbooks("Pasta1.xlsx").Worksheets("Plan1").
  Range("A1")
```

- **Se você omitir as referências específicas, o Excel usará os objetos *ativos*.** Se `Pasta1.xlsx` for a pasta de trabalho ativa, será possível simplificar a referência anterior como a seguir:

```
Worksheets("Pasta1").Range("A1")
```

Se souber que Plan1 é a planilha ativa, poderá simplificar ainda mais a referência:

```
Range("A1")
```

- **Os objetos têm propriedades.** É possível pensar em uma propriedade como uma *configuração* para um objeto. Por exemplo, um objeto Range tem propriedades como Value e Address. Um objeto Chart tem propriedades como HasTitle e Type. Você pode usar o VBA para determinar as propriedades do objeto e também alterar suas propriedades.
- **Uma propriedade de um objeto é referida pela combinação do nome do objeto com o nome da propriedade, separados por um ponto.** Por exemplo, você pode referir-se à propriedade Value na célula A1 em Plan1 como a seguir:

```
Worksheets("Plan1").Range("A1").Value
```

- **Você pode atribuir valores a variáveis.** *Variável* é um elemento nomeado que armazena informações. É possível usar variáveis em seu código VBA para armazenar coisas como valores, texto e configurações de propriedade. Para atribuir um valor na célula A1 em Plan1 a uma variável chamada *Juros*, use a seguinte declaração VBA:

```
Juros = Worksheets("Plan1").Range("A1").Value
```

- **Os objetos têm métodos.** *Método* é uma ação que o Excel executa com um objeto. Por exemplo, um dos métodos para um objeto Range é ClearContents. Esse método nomeado adequadamente limpa o conteúdo do intervalo.

- **Você especifica um método combinando o objeto com o método, separados por um ponto.** Por exemplo, a seguinte declaração limpa o conteúdo da célula A1:

```
Worksheets("Plan1").Range("A1").ClearContents
```

- **O VBA inclui todas as construções de linguagens modernas de programação, incluindo variáveis, arrays e looping.** Em outras palavras, se você estiver disposto a gastar um pouco de tempo aprendendo o assunto, poderá escrever códigos que fazem algumas coisas incríveis.

Acredite se quiser, a lista anterior descreve bem o VBA resumidamente. Agora você só precisa descobrir os detalhes. É por isso que este livro tem mais páginas.

Compatibilidade do Excel

LEMBRE-SE

Este livro foi escrito para as versões de desktop do Excel 2016 e Excel 2019. Se você não tiver uma dessas versões, correrá o risco de se confundir algumas vezes.

Se planeja distribuir seus arquivos Excel/VBA para outros usuários, é muito importante que entenda quais versões do Excel eles usam. Pessoas com versões anteriores não serão capazes de aproveitar os recursos inclusos nas versões posteriores. Por exemplo, se você escrever um código VBA que referencia a célula XFD1048576 (a última célula em uma pasta de trabalho), as pessoas que usam uma versão anterior ao Excel 2007 obterão um erro, pois as planilhas anteriores ao Excel 2007 tinham apenas 65.536 linhas e 255 colunas (a última célula é a IV65536).

O Excel 2010 e posteriores também têm objetos, métodos e propriedades novos. Se usá-los em seu código, os usuários com versões anteriores obterão um erro ao executar sua macro, e você levará a culpa.

NESTE CAPÍTULO

- Desenvolvendo uma macro VBA útil: um exemplo prático, passo a passo
- Gravando suas ações com o gravador de macros do Excel
- Examinando e testando o código gravado
- Mudando uma macro gravada
- Lidando com questões de segurança da macro

Capítulo 2
Mergulhando

A melhor maneira de entrar na água fria é mergulhar direto — não faz sentido prolongar a agonia. Percorrendo este capítulo, você molha os pés imediatamente, mas procure não se afogar.

Quando chegar ao final deste capítulo, você talvez comece a se sentir melhor em relação a essa coisa de programação em Excel, e ficará satisfeito de ter dado o mergulho. Este capítulo oferece uma demonstração passo a passo de como desenvolver uma macro VBA simples, mas útil.

Começando pelo Começo

Antes de se autoproclamar programador de Excel, você deve passar pelos rituais de iniciação. Isso significa que precisa fazer uma pequena mudança para que o Excel exiba uma nova guia no alto da tela: Desenvolvedor. Fazer com que o Excel exiba a guia Desenvolvedor é fácil (e só precisa ser feito uma vez). É só seguir estes passos:

1. **Clique com o botão direito em qualquer parte da Faixa de Opções e escolha Personalizar a Faixa de Opções no menu de atalho.**

2. **Na guia Personalizar Faixa de Opções da caixa de diálogo Opções do Excel, localize Desenvolvedor na segunda coluna.**

3. **Marque a caixa ao lado de Desenvolvedor.**

4. **Clique em Ok.**

   Você está de volta ao Excel com uma guia nova em folha: Desenvolvedor.

Ao clicar na guia Desenvolvedor, a Faixa de Opções exibe informações de interesse dos programadores (este é você!). A Figura 2-1 mostra como a Faixa de Opções fica quando a guia Desenvolvedor é selecionada no Excel 2019.

FIGURA 2-1: Normalmente, a guia Desenvolvedor está oculta, mas é fácil exibi-la.

O que Você Fará

Neste capítulo você criará sua primeira macro, que fará o seguinte:

- Digita o seu nome em uma célula
- Insere a data e a hora atuais na célula abaixo
- Formata ambas as células para exibir em negrito
- Muda o tamanho da fonte de ambas as células para 16

Essa macro não ganhará nenhum prêmio da Competição Anual de Programação VBA, mas precisamos começar de algum lugar. A macro realiza todos esses passos em uma única ação. Como descrito nas seções seguintes, começamos gravando as ações ao realizar esses passos. Depois, testamos a macro para ver se funciona. Por fim, editamos a macro para dar os retoques finais. Pronto?

Dando os Primeiros Passos

Esta seção descreve os passos que você deve seguir antes de gravar a macro. Em outras palavras, precisa se preparar antes de começar a diversão:

1. **Inicie o Excel, se ele ainda não estiver em execução.**

2. **Se necessário, crie uma nova pasta de trabalho vazia.**

   Pressionar Ctrl+N é a maneira preferida e rápida de fazer isso.

3. **Clique na guia Desenvolvedor e dê uma olhada no botão Usar Referências Relativas no grupo Código.**

   Se a cor desse botão for diferente da dos outros, quer dizer que está tudo certo. Se o botão Usar Referências Relativas for da mesma cor dos outros botões, então será preciso clicá-lo para habilitar essa opção.

Exploramos o botão Usar Referências Relativas no Capítulo 6. Por ora, apenas garanta que a opção esteja ativada. Quando estiver, o botão Usar Referências Relativas terá uma cor diferente.

Gravando a Macro

Aqui está a parte prática. Siga estas instruções cuidadosamente:

1. **Selecione uma célula.**

   Qualquer célula serve.

2. **Selecione Desenvolvedor ➪ Código ➪ Gravar Macro ou clique no botão de gravação de macro na barra de status.**

   A caixa de diálogo Gravar Macro aparece, conforme mostrado na Figura 2-2.

3. **Insira um nome para a macro.**

   O Excel oferece um nome padrão (algo como *Macro1*), mas é melhor usar um nome mais descritivo. *NomeEData* (sem espaços) é um bom nome para essa macro.

4. **Clique na caixa Tecla de Atalho e insira Shift+N (para um N maiúsculo), como a tecla de atalho.**

   Especificar uma tecla de atalho é opcional. Se você especificar uma, poderá executar a macro pressionando uma combinação de teclas; nesse caso, Ctrl+Shift+N. Saiba que, ao atribuir uma tecla de atalho comum (por exemplo, Ctrl+C), você perderá a funcionalidade normal desse atalho e o Excel iniciará a macro em seu lugar.

5. **Verifique se a configuração Armazenar Macro Em é Esta Pasta de Trabalho.**

6. **Se quiser, você pode inserir algum texto na caixa Descrição.**

Esse passo é opcional. Algumas pessoas gostam de descrever o que a macro faz (ou o que *supostamente* deve fazer).

7. **Clique em OK.**

   A caixa de diálogo Gravar Macro fecha e o gravador de macro do Excel é ativado. A partir desse ponto, o Excel monitora tudo o que você faz e converte no código VBA.

8. **Digite seu nome na célula ativa.**

9. **Mova o indicador da célula para a célula abaixo e insira a fórmula:**

   ```
   =AGORA()
   ```

   A fórmula exibe a data e a hora atuais.

10. **Selecione a célula com a fórmula e pressione Ctrl+C para copiar a célula para a Área de Transferência.**

11. **Selecione Página Inicial ➪ Área de Transferência ➪ Colar ➪ Colar Valores.**

    Esse comando converte a fórmula em seus valores.

12. **Com a célula de data selecionada, pressione Shift+seta para cima para selecionar aquela célula e uma acima dela (que contém o seu nome).**

13. **Use os controles no grupo Página Inicial ➪ Fonte para mudar a formatação para Negrito e o tamanho da fonte para 16.**

14. **Escolha Desenvolvedor ➪ Código ➪ Parar Gravação.**

    O gravador de macro é desativado.

FIGURA 2-2: A caixa de diálogo Gravar Macro aparecerá quando você for gravar uma macro.

Parabéns! Você acabou de criar sua primeira macro VBA no Excel. Talvez queira ligar para sua mãe e contar a boa notícia.

Testando a Macro

Agora, teste a macro e veja se ela funciona corretamente. Para testá-la, clique em uma célula vazia e pressione Ctrl+Shift+N (ou qualquer atalho que tenha inserido).

Em um piscar de olhos, o Excel executará a macro. Seu nome e a data atual aparecerão em letras grandes e negrito.


DICA

Outro modo de executar a macro é selecionar Desenvolvedor ➪ Código ➪ Macros (ou pressionar Alt+F8) para exibir a caixa de diálogo Macros. Selecione a macro da lista (nesse caso, NomeEData) e clique em Executar. Selecione a célula que conterá o seu nome antes de executar a macro.

Examinando a Macro

Você gravou e testou uma macro. Se for do tipo curioso, provavelmente está imaginando a aparência dessa macro, e pode até imaginar onde está armazenada.

Lembra quando começou a gravar a macro? Você indicou que o Excel deveria armazená-la em Esta Pasta de Trabalho. A macro está armazenada na pasta de trabalho, mas é necessário ativar o Visual Basic Editor (VBE) para vê-la.

Siga estes passos para ver a macro:

1. **Selecione Desenvolvedor ➪ Código ➪ Visual Basic (ou pressione Alt+F11).**

   A janela do programa Visual Basic Editor aparece, conforme mostrado na Figura 2-3. Essa janela é altamente personalizável, portanto, a sua janela VBE pode ser um pouco diferente. A janela do VBE contém várias outras janelas e isso, provavelmente, é muito assustador. Não se aflija; você se acostumará.

2. **Na janela VBE, localize a janela chamada Projeto.**

   A janela Projeto (também conhecida como janela Project Explorer) contém uma lista de todas as planilhas e add-ins abertos no momento. Cada projeto é organizado como uma *árvore* e pode ser expandido (para mostrar mais informações) ou contraído (para mostrar menos).

DICA

O VBE usa algumas janelas diferentes, qualquer delas podendo ser aberta ou fechada. Se uma janela não estiver imediatamente visível no VBE, escolha uma opção no menu Exibir para mostrá-la. Por exemplo, se a janela Projeto não estiver visível, é possível escolher Exibir ➪ Project Explorer (ou pressionar Ctrl+R) para exibi-la. Você pode exibir qualquer outra janela do VBE da mesma forma. Os componentes do VBE são tratados no Capítulo 3.

3. Selecione o projeto que corresponde à pasta de trabalho onde gravou a macro.

Se você não salvou a pasta de trabalho, o projeto provavelmente tem o nome VBAProject (Pasta1).

4. Clique no sinal de adição (+) à esquerda da pasta chamada Módulos.

A árvore se expande para mostrar Módulo1, que é o único módulo no projeto.

5. Clique duas vezes em Módulo1.

O código VBA nesse módulo é exibido em uma janela Código (veja a Figura 2-3). A sua tela pode não ser exatamente igual. O código gravado depende de ações específicas realizadas ao gravar a macro.

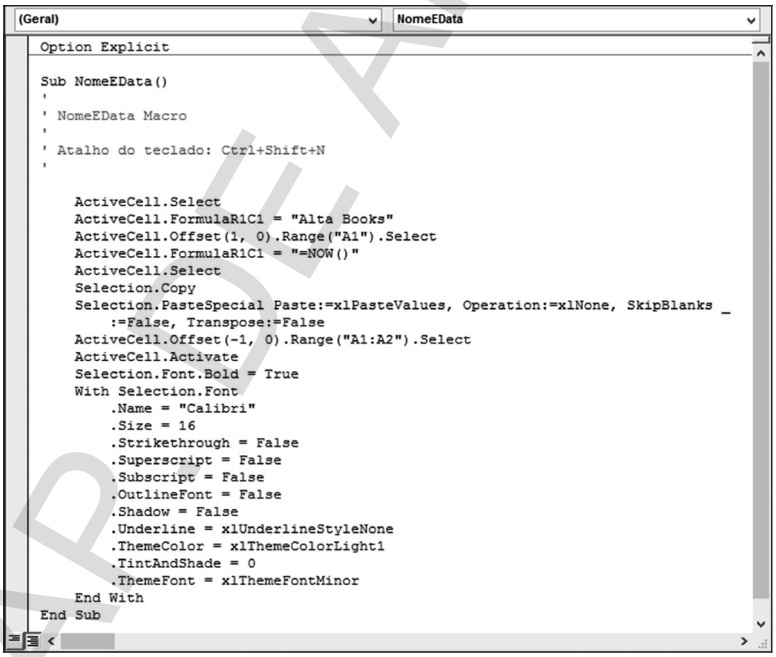

FIGURA 2-3: O VBE exibe o código VBA em Módulo1 da Pasta1.

A essa altura, a macro provavelmente parece grego para você. Não se preocupe. Viaje por alguns capítulos e tudo ficará tão claro quanto a vista do Olimpo.

A macro NomeEData consiste em várias instruções. O Excel executa as instruções uma por uma, de cima para baixo. Uma instrução precedida por um apóstrofo (') é um comentário. Os comentários são incluídos apenas para sua informação e são ignorados pelo Excel. Em outras palavras, o Excel passa direto por eles.

A primeira instrução VBA (que começa com a palavra *Sub*) identifica a macro como um procedimento Sub e dá o seu nome; você forneceu esse nome antes de começar a gravar a macro. Se ler cuidadosamente o código, será capaz de entendê-lo um pouco. Você verá o seu nome, a fórmula que forneceu e muito do código adicional que altera a fonte. O procedimento Sub termina com a instrução End Sub.

Modificando a Macro

Como seria esperado, você pode não apenas ver a sua macro no VBE, como também pode alterá-la. Mesmo que a essa altura não tenha ideia do que está fazendo, estas são mudanças fáceis de fazer:

- Mudar o nome fornecido na célula ativa. Se você tiver um cachorro, use o nome dele.
- Mudar o nome ou o tamanho da fonte.
- Veja se você descobre um lugar adequado para essa nova instrução que deixa a célula em itálico:

```
Selection.Font.Italic = True
```

EI, EU NÃO GRAVEI ISSO!

Gravar uma macro é como gravar som em um gravador. Quando você coloca para tocar e escuta sua própria voz, invariavelmente diz: "Minha voz não é essa". E quando olhar sua gravação de macro, talvez veja algumas ações que não achou que tinha gravado.

Ao gravar o exemplo NomeEData, você mudou apenas o tamanho da fonte, ainda que o código gravado exiba todos os tipos de instruções de mudança de fonte (Strikethrough, Superscript, Shadow e assim por diante). Não se preocupe; isso acontece o tempo todo. O Excel costuma gravar muitos códigos aparentemente inúteis. Nos capítulos posteriores, você verá como remover o código extra de uma macro gravada.

DICA

Trabalhar com um módulo de código VBA é quase como trabalhar com um documento em um processador de textos (exceto que não há quebra de linha e não é possível formatar o texto). Pensando melhor, é mais como trabalhar no Bloco de Notas do Windows. Você pode pressionar Enter para iniciar uma nova linha e as teclas de edição conhecidas funcionam conforme o esperado.

Depois de ter feito as alterações, volte para o Excel e experimente a macro revisada para ver como ela funciona. Assim como você pode pressionar Alt+F11 no Excel para exibir o VBE, pode pressionar Alt+F11 no VBE para voltar ao Excel.

Salvando Pastas de Trabalho que Contêm Macros

Se você armazena uma ou mais macros em uma pasta de trabalho, o arquivo deve ser salvo como um tipo de arquivo com *macros habilitadas*. Em outras palavras, o arquivo deve ser salvo com uma extensão XLSM, em vez da extensão XLSX normal.

Por exemplo, quando você salva a pasta de trabalho que contém a sua macro NomeEData, o formato de arquivo na caixa de diálogo Salvar Como padroniza para XLSX (um formato que não pode conter macros). A menos que mude o formato de arquivo para XLSM, o Excel exibirá o aviso mostrado na Figura 2-4. Clique em Não e depois escolha Pasta de Trabalho Habilitada para Macro do Excel (*.xlsm) na lista suspensa Tipo em Salvar Como.

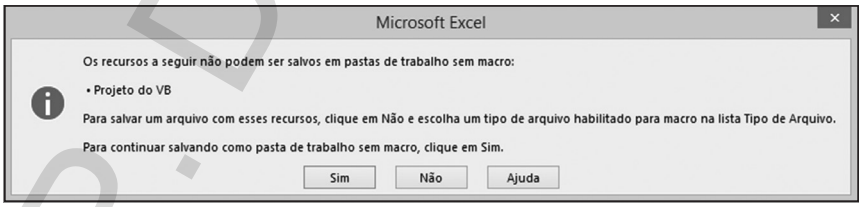

FIGURA 2-4: Se sua pasta de trabalho contiver macros e você tentar salvá-la em um formato de arquivo sem macro, o Excel o avisará.

Entendendo a Segurança de Macro

Segurança de macro é um recurso importante no Excel. Isso porque o VBA é uma linguagem poderosa, tão poderosa que é possível criar uma macro que pode causar sérios danos ao computador. Uma macro pode apagar arquivos, enviar informações a outros computadores e até destruir o Windows, de modo que você não poderá sequer iniciar o seu sistema.

Os recursos de segurança da macro introduzidos no Excel 2007 foram criados para ajudar a evitar tais tipos de problemas.

A Figura 2-5 mostra a seção Configurações de Macro da caixa de diálogo Central de Confiabilidade. Para exibir essa caixa de diálogo, escolha Desenvolvedor ➪ Código ➪ Segurança de Macro.

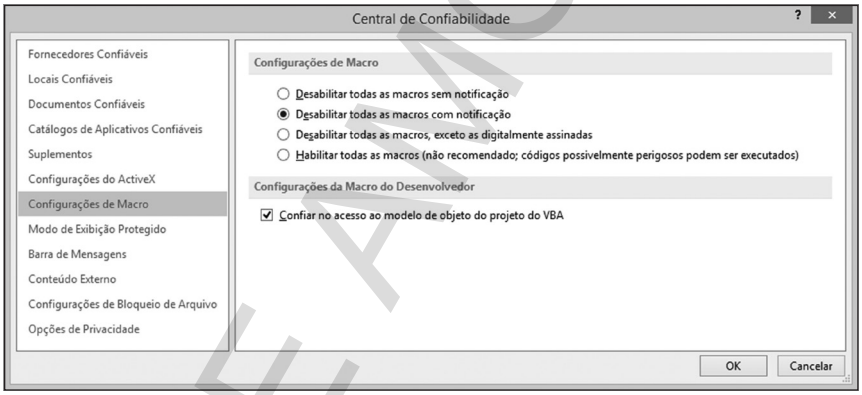

FIGURA 2-5: A seção Configurações de Macro da caixa de diálogo Central de Confiabilidade.

Por padrão, o Excel usa a opção Desabilitar Todas as Macros com Notificação. Com essa configuração ativa, ao abrir uma pasta de trabalho que contenha macros (e o arquivo não estiver digitalmente "assinado" ou armazenado em um local confiável), o Excel exibe um aviso como o da Figura 2-6. Se tiver certeza de que a pasta de trabalho vem de uma fonte confiável, clique em Habilitar Macros, e as macros serão habilitadas.

FIGURA 2-6: O aviso do Excel de que o arquivo aberto contém macros.

LEMBRE-SE

A caixa pop-up da Figura 2-6 só será visualizada se o VBE estiver aberto. Caso contrário, o Excel exibirá um de aviso de segurança (Security Warning) que chama a atenção acima da barra de Fórmula, como mostrado na Figura 2-7. Se você sabe que a pasta de trabalho é segura, clique no botão Habilitar Conteúdo para habilitar as macros. Para usar a pasta de trabalho sem macros, clique no X para dispensar o aviso.

O Excel lembrará se você designou uma pasta de trabalho como segura, para que não veja o Aviso de Segurança na próxima vez que abri-la.

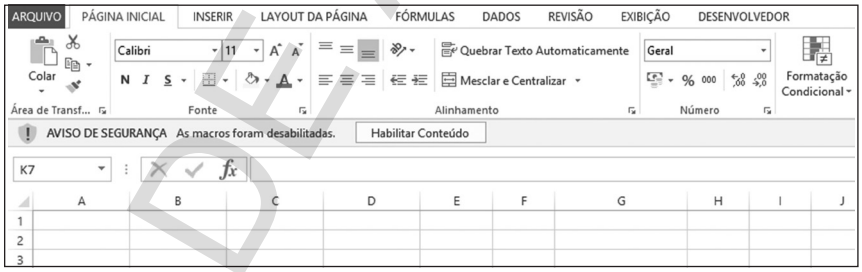

FIGURA 2-7: O aviso do Excel de que a pasta de trabalho recém-aberta contém macros. Você verá esse aviso se o VBE não estiver aberto.

Talvez a melhor maneira de lidar com a segurança da macro seja designar uma ou mais pastas como *locais confiáveis*. Todas as pastas de trabalho em um local confiável são abertas sem um aviso de macro. As pastas confiáveis são designadas na seção Locais Confiáveis, na caixa de diálogo Central de Confiabilidade.

Se quiser descobrir o que significam as outras configurações de segurança, pressione F1 com a seção Configurações de Macro da caixa de diálogo Central de Confiabilidade aberta. Você verá a tela de Ajuda que descreve as configurações de segurança.

Revelando Mais sobre a Macro NomeEData

Quando terminar este livro, entenderá completamente como a macro NomeEData funciona, e estará preparado para desenvolver macros mais sofisticadas. Por enquanto, este capítulo se encerra com alguns pontos adicionais sobre a macro:

- Para a macro funcionar, sua pasta de trabalho deve estar aberta. Se você fechar a pasta de trabalho, a macro não funcionará (e seu atalho não terá efeito).
- Desde que a pasta de trabalho contendo a macro esteja aberta, você pode rodar a macro enquanto qualquer pasta de trabalho estiver ativa. Em outras palavras, a pasta de trabalho da própria macro não precisa estar ativa.
- A macro não é código de "pró-qualidade". Ela sobrescreverá o texto existente sem aviso algum, e seus efeitos não poderão ser desfeitos.
- Antes de começar a gravar a macro, você designou a ela uma nova tecla de atalho. Essa é apenas uma de várias maneiras de executar a macro (descubra outras maneiras no Capítulo 5).
- É possível criar a macro manualmente em vez de gravá-la. Para tanto, é necessário um bom entendimento de VBA (tenha paciência, você chegará lá).
- É possível armazenar a macro em sua Pasta de Trabalho de Macros Pessoal. Se o fizer, a macro estará automaticamente disponível sempre que iniciar o Excel (veja o Capítulo 6 para obter detalhes sobre a Pasta de Trabalho de Macro Pessoal).
- Também é possível converter a pasta de trabalho a um arquivo add-in (mais sobre isso no Capítulo 21).

Parabéns. Você foi iniciado no mundo de programação do Excel. (Lamento, não há um aperto de mão secreto ou anel decodificador.)