



Bill Jelen  
Tracy Syrstad

# **VBA i makra w Excel® 2016**

Przekład: Leszek Biolik

APN Promise  
Warszawa 2016

VBA i makra w Excel® 2016

Authorized Polish translation of the English language edition entitled: Excel® 2016 VBA and Macros, ISBN: 978-0-7897-5585-8, by Bill Jelen and Tracy Syrstad, published by Pearson Education, Inc., publishing as Que Publishing.

Copyright © 2016 by Pearson Education, Inc.

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from Pearson Education, Inc.

Polish language edition published by APN PROMISE SA Copyright © 2016

Autoryzowany przekład z wydania w języku angielskim, zatytułowanego: Excel® 2016 VBA and Macros, ISBN: 978-0-7897-5585-8, by Bill Jelen and Tracy Syrstad, opublikowanego przez Pearson Education, Inc, publikującego jako Que Publishing.

Wszystkie prawa zastrzeżone. Żadna część niniejszej książki nie może być powielana ani rozpowszechniana w jakiejkolwiek formie i w jakikolwiek sposób (elektroniczny, mechaniczny), włącznie z fotokopiowaniem, nagrywaniem na taśmy lub przy użyciu innych systemów bez pisemnej zgody wydawcy.

APN PROMISE SA, ul. Domaniewska 44a, 02-672 Warszawa

tel. +48 22 35 51 600, fax +48 22 35 51 699

e-mail: mspress@promise.pl

Książka ta przedstawia poglądy i opinie autorów. Przykłady firm, produktów, osób i wydarzeń opisane w niniejszej książce są fikcyjne i nie odnoszą się do żadnych konkretnych firm, produktów, osób i wydarzeń, chyba że zostanie jednoznacznie stwierdzone, że jest inaczej. Ewentualne podobieństwo do jakiejkolwiek rzeczywistej firmy, organizacji, produktu, nazwy domeny, adresu poczty elektronicznej, logo, osoby, miejsca lub zdarzenia jest przypadkowe i niezamierzone.

Wszystkie nazwy i znaki towarowe wymienione w książce mogą być zastrzeżonymi znakami towarowymi ich odnośnych właścicieli i zostały użyte w celach identyfikacyjnych.

APN PROMISE SA dołożyła wszelkich starań, aby zapewnić najwyższą jakość tej publikacji. Jednakże nikomu nie udziela się rękojmi ani gwarancji.

APN PROMISE SA nie jest w żadnym wypadku odpowiedzialna za jakiegokolwiek szkody będące następstwem korzystania z informacji zawartych w niniejszej publikacji, nawet jeśli APN PROMISE została powiadomiona o możliwości wystąpienia szkód.

ISBN: 978-83-7541-211-6

Przekład: Leszek Biolik

Redakcja: Marek Włodarz

Korekta: Ewa Świędrowska

Skład i łamanie: MAWart Marek Włodarz

# Spis treści

|                                                                                       |          |
|---------------------------------------------------------------------------------------|----------|
| <b>Wprowadzenie</b>                                                                   | <b>1</b> |
| Co zawiera niniejsza książka?                                                         | 1        |
| Skrócenie procesu przyswajania wiedzy                                                 | 1        |
| Możliwości programu Excel VBA                                                         | 2        |
| Materiały naukowe potrzebne do tworzenia aplikacji                                    | 2        |
| Czy książka ta uczy programu Excel?                                                   | 2        |
| Funkcje VBA a wersje programu Excel w systemie Windows                                | 5        |
| Wersje programu Excel                                                                 | 5        |
| Różnice dla użytkowników systemu Mac                                                  | 5        |
| Elementy specjalne i konwencje typograficzne                                          | 5        |
| Pliki kodu                                                                            | 6        |
| Następne kroki                                                                        | 6        |
| <b>1 Zwiększanie możliwości programu Excel za pomocą języka VBA</b>                   | <b>7</b> |
| Możliwości Excela                                                                     | 7        |
| Początkowe przeszkody                                                                 | 7        |
| Rejestrator makr nie działa!                                                          | 7        |
| Nikt w zespole programu Excel nie poświęca wiele uwagi rejestratorowi makr            | 8        |
| Visual Basic nie przypomina języka BASIC                                              | 8        |
| Dobra wiadomość: „wspinaczka” związana z poznaniem języka VBA nie jest trudna         | 9        |
| Doskonała wiadomość: program Excel wraz z językiem VBA wart jest włożonego wysiłku    | 9        |
| Znajomość narzędzi: karta Deweloper                                                   | 10       |
| Typy plików, dla których dopuszczane są makra                                         | 11       |
| Bezpieczeństwo makr                                                                   | 12       |
| Dodawanie zaufanej lokalizacji                                                        | 13       |
| Zastosowanie ustawień makr w celu włączenia obsługi makr poza zaufanymi lokalizacjami | 14       |
| Stosowanie opcji Wyłącz wszystkie makra i wyświetl powiadomienie                      | 15       |
| Przegląd informacji dotyczących rejestrowania, zapisywania i uruchamiania makr        | 15       |
| Wypełnianie okna dialogowego Rejestrowanie makra                                      | 16       |
| Uruchamianie makra                                                                    | 17       |
| Tworzenie przycisku makra na wstążce                                                  | 17       |
| Tworzenie przycisku makra na pasku narzędzi Szybki dostęp                             | 18       |
| Przypisywanie makra do kontrolki formularza, pola tekstowego lub kształtu             | 19       |
| Działanie edytora Visual Basic                                                        | 21       |
| Ustawienia narzędzia VB Editor                                                        | 21       |
| Eksplorator projektu                                                                  | 22       |
| Okno Properties                                                                       | 22       |
| Mankamenty rejestratora makr                                                          | 23       |
| Rejestrowanie makra                                                                   | 24       |
| Analiza kodu w oknie programowania                                                    | 25       |
| Sprawdzanie każdego makra                                                             | 27       |

|                                                                                                          |           |
|----------------------------------------------------------------------------------------------------------|-----------|
| Uruchomienie tego samego makra innego dnia generuje nieoczekiwane wyniki . . . . .                       | 27        |
| Możliwe rozwiązanie: wykorzystywanie odwołań względnych podczas rejestrowania . . . . .                  | 28        |
| Podczas rejestrowania nigdy nie używaj przycisku Autosumowanie lub Szybka analiza . . . . .              | 32        |
| Cztery wskazówki dotyczące używania rejestratora makr . . . . .                                          | 33        |
| Następne kroki . . . . .                                                                                 | 34        |
| <b>2 Skoro nazywa się BASIC, dlaczego nie wygląda znajomo? . . . . .</b>                                 | <b>35</b> |
| Nie rozumiem tego kodu! . . . . .                                                                        | 35        |
| „Części mowy” języka VBA . . . . .                                                                       | 36        |
| Język VBA nie jest bardzo trudny . . . . .                                                               | 40        |
| Pliki pomocy VBA: Stosowanie klawisza F1 do wyszukiwania potrzebnych informacji . . . . .                | 40        |
| Korzystanie z pomocy . . . . .                                                                           | 40        |
| Analiza kodu zarejestrowanego makra: korzystanie z edytora VB i tematów pomocy . . . . .                 | 41        |
| Parametry opcjonalne . . . . .                                                                           | 42        |
| Zdefiniowane stałe . . . . .                                                                             | 43        |
| Właściwości mogą zwracać obiekty . . . . .                                                               | 46        |
| Stosowanie narzędzi debugowania do analizy zarejestrowanego kodu . . . . .                               | 47        |
| Krokowe wykonywanie kodu . . . . .                                                                       | 47        |
| Inne opcje debugowania: punkty przerywania . . . . .                                                     | 48        |
| Poruszanie się w kodzie w przód lub w tył . . . . .                                                      | 49        |
| Uruchamianie fragmentu kodu bez trybu krokowego . . . . .                                                | 49        |
| Tworzenie zapytań podczas krokowego wykonywania kodu . . . . .                                           | 49        |
| Wykorzystywanie czujek do ustawiania punktów przerywań . . . . .                                         | 52        |
| Stosowanie czujek do obiektów . . . . .                                                                  | 53        |
| Narzędzie Object Browser: odwołanie ostateczne . . . . .                                                 | 54        |
| Siedem wskazówek poprawiania zarejestrowanego kodu . . . . .                                             | 55        |
| Wskazówka 1: Niczego nie zaznaczaj . . . . .                                                             | 55        |
| Wskazówka 2: Używaj Cells(2,5), ponieważ jest to znacznie bardziej praktyczne, niż Range("E2") . . . . . | 56        |
| Wskazówka 3: Używaj bardziej niezawodnych sposobów wyszukiwania ostatniego wiersza . . . . .             | 57        |
| Wskazówka 4: Stosuj zmienne, by unikać „sztywnego” kodowania wierszy i formuł . . . . .                  | 58        |
| Wskazówka 5: Używaj formuł typu R1C1, które ułatwiają życie . . . . .                                    | 58        |
| Wskazówka 6: Kopiuj i wklejaj w pojedynczej instrukcji . . . . .                                         | 58        |
| Wskazówka 7: Używaj konstrukcji With...End With do wykonywania wielu działań . . . . .                   | 59        |
| Następne kroki . . . . .                                                                                 | 62        |
| <b>3 Odwoływanie się do zakresów . . . . .</b>                                                           | <b>63</b> |
| Obiekt Range . . . . .                                                                                   | 63        |
| Składnia specyfikowania zakresu . . . . .                                                                | 64        |
| Zakresy nazwane . . . . .                                                                                | 64        |
| Skróty stosowane przy odwołaniach do zakresów . . . . .                                                  | 65        |
| Odwoływanie się do zakresów w innych arkuszach . . . . .                                                 | 65        |
| Odwoływanie się do zakresu względem innego zakresu . . . . .                                             | 66        |
| Stosowanie właściwości Cells do zaznaczania zakresu . . . . .                                            | 67        |
| Stosowanie właściwości Offset do odwoływania się do zakresu . . . . .                                    | 68        |
| Używanie właściwości Resize do zmiany rozmiaru zakresu . . . . .                                         | 70        |
| Stosowanie właściwości Columns i Rows do określania zakresu . . . . .                                    | 71        |
| Stosowanie metody Union do łączenia wielu zakresów . . . . .                                             | 71        |

|                                                                                                              |            |
|--------------------------------------------------------------------------------------------------------------|------------|
| Stosowanie metody <b>Intersect</b> do tworzenia nowego zakresu na podstawie nakładających się zakresów . . . | 72         |
| Stosowanie funkcji <b>IsEmpty</b> do sprawdzenia, czy komórka jest pusta . . . . .                           | 72         |
| Stosowanie właściwości <b>CurrentRegion</b> do zaznaczania zakresu danych Data . . . . .                     | 73         |
| Stosowanie kolekcji <b>Areas</b> do zwracania nieciągłego zakresu . . . . .                                  | 76         |
| Odwołania do tabel . . . . .                                                                                 | 77         |
| Następne kroki . . . . .                                                                                     | 78         |
| <b>4 Pętle i sterowanie przepływem . . . . .</b>                                                             | <b>79</b>  |
| Pętla <b>For . . . Next</b> . . . . .                                                                        | 79         |
| Stosowanie zmiennych w instrukcji <b>For</b> . . . . .                                                       | 81         |
| Uwagi dotyczące pętli <b>For . . . Next</b> . . . . .                                                        | 82         |
| Wcześniejsze przerywanie pętli po spełnieniu warunku . . . . .                                               | 83         |
| Zagnieżdżanie pętli . . . . .                                                                                | 84         |
| Pętla <b>Do</b> . . . . .                                                                                    | 85         |
| Stosowanie klauzul <b>While</b> lub <b>Until</b> w pętlach <b>Do</b> . . . . .                               | 87         |
| Pętla VBA: <b>For Each</b> . . . . .                                                                         | 89         |
| Zmienne obiektowe . . . . .                                                                                  | 89         |
| Sterowanie przepływem: stosowanie konstrukcji <b>If . . . Then . . . Else</b> i <b>Select Case</b> . . . . . | 92         |
| Podstawowe sterowanie przepływem: <b>If . . . Then . . . Else</b> . . . . .                                  | 93         |
| Stosowanie konstrukcji <b>If . . . ElseIf . . . End If</b> dla wielu warunków . . . . .                      | 94         |
| Stosowanie konstrukcji <b>Select Case . . . End Select</b> dla wielu warunków . . . . .                      | 95         |
| Następne korki . . . . .                                                                                     | 98         |
| <b>5 Formuły w stylu R1C1 . . . . .</b>                                                                      | <b>99</b>  |
| Odwoływanie się do komórek: porównanie odwołań typu A1 i R1C1 . . . . .                                      | 99         |
| Zmiana odwołań na styl R1C1 . . . . .                                                                        | 100        |
| Magia formuł programu Excel . . . . .                                                                        | 101        |
| Wprowadź formułę raz i skopiuj ją 1000 razy! . . . . .                                                       | 101        |
| Nie ma w tym niczego nadzwyczajnego . . . . .                                                                | 102        |
| Omówienie stylu odwołań R1C1 . . . . .                                                                       | 104        |
| Używanie stylu R1C1 dla odwołań względnych . . . . .                                                         | 104        |
| Stosowanie stylu R1C1 dla odwołań bezwzględnych . . . . .                                                    | 105        |
| Stosowanie notacji R1C1 przy odwołaniach mieszanych . . . . .                                                | 105        |
| Odwoływanie się do całych kolumn lub wierszy za pomocą notacji R1C1 . . . . .                                | 106        |
| Zastępowanie wielu formuł A1 za pomocą pojedynczej formuły R1C1 . . . . .                                    | 106        |
| Pamiętanie numerów kolumn powiązanych z literą kolumn . . . . .                                              | 108        |
| Stosowanie formuł R1C1 w formułach tablicowych . . . . .                                                     | 108        |
| Następne kroki . . . . .                                                                                     | 109        |
| <b>6 Tworzenie nazw i operacje na nazwach w VBA . . . . .</b>                                                | <b>111</b> |
| Porównanie nazw globalnych i lokalnych . . . . .                                                             | 111        |
| Dodawanie nazw . . . . .                                                                                     | 112        |
| Usuwanie nazw . . . . .                                                                                      | 114        |
| Dodawanie komentarzy . . . . .                                                                               | 114        |
| Typy nazw . . . . .                                                                                          | 115        |
| Formuły . . . . .                                                                                            | 115        |
| Ciągi . . . . .                                                                                              | 115        |

|          |                                                                                                |            |
|----------|------------------------------------------------------------------------------------------------|------------|
|          | Liczby .....                                                                                   | 117        |
|          | Tabele .....                                                                                   | 117        |
|          | Używanie tablic w nazwach .....                                                                | 118        |
|          | Nazwy zastrzeżone .....                                                                        | 119        |
|          | Ukrywanie nazw .....                                                                           | 120        |
|          | Sprawdzanie istnienia nazwy .....                                                              | 120        |
|          | Następne kroki .....                                                                           | 123        |
| <b>7</b> | <b>Programowanie zdarzeń .....</b>                                                             | <b>125</b> |
|          | Poziomy zdarzeń .....                                                                          | 125        |
|          | Stosowanie zdarzeń .....                                                                       | 126        |
|          | Parametry zdarzenia .....                                                                      | 127        |
|          | Włączanie zdarzeń .....                                                                        | 127        |
|          | Zdarzenia dotyczące skoroszytu .....                                                           | 127        |
|          | Zdarzenia dotyczące arkusza i wykresu na poziomie skoroszytu .....                             | 130        |
|          | Zdarzenia dotyczące arkusza .....                                                              | 131        |
|          | Zdarzenia dotyczące wykresów .....                                                             | 134        |
|          | Wykresy osadzone .....                                                                         | 134        |
|          | Zdarzenia dotyczące wykresu osadzonego i arkusza wykresu .....                                 | 135        |
|          | Zdarzenia na poziomie aplikacji .....                                                          | 136        |
|          | Następne kroki .....                                                                           | 143        |
| <b>8</b> | <b>Tablice .....</b>                                                                           | <b>145</b> |
|          | Deklarowanie tablicy .....                                                                     | 145        |
|          | Deklarowanie wielowymiarowej tablicy .....                                                     | 146        |
|          | Wypełnianie tablicy .....                                                                      | 147        |
|          | Pobieranie danych z tablicy .....                                                              | 148        |
|          | Wykorzystywanie tablic do przyspieszenia działania kodu .....                                  | 150        |
|          | Wykorzystywanie tablic dynamicznych .....                                                      | 150        |
|          | Przekazywanie tablicy .....                                                                    | 152        |
|          | Następne kroki .....                                                                           | 152        |
| <b>9</b> | <b>Tworzenie klas, rekordów i kolekcji .....</b>                                               | <b>153</b> |
|          | Wstawianie modułu klasy .....                                                                  | 153        |
|          | Śledzenie zdarzeń dotyczących aplikacji i wykresów osadzonych .....                            | 154        |
|          | Zdarzenia dotyczące aplikacji .....                                                            | 154        |
|          | Zdarzenia dotyczące wykresów osadzonych .....                                                  | 156        |
|          | Tworzenie obiektu niestandardowego .....                                                       | 157        |
|          | Stosowanie obiektu niestandardowego .....                                                      | 159        |
|          | Wykorzystywanie kolekcji .....                                                                 | 160        |
|          | Tworzenie kolekcji .....                                                                       | 160        |
|          | Tworzenie kolekcji w module standardowym .....                                                 | 161        |
|          | Tworzenie kolekcji w module klasy .....                                                        | 163        |
|          | Stosowanie słowników .....                                                                     | 165        |
|          | Stosowanie typów zdefiniowanych przez użytkownika do tworzenia właściwości niestandardowych .. | 169        |
|          | Następne kroki .....                                                                           | 172        |

|           |                                                                                                             |            |
|-----------|-------------------------------------------------------------------------------------------------------------|------------|
| <b>10</b> | <b>Obiekty UserForm – wprowadzenie</b>                                                                      | <b>173</b> |
|           | Pola wprowadzania danych                                                                                    | 173        |
|           | Pola komunikatów                                                                                            | 174        |
|           | Tworzenie formularza użytkownika                                                                            | 174        |
|           | Wywoływanie i ukrywanie formularza użytkownika                                                              | 176        |
|           | Programowanie formularzy użytkownika                                                                        | 176        |
|           | Zdarzenia związane z formularzami użytkownika                                                               | 176        |
|           | Programowanie kontrolek                                                                                     | 178        |
|           | Stosowanie podstawowych kontrolek formularza                                                                | 179        |
|           | Stosowanie etykiet, pól tekstowych i przycisków poleceń                                                     | 180        |
|           | Wybór pomiędzy polami list a polami kombi w formularzach                                                    | 182        |
|           | Dodawanie przycisków opcji do formularza użytkownika                                                        | 184        |
|           | Dodawanie grafiki do formularza użytkownika                                                                 | 187        |
|           | Stosowanie przycisku pokrętki na formularzu użytkownika                                                     | 188        |
|           | Stosowanie kontrolki <b>MultiPage</b> do łączenia formularzy                                                | 189        |
|           | Weryfikowanie wpisów w polach                                                                               | 192        |
|           | Nieprawidłowe zamknięcie okna                                                                               | 192        |
|           | Uzyskiwanie nazwy pliku                                                                                     | 193        |
|           | Następne kroki                                                                                              | 194        |
| <b>11</b> | <b>Analiza danych za pomocą funkcji Filtr zaawansowany</b>                                                  | <b>195</b> |
|           | Zastępowanie pętli funkcją Autofiltr                                                                        | 195        |
|           | Wykorzystywanie funkcji Autofiltr                                                                           | 198        |
|           | Zaznaczanie tylko widocznych komórek                                                                        | 202        |
|           | Filtr zaawansowany – łatwiej w VBA niż w programie Excel                                                    | 204        |
|           | Korzystanie z interfejsu użytkownika do budowania filtru zaawansowanego                                     | 204        |
|           | Stosowanie filtru zaawansowanego do uzyskiwania unikatowej listy wartości                                   | 205        |
|           | Uzyskiwanie listy unikatowych wartości za pomocą interfejsu użytkownika                                     | 205        |
|           | Wyodrębnianie unikatowej listy wartości za pomocą kodu VBA                                                  | 207        |
|           | Uzyskiwanie unikatowej kombinacji dwóch lub większej liczby pól                                             | 211        |
|           | Stosowanie filtru zaawansowanego za pomocą zakresów kryteriów                                               | 212        |
|           | Łączenie wielu kryteriów za pomocą sumy logicznej OR                                                        | 214        |
|           | Łączenie dwóch kryteriów za pomocą iloczynu logicznego AND                                                  | 214        |
|           | Inne trochę bardziej złożone zakresy kryteriów                                                              | 214        |
|           | Najbardziej złożone kryteria: Zastępowanie listy wartości warunkiem utworzonym jako wynik formuły           | 216        |
|           | Stosowanie funkcji Filtr zamiast Filtr zaawansowany                                                         | 222        |
|           | Brak rekordów w wyniku przy użyciu opcji Filtruj listę na miejscu                                           | 223        |
|           | Wyświetlanie wszystkich rekordów po uruchomieniu filtrowania listy na miejscu                               | 223        |
|           | Prawdziwy koń pociągowy: <b>xlFilterCopy</b> dla wszystkich rekordów, zamiast opcji Tylko unikatowe rekordy | 224        |
|           | Kopiowanie wszystkich kolumn                                                                                | 224        |
|           | Kopiowanie podzestawu kolumn i zmiana ich kolejności                                                        | 225        |
|           | Program Excel w praktyce: Wyłączanie kilku list rozwijanych funkcji Autofiltr                               | 231        |
|           | Następne kroki                                                                                              | 232        |
| <b>12</b> | <b>Wykorzystywanie VBA do tworzenia tabel przestawnych</b>                                                  | <b>233</b> |
|           | Ewolucja tabel przestawnych w różnych wersjach programu Excel                                               | 233        |

|                                                                                             |            |
|---------------------------------------------------------------------------------------------|------------|
| Tworzenie tabel przestawnych w języku Excel VBA .....                                       | 234        |
| Definiowanie bufora tabeli przestawnej .....                                                | 235        |
| Tworzenie i konfigurowanie tabeli przestawnej .....                                         | 235        |
| Dodawanie pól do obszaru danych .....                                                       | 236        |
| Powody, dla których nie można przenosić lub zmieniać fragmentów raportu przestawnego .....  | 239        |
| Określanie rozmiaru gotowej tabeli przestawnej w celu przekształcenia jej na wartości ..... | 239        |
| Stosowanie zaawansowanych funkcji tabel przestawnych .....                                  | 242        |
| Używanie wielu pól wartości .....                                                           | 242        |
| Grupowanie dat poszczególnych dni według miesięcy, kwartałów lub lat .....                  | 243        |
| Zmiana obliczeń w celu prezentowania wartości procentowych .....                            | 245        |
| Eliminowanie pustych komórek w obszarze wartości .....                                      | 247        |
| Kontrolowanie kolejności sortowania za pomocą opcji autosortowania (AutoSort) .....         | 248        |
| Powielanie raportu dla każdego produktu .....                                               | 248        |
| Filtrowanie zestawu danych .....                                                            | 251        |
| Ręczne filtrowanie dwóch lub kilku elementów w polu tabeli przestawnej .....                | 251        |
| Stosowanie filtrów pojęciowych .....                                                        | 253        |
| Stosowanie filtrów wyszukiwania .....                                                       | 256        |
| Konfigurowanie fragmentatorów w celu filtrowania tabeli przestawnej .....                   | 259        |
| Konfigurowanie osi czasu tabeli przestawnej programu Excel 2016 .....                       | 263        |
| Wykorzystywanie modelu danych w programie Excel 2016 .....                                  | 266        |
| Dodanie obu tabel do Modelu danych .....                                                    | 266        |
| Tworzenie relacji pomiędzy dwoma tabelami .....                                             | 267        |
| Definiowanie bufora tabeli przestawnej i tworzenie tabeli przestawnej .....                 | 267        |
| Dodawanie pól modelu do tabeli przestawnej .....                                            | 268        |
| Dodawanie pól numerycznych do obszaru wartości .....                                        | 268        |
| Podsumowanie .....                                                                          | 269        |
| Stosowanie innych funkcji tabel przestawnych .....                                          | 271        |
| Obliczeniowe pola danych .....                                                              | 271        |
| Elementy obliczeniowe .....                                                                 | 272        |
| Używanie właściwości <b>ShowDetail</b> do filtrowania zestawu rekordów .....                | 272        |
| Zmiana układu na karcie Projektowanie .....                                                 | 272        |
| Ustawienia układu raportu .....                                                             | 273        |
| Wyłączanie sum częściowych dla wielu pól wierszy .....                                      | 273        |
| Następne kroki .....                                                                        | 274        |
| <b>13 Zaawansowane możliwości programu Excel .....</b>                                      | <b>275</b> |
| Operacje na plikach .....                                                                   | 275        |
| Tworzenie listy plików w katalogu .....                                                     | 275        |
| Importowanie i usuwanie pliku CSV .....                                                     | 278        |
| Wczytanie pliku tekstowego do pamięci i analiza pliku .....                                 | 278        |
| Łączenie i rozdzielanie skoroszytów .....                                                   | 279        |
| Rozdzielanie arkuszy w oddzielnych skoroszytach .....                                       | 279        |
| Łączenie skoroszytów .....                                                                  | 280        |
| Filtrowanie i kopiowanie danych do oddzielnych arkuszy .....                                | 281        |
| Kopiowanie danych do oddzielnych arkuszy bez użycia filtra .....                            | 282        |
| Eksportowanie danych do pliku XML .....                                                     | 283        |

|                                                                                     |            |
|-------------------------------------------------------------------------------------|------------|
| Komentarze komórek .....                                                            | 284        |
| Zmiana rozmiaru komentarzy .....                                                    | 284        |
| Umieszczanie wykresu w komentarzu .....                                             | 285        |
| Zaznaczanie komórek .....                                                           | 287        |
| Stosowanie formatowania warunkowego do wyróżniania zaznaczonych komórek .....       | 287        |
| Wyróżnienie zaznaczonych komórek bez stosowania formatowania warunkowego .....      | 289        |
| Zaznaczanie/usuwanie zaznaczenia nieciągłych komórek .....                          | 290        |
| Tworzenie ukrytego pliku dziennika .....                                            | 292        |
| Metody dla profesjonalistów języka VBA .....                                        | 293        |
| Tworzenie w programie Excel modułu klasy stanu .....                                | 293        |
| Pogłębione analizy tabel przestawnych .....                                         | 295        |
| Filtrowanie tabeli przestawnej OLAP według listy elementów .....                    | 296        |
| Tworzenie niestandardowej kolejności sortowania .....                               | 298        |
| Tworzenie wskaźnika postępu .....                                                   | 299        |
| Stosowanie chronionych pól haseł .....                                              | 300        |
| Zmiana wielkości liter .....                                                        | 302        |
| Zaznaczanie za pomocą SpecialCells .....                                            | 304        |
| Resetowanie formatu tabeli .....                                                    | 304        |
| Sprytne aplikacje .....                                                             | 305        |
| Uzyskiwanie danych historycznych notowań giełdowych .....                           | 305        |
| Używanie możliwości rozbudowy VBA w celu dodawania kodu do nowych skoroszytów ..... | 306        |
| Następne kroki .....                                                                | 308        |
| <b>14 Przykłady funkcji definiowanych przez użytkownika .....</b>                   | <b>309</b> |
| Tworzenie funkcji definiowanych przez użytkownika .....                             | 309        |
| Udostępnianie funkcji UDF .....                                                     | 311        |
| Użycie niestandardowych funkcji programu Excel .....                                | 312        |
| Ustawianie nazwy bieżącego skoroszytu w komórce .....                               | 312        |
| Ustawianie w komórce nazwy bieżącego skoroszytu i ścieżki pliku .....               | 312        |
| Sprawdzenie, czy skoroszyt jest otwarty .....                                       | 313        |
| Sprawdzenie, czy istnieje arkusz w otwartym skoroszytcie .....                      | 313        |
| Określenie liczby skoroszytów w katalogu .....                                      | 314        |
| Pobieranie ID użytkownika .....                                                     | 315        |
| Pobieranie informacji o dacie i godzinie ostatniego zapisu .....                    | 317        |
| Pobieranie informacji o niezmiennącej się dacie i godzinie .....                    | 317        |
| Sprawdzanie poprawności adresu e-mail .....                                         | 318        |
| Sumowanie komórek w oparciu o kolor ich wypełnienia .....                           | 319        |
| Zliczanie wartości unikatowych .....                                                | 320        |
| Usuwanie duplikatów z zakresu .....                                                 | 321        |
| Wyszukiwanie w zakresie pierwszej komórki o niezerowej długości .....               | 323        |
| Zastępowanie wielu znaków .....                                                     | 323        |
| Uzyskiwanie liczb z mieszanego tekstu .....                                         | 325        |
| Przekształcanie numeru tygodnia na datę .....                                       | 325        |
| Wyodrębnianie pojedynczego elementu z ograniczonego ciągu .....                     | 326        |
| Sortowanie i łączenie .....                                                         | 326        |
| Sortowanie liczb i znaków alfanumerycznych .....                                    | 328        |

|                                                                                              |            |
|----------------------------------------------------------------------------------------------|------------|
| Wyszukiwanie ciągu wewnątrz tekstu . . . . .                                                 | 329        |
| Odwracanie zawartości komórki . . . . .                                                      | 330        |
| Zwracanie adresów duplikatów wartości maksymalnych . . . . .                                 | 331        |
| Zwracanie adresu hiperłącza . . . . .                                                        | 332        |
| Zwracanie litery kolumny na podstawie adresu komórki . . . . .                               | 332        |
| Stosowanie statycznej funkcji losowej . . . . .                                              | 333        |
| Używanie <b>Select Case</b> na arkuszu . . . . .                                             | 333        |
| Następne kroki . . . . .                                                                     | 334        |
| <b>15 Tworzenie wykresów . . . . .</b>                                                       | <b>335</b> |
| Porównanie dobrego i niedobrego kodu VBA używanego do tworzenia wykresów . . . . .           | 335        |
| Planowanie przeróbki wielu wykresów . . . . .                                                | 336        |
| Stosowanie metody <b>AddChart2</b> do tworzenia wykresu . . . . .                            | 337        |
| Style wykresu . . . . .                                                                      | 338        |
| Formatowanie wykresu . . . . .                                                               | 341        |
| Odwoływanie się do określonego wykresu . . . . .                                             | 341        |
| Specyfikowanie tytułu wykresu . . . . .                                                      | 342        |
| Stosowanie koloru wykresu . . . . .                                                          | 343        |
| Filtrowanie wykresu . . . . .                                                                | 344        |
| Używanie metody <b>SetElement</b> do emulowania zmian dostępnych w menu ikony Plus . . . . . | 345        |
| Stosowanie metody <b>Format</b> do zarządzania formatowaniem . . . . .                       | 350        |
| Zmiana wypełnienia obiektu . . . . .                                                         | 351        |
| Formatowanie ustawień linii . . . . .                                                        | 353        |
| Tworzenie wykresów kombi . . . . .                                                           | 354        |
| Eksportowanie wykresu jako grafiki PowerPoint . . . . .                                      | 357        |
| Rozważanie kompatybilności wstecznej . . . . .                                               | 358        |
| Następne kroki . . . . .                                                                     | 358        |
| <b>16 Wizualizacje danych i formatowanie warunkowe . . . . .</b>                             | <b>359</b> |
| Metody i ich właściwości w VBA do wizualizacji danych . . . . .                              | 361        |
| Dodawanie pasków danych do zakresów . . . . .                                                | 362        |
| Dodawanie do zakresu skali kolorów . . . . .                                                 | 366        |
| Dodawanie do zakresu zestawów ikon . . . . .                                                 | 367        |
| Specyfikowanie zestawu ikon . . . . .                                                        | 368        |
| Specyfikowanie zakresów dla każdej ikony . . . . .                                           | 370        |
| Stosowanie sztuczek przy wizualizacji danych . . . . .                                       | 370        |
| Tworzenie zestawu ikon dla podzbioru zakresu . . . . .                                       | 371        |
| Używanie w zakresie dwóch kolorów dla pasków danych . . . . .                                | 373        |
| Stosowanie innych metod formatowania warunkowego . . . . .                                   | 375        |
| Formatowanie komórek, których wartości są powyżej lub poniżej średniej . . . . .             | 375        |
| Formatowanie komórek za pomocą reguły Pierwsze 10 lub Ostatnie 5 . . . . .                   | 376        |
| Formatowanie unikatowych lub duplikowanych komórek . . . . .                                 | 376        |
| Formatowanie komórek w oparciu o ich wartość . . . . .                                       | 378        |
| Formatowanie komórek zawierających tekst . . . . .                                           | 378        |
| Formatowanie komórek, które zawierają daty . . . . .                                         | 379        |
| Formatowanie komórek, które są puste lub zawierają błędy . . . . .                           | 379        |
| Używanie formuły do określenia, które komórki mają być formatowane . . . . .                 | 380        |

|                                                                                           |            |
|-------------------------------------------------------------------------------------------|------------|
| Stosowanie nowej właściwości <b>NumberFormat</b> .....                                    | 381        |
| Następne kroki .....                                                                      | 382        |
| <b>17 Tworzenie pulpitów nawigacyjnych za pomocą wykresów przebiegu w czasie .....</b>    | <b>383</b> |
| Tworzenie wykresów przebiegu w czasie .....                                               | 384        |
| Skalowanie wykresów przebiegu w czasie .....                                              | 386        |
| Formatowanie wykresów przebiegu w czasie .....                                            | 389        |
| Stosowanie kolorów motywu .....                                                           | 390        |
| Stosowanie kolorów RGB .....                                                              | 393        |
| Formatowanie elementów wykresów przebiegu w czasie .....                                  | 394        |
| Formatowanie wykresów Zysk/strata .....                                                   | 397        |
| Tworzenie pulpitów nawigacyjnych .....                                                    | 398        |
| Uwagi dotyczące wykresów przebiegu w czasie .....                                         | 399        |
| Tworzenie setek poszczególnych wykresów przebiegu w czasie na pulpicie nawigacyjnym ..... | 399        |
| Następne kroki .....                                                                      | 404        |
| <b>18 Odczytywanie i zapisywanie na stronach sieci Web .....</b>                          | <b>405</b> |
| Uzyskiwanie danych z sieci Web .....                                                      | 405        |
| Tworzenie wielu zapytań za pomocą VBA .....                                               | 407        |
| Wyszukiwanie wyników w pobranych danych .....                                             | 408        |
| Podsumowanie .....                                                                        | 409        |
| Przykłady pobierania danych z witryn sieci Web za pomocą kwerend .....                    | 411        |
| Używanie metody <b>Application.OnTime</b> do okresowej analizy danych .....               | 411        |
| Używanie trybu Ready w zaplanowanych procedurach .....                                    | 412        |
| Specyfikowanie czasu systemu Windows dla aktualizacji .....                               | 412        |
| Anulowanie poprzednio zaplanowanego makro .....                                           | 413        |
| Zamknięcie programu Excel anuluje wszystkie oczekujące zaplanowane makra .....            | 413        |
| Planowanie uruchamiania makra x minut w przyszłości .....                                 | 414        |
| Planowanie przypomnienia słownego .....                                                   | 414        |
| Zaplanowanie uruchamiania makra co dwie minuty .....                                      | 415        |
| Publikowanie danych na stronie sieci Web .....                                            | 416        |
| Stosowanie VBA do tworzenia niestandardowych stron sieci Web .....                        | 417        |
| Stosowanie programu Excel jako systemu zarządzania zawartością .....                      | 418        |
| Premia: FTP w programie Excel .....                                                       | 421        |
| Następne kroki .....                                                                      | 421        |
| <b>19 Przetwarzanie plików tekstowych .....</b>                                           | <b>423</b> |
| Importowanie z plików tekstowych .....                                                    | 423        |
| Importowanie plików tekstowych, które mają mniej niż 1 048 576 wierszy .....              | 423        |
| Obsługa plików tekstowych zawierających więcej niż 1 048 576 wierszy .....                | 430        |
| Zapisywanie plików tekstowych .....                                                       | 435        |
| Następne kroki .....                                                                      | 435        |
| <b>20 Automatyzowanie programu Word .....</b>                                             | <b>437</b> |
| Stosowanie wiązania wczesnego do odwoływania się do obiektów programu Word .....          | 438        |
| Stosowanie wiązania późnego do odwołań do obiektów programu Word .....                    | 440        |
| Stosowanie słowa kluczowego <b>New</b> w odwołaniach do aplikacji Word .....              | 441        |

|                                                                                                                                      |            |
|--------------------------------------------------------------------------------------------------------------------------------------|------------|
| Stosowanie funkcji <b>CreateObject</b> do tworzenia nowej instancji obiektu .....                                                    | 441        |
| Stosowanie funkcji <b>GetObject</b> do odwoływania się do istniejącej instancji programu Word .....                                  | 442        |
| Używanie wartości stałych .....                                                                                                      | 443        |
| Używanie okna czujek do uzyskiwania rzeczywistych wartości stałych .....                                                             | 443        |
| Stosowanie wyszukiwarki obiektów do uzyskania rzeczywistych wartości stałych .....                                                   | 444        |
| Działanie obiektów programu Word .....                                                                                               | 445        |
| Obiekt <b>Document</b> .....                                                                                                         | 446        |
| Obiekt <b>Selection</b> .....                                                                                                        | 448        |
| Obiekt <b>Range</b> .....                                                                                                            | 449        |
| Zakładki .....                                                                                                                       | 452        |
| Kontrolowanie pól formularzy w programie Word .....                                                                                  | 454        |
| Następne kroki .....                                                                                                                 | 456        |
| <b>21 Stosowanie programu Access jako wewnętrznej bazy danych w celu usprawnienia dostępu do danych dla wielu użytkowników .....</b> | <b>457</b> |
| Porównanie ADO i DAO .....                                                                                                           | 458        |
| Narzędzia obiektów ADO .....                                                                                                         | 460        |
| Dodawanie rekordów do bazy danych .....                                                                                              | 462        |
| Pobieranie rekordów z bazy danych .....                                                                                              | 464        |
| Aktualizowanie istniejącego rekordu .....                                                                                            | 466        |
| Usuwanie rekordów poprzez obiekt ADO .....                                                                                           | 468        |
| Sumowanie rekordów za pośrednictwem obiektów ADO .....                                                                               | 468        |
| Inne narzędzia za pośrednictwem ADO .....                                                                                            | 469        |
| Sprawdzanie, czy tabela istnieje .....                                                                                               | 470        |
| Sprawdzenie, czy istnieje pole .....                                                                                                 | 470        |
| Dodawanie tabeli na bieżąco .....                                                                                                    | 471        |
| Dodawanie pola na bieżąco .....                                                                                                      | 472        |
| Przykłady dla SQL Server .....                                                                                                       | 472        |
| Następne kroki .....                                                                                                                 | 474        |
| <b>22 Zaawansowane techniki stosowania obiektów UserForm .....</b>                                                                   | <b>475</b> |
| Stosowanie paska narzędzi UserForm w projektowaniu kontrolki na formularzach .....                                                   | 475        |
| Dodatkowe kontrolki użytkownika .....                                                                                                | 476        |
| Kontrolki <b>Checkbox</b> .....                                                                                                      | 476        |
| Kontrolki i kolekcje .....                                                                                                           | 484        |
| Niemodalne formularze użytkownika .....                                                                                              | 486        |
| Stosowanie hiperłączy w formularzach użytkownika .....                                                                               | 486        |
| Dodawanie kontrolki w czasie działania .....                                                                                         | 487        |
| Zmiana na bieżąco rozmiaru formularza użytkownika .....                                                                              | 489        |
| Dodawanie kontrolki na bieżąco .....                                                                                                 | 489        |
| Zmiana rozmiaru na bieżąco .....                                                                                                     | 490        |
| Dodawanie innych kontrolki .....                                                                                                     | 490        |
| Dodawanie obrazu na bieżąco .....                                                                                                    | 491        |
| Podsumowanie .....                                                                                                                   | 491        |
| Dodawanie pomocy do formularza użytkownika .....                                                                                     | 494        |
| Wyświetlanie klawiszy akceleratorów .....                                                                                            | 494        |
| Dodawanie kontrolki porady tekstowej .....                                                                                           | 494        |

|                                                                                         |            |
|-----------------------------------------------------------------------------------------|------------|
| Określanie kolejności kart                                                              | 495        |
| Kolorowanie kontrolki aktywnej                                                          | 495        |
| Tworzenie przezroczystych formularzy                                                    | 498        |
| Następne kroki                                                                          | 499        |
| <b>23 Interfejs programowania aplikacji (API)</b>                                       | <b>501</b> |
| Deklaracje interfejsu API                                                               | 502        |
| Używanie deklaracji API                                                                 | 503        |
| Tworzenie deklaracji API zgodnych z systemami 32- i 64-bitowymi                         | 503        |
| Przykłady funkcji API                                                                   | 505        |
| Uzyskiwanie nazwy komputera                                                             | 505        |
| Sprawdzenie, czy plik programu Excel jest otwarty w sieci                               | 506        |
| Uzyskiwanie informacji dotyczących rozdzielczości ekranu                                | 506        |
| Dostosowywanie okna dialogowego Windows – informacje                                    | 507        |
| Blokowanie przycisku X do zamykania formularza użytkownika                              | 508        |
| Tworzenie czasomierza                                                                   | 509        |
| Odtwarzanie dźwięków                                                                    | 510        |
| Następne kroki                                                                          | 510        |
| <b>24 Obsługa błędów</b>                                                                | <b>511</b> |
| Co dzieje się, kiedy pojawia się błąd?                                                  | 511        |
| Mylące błędy debugowania kodu formularza użytkownika                                    | 513        |
| Podstawowa obsługa błędów za pomocą składni <b>On Error GoTo</b>                        | 514        |
| Ogólne programy obsługi błędów                                                          | 516        |
| Obsługa błędów poprzez wybranie opcji ignorowania błędów                                | 516        |
| Blokowanie ostrzeżeń programu Excel                                                     | 518        |
| Celowe wywoływanie błędu                                                                | 518        |
| Szkolenia klientów                                                                      | 519        |
| Błędy podczas projektowania w porównaniu z błędami występującymi kilka miesięcy później | 520        |
| Runtime Error 9: Indeks poza zakresem                                                   | 520        |
| Runtime Error 1004: Nie zadziałała metoda Range dla obiektu globalnego                  | 521        |
| Dolegliwości kodu chronionego                                                           | 522        |
| Więcej problemów dotyczących hasła                                                      | 523        |
| Błędy powodowane przez zmiany wersji                                                    | 524        |
| Następne kroki                                                                          | 524        |
| <b>25 Dostosowywanie wstążki w celu uruchamiania makr</b>                               | <b>525</b> |
| Gdzie dodawać kod: Folder i plik customui                                               | 526        |
| Tworzenie kart i grup                                                                   | 528        |
| Dodawanie kontrolki do wstążki                                                          | 528        |
| Uzyskiwanie dostępu do struktury plików                                                 | 535        |
| Działanie pliku RELS                                                                    | 535        |
| Zmiana nazwy pliku Excel i otwieranie skoroszytu                                        | 536        |
| Używanie obrazów na przyciskach                                                         | 537        |
| Stosowanie na wstążce ikon pakietu Microsoft Office                                     | 537        |
| Dodawanie obrazów ikon do wstążki                                                       | 538        |
| Rozwiązywanie problemów dotyczących komunikatów o błędach                               | 540        |

|                                                                                                                   |            |
|-------------------------------------------------------------------------------------------------------------------|------------|
| Atrybut "Attribute Name" dla elementu "customui Ribbon" nie jest zdefiniowany w schemacie lub definicji DTD ..... | 540        |
| Niedozwolony znak w nazwie kwalifikowanej .....                                                                   | 540        |
| Nazwa znacznika "customui" nie jest akceptowana w odniesieniu do modelu zawartości elementu nadrzędnego .....     | 541        |
| Problemy z pewnymi zawartościami .....                                                                            | 541        |
| Nieprawidłowa liczba argumentów lub nieprawidłowe przypisanie właściwości .....                                   | 542        |
| Nieprawidłowy format lub rozszerzenie pliku .....                                                                 | 543        |
| Nic się nie stało .....                                                                                           | 543        |
| Inne sposoby uruchomienia makra .....                                                                             | 543        |
| Stosowanie skrótów klawiszowych do uruchamiania makra .....                                                       | 543        |
| Dołączanie makra do przycisku polecenia .....                                                                     | 544        |
| Dołączanie makra do kształtu .....                                                                                | 545        |
| Dołączanie makra do kontrolki ActiveX .....                                                                       | 546        |
| Uruchamianie makra za pomocą hiperłącza .....                                                                     | 547        |
| Następne kroki .....                                                                                              | 548        |
| <b>26 Tworzenie dodatków .....</b>                                                                                | <b>549</b> |
| Cechy standardowych dodatków .....                                                                                | 549        |
| Przekształcanie skoroszytu programu Excel na dodatek .....                                                        | 550        |
| Używanie funkcji Zapisywanie jako do przekształcenia pliku na dodatek .....                                       | 551        |
| Stosowanie edytora VB do konwersji pliku na dodatek .....                                                         | 552        |
| Instalowanie dodatku w systemie klienckim .....                                                                   | 553        |
| Zamykanie dodatków .....                                                                                          | 555        |
| Usuwanie dodatków .....                                                                                           | 555        |
| Stosowanie ukrytego arkusza jako alternatywa dla dodatku .....                                                    | 555        |
| Następne kroki .....                                                                                              | 557        |
| <b>27 Omówienie sposobów tworzenia dodatków pakietu Office .....</b>                                              | <b>559</b> |
| Tworzenie pierwszego dodatku pakietu Office – Hello World .....                                                   | 560        |
| Dodawanie mechanizmów interakcji do dodatku pakietu Office .....                                                  | 564        |
| Wstęp do języka HTML .....                                                                                        | 567        |
| Stosowanie znaczników .....                                                                                       | 567        |
| Dodawanie przycisków .....                                                                                        | 567        |
| Wykorzystanie plików CSS .....                                                                                    | 568        |
| Stosowanie XML do definiowania dodatku pakietu Office .....                                                       | 568        |
| Wykorzystanie kodu JavaScript w celu dodania do dodatku pakietu Office mechanizmów interakcji .....               | 569        |
| Struktura funkcji .....                                                                                           | 570        |
| Zmienne .....                                                                                                     | 571        |
| Ciągi .....                                                                                                       | 572        |
| Tablice .....                                                                                                     | 572        |
| Pętle <b>for</b> w kodzie JavaScript .....                                                                        | 573        |
| Działanie instrukcji <b>if</b> w kodzie JavaScript .....                                                          | 574        |
| Działanie instrukcji <b>Select..Case</b> w kodzie JavaScript .....                                                | 574        |
| Działanie instrukcji <b>For each..next</b> w kodzie JavaScript .....                                              | 576        |
| Operatory matematyczne, logiczne i używane do przypisywania .....                                                 | 576        |
| Funkcje matematyczne w kodzie JavaScript .....                                                                    | 578        |

|                                                                                |            |
|--------------------------------------------------------------------------------|------------|
| Zapisywanie w okienku zawartości lub okienku zadań .....                       | 579        |
| Modyfikacje kodu JavaScript, by działał w dodatku pakietu Office .....         | 579        |
| Napa Office 365 Development Tools .....                                        | 581        |
| Następne kroki .....                                                           | 581        |
| <b>28 Nowości i zmiany w programie Excel 2016 .....</b>                        | <b>583</b> |
| Jeśli coś zmieniło się w interfejsie użytkownika, zmieniło się też w VBA ..... | 583        |
| Wstążka .....                                                                  | 583        |
| Interfejs SDI (Single Document Interface) .....                                | 584        |
| Narzędzie Szybka analiza .....                                                 | 585        |
| Wykresy .....                                                                  | 585        |
| Tabele przestawne .....                                                        | 586        |
| Fragmentatory .....                                                            | 586        |
| Grafiki SmartArt .....                                                         | 586        |
| Poznanwanie nowych obiektów i metod .....                                      | 587        |
| Tryb zgodności .....                                                           | 587        |
| Stosowanie właściwości <b>Version</b> .....                                    | 588        |
| Stosowanie właściwości <b>Excel8CompatibilityMode</b> .....                    | 588        |
| Następne kroki .....                                                           | 589        |
| <b>Indeks .....</b>                                                            | <b>591</b> |

# 0 autorach

**Bill Jelen**, posiadacz tytułu Excel MVP i gospodarz witryny MrExcel.com, posługuje się arkuszami kalkulacyjnymi od roku 1985, zaś witrynę MrExcel.com utworzył w roku 1998. Jest regularnym gościem wideobloga *Call for Help with Leo Laporte* i wyprodukował ponad 1900 odcinków swojego codziennego podcastu wideo, *Learn Excel from MrExcel*. Jest autorem 44 książek na temat programu Microsoft Excel i comiesięcznego felietonu dla *Strategic Finance*. Przed utworzeniem witryny MrExcel.com Bill Jelen spędził 12 lat na froncie, pracując w działach analiz finansowych, marketingu, księgowości i zarządzania publicznej spółki o kapitale przekraczającym 500 milionów dolarów. Mieszka w Merritt Island na Florydzie ze swą żoną Mary Ellen.

**Tracy Syrstad** jest programistą i projektantem rozwiązań dla Microsoft Excel oraz autorką ośmiu książek o tej tematyce. Zajmuje się wsparciem w zakresie Microsoft Office od roku 1997, gdy odkryła istnienie publicznych forum online, w których każdy mógł zadać pytanie lub udzielić odpowiedzi. Tracy odkryła, że uczenie innych sprawia jej przyjemność i gdy zaczęła pracować jako projektant, połączyła radość uczenia z codzienną pracą. Mieszka w odludnym rejonie Południowej Dakoty w towarzystwie męża, psa, dwóch kotów, jednego konia (choć ma nadzieję, że niedługo pojawi się drugi) oraz rozlicznych dzikich lisów, wiewiórek i królików.

## Dedykacje

*Dla Roberta K. Jelenia*

– *Bill Jelen*

*Dla Marlee Jo Jacobson*

– *Tracy Syrstad*

# Podziękowania

Przede wszystkim dziękuję Tracy Syrstad za to, że jest wspaniałym współautorem. Bob Umlas jest jednym z najinteligentniejszych specjalistów od Excela, jakich znam i jest wspaniałym redaktorem technicznym. Joan Murray z wydawnictwa Pearson okazała się świetnym redaktorem merytorycznym.

Zanim mogłem zacząć pracę nad tą książką, nauczyłem się bardzo wiele o programowaniu VBA od wspaniałej społeczności zgromadzonej wokół witryny MrExcel.com. Szczególnie wyróżniają się VoG, Richard Schollar oraz Jon von der Heyden, którzy wnieśli zasadniczy wkład w koncepcje, które doprowadziły do powstania tej książki. Podziękować chcę też Pam Gensel za pierwszą lekcję tworzenia makr w Excelu 1. Mala Singh nauczył mnie, jak tworzyć wykresy w VBA, zaś Oliver Holloway pokazał mi, jak szybko i sprawnie uzyskiwać dostęp do SQL Server. Scott Ruble i Robin Wakefield z firmy Microsoft pomogli w tworzeniu rozdziału o wykresach.

Na koniec chcę podziękować mojej rodzinie za wsparcie i pomoc podczas pracy nad książką.

—*Bill*

Juan Pablo Gonzalez Ruiz i Zack Barresse są wspaniałymi programistami i szczerze doceniam czas i cierpliwość, które poświęcili na nauczanie mnie nowych sposobów pisania lepszych programów. Chris „Smitty” Smith był szczególnie pomocny przy doskonaleniu moich umiejętności programistycznych i biznesowych.

Dziękuję wszystkim moderatorom forum MrExcel, którzy dbają o porządek pomimo wielkich wysiłków ze strony spamerów. Programowanie to ciągła nauka i dziękuję tym wszystkim, którzy przychodzą z problemami i pozwalają odkrywać nowe zagadnienia i rozwiązania, z których inaczej w ogóle nie zdawałabym sobie sprawy.

Na koniec chcę podziękować Billowi. Jego witryna, MrExcel.com, jest tym miejscem, do którego tysiące przychodzi po pomoc. Jest to również miejsce, w którym ja i inni podobni do mnie znajdują okazję, aby uczyć się i pomagać uczyć się innym.

—*Tracy*

# Czekamy na Twoją opinię!

Jako Czytelnik tej książki, jesteś naszym najważniejszym krytykiem i komentatorem. Cienimy Twoją opinię i chcielibyśmy dowiedzieć się, co robimy dobrze, co moglibyśmy zrobić lepiej oraz z jakich innych obszarów publikacjami byłbyś najbardziej zainteresowany. Oczywiście chętnie przyjmiemy dowolne inne komentarze wysłane pod naszym adresem.

Twoje opinie są zawsze mile widziane. Możesz do nas napisać lub wysłać nam wiadomość e-mail i poinformować nas, co Ci się podobało lub nie podobało w tej książce, a także co możemy zrobić, aby nasze książki były jeszcze lepsze.

Prosimy jednak zwrócić uwagę, że nie jesteśmy w stanie nikomu pomóc przy rozwiązywaniu problemów technicznych związanych z tematem tej książki.

Jeśli zdecydujesz się do nas napisać, pamiętaj o zawarciu w treści wiadomości tytułu tej książki, nazwisk jej autorów, a także swojego imienia i adresu e-mail. Z uwagą przejrzymy Twoje komentarze i podzielimy się nimi z autorami i recenzentami, którzy pracowali nad niniejszym podręcznikiem.

**E-mail:** [feedback@quepublishing.com](mailto:feedback@quepublishing.com)

**Pocztą:** Que Publishing

ATTN: Reader Feedback

800 East 96th Street

Indianapolis, IN 46240 USA

# Wprowadzenie

Kiedy okazało się, że czasy realizowania żądań przez działy IT ciągle się wydłużają, użytkownicy programu Excel odkryli, że przy użyciu języka makr *Visual Basic for Applications* (VBA) sami mogą tworzyć raporty potrzebne do prowadzenia przedsiębiorstwa. VBA umożliwia nam uzyskać niewiarygodną efektywność w codziennym użytkowaniu programu Excel. VBA pomaga nam znaleźć sposób na zaimportowanie danych i wygenerowanie raportów w programie Excel, dzięki czemu nie musimy czekać, by w tym zadaniu pomógł nam dział IT.

## Co zawiera niniejsza książka?

Zakup tej książki to dobra decyzja – pozwoli skrócić i ułatwić proces opanowania umiejętności pozwalających samemu napisać makra VBA, a w konsekwencji zlikwidować obciążenia związane z ręcznym generowaniem raportów.

### Skrócenie procesu przyswajania wiedzy

We Wprowadzeniu przedstawiono analizę przypadku, która ilustruje możliwości makr. W rozdziale 1 „Zwiększanie możliwości programu Excel za pomocą języka VBA” znajdziemy omówienie narzędzi i potwierdzenie znanego faktu: rejestrator makr nie działa stabilnie. Rozdział 2 „Skoro nazywa się BASIC, dlaczego nie wygląda znajomo?” ułatwia zrozumienie szalonej składni języka VBA, a rozdział 3 „Odwoływanie się do zakresów” wyjaśnia, jak sprawnie pracować z zakresami i komórkami.

W rozdziale 4 „Pętle i sterowanie przepływem” omówiono możliwości mechanizmu pętli w języku VBA. Analiza przypadku w tym rozdziale prezentuje proces tworzenia programu generowania raportu dla działu, a następnie „opakowanie” go procedurą, która za pomocą pętli generuje 46 raportów.

|                                                                 |   |
|-----------------------------------------------------------------|---|
| Co zawiera niniejsza książka? .....                             | 1 |
| Funkcje VBA a wersje programu Excel<br>w systemie Windows ..... | 5 |
| Elementy specjalne i konwencje typograficzne ...                | 5 |
| Pliki kodu .....                                                | 6 |
| Następne kroki .....                                            | 6 |

W rozdziale 5 „Formuły w stylu R1C1” rzecz jasna opisano działanie tych formuł, a w rozdziale 6 „Tworzenie nazw i operacje na nazwach w VBA” omówiono nazwy. Rozdział 7 „Programowanie zdarzeń” przedstawia niektóre sztuczki używane w programowaniu zdarzeń. Rozdział 8 „Tablice” i rozdział 9 „Tworzenie klas, rekordów i kolekcji” to omówienie tablic, klas i kolekcji, a w rozdziale 10 „Obiekty UserForm – wprowadzenie” opisano niestandardowe okna dialogowe, których można używać do zbierania informacji od użytkowników za pomocą programu Excel.

### **Możliwości programu Excel VBA**

Rozdział 11 „Analiza danych za pomocą funkcji Filtr zaawansowany” i rozdział 12 „Wykorzystywanie VBA do tworzenia tabel przestawnych” to dokładne przedstawienie narzędzi Filtr, Filtr zaawansowany i tabele przestawne. Narzędzia automatyzacji raportów rzadko wykorzystują te koncepcje. W rozdziale 13 „Zaawansowane możliwości programu Excel” i w rozdziale 14 „Przykłady funkcji zdefiniowanych przez użytkownika” omówiono dziesiątki przykładów kodu opracowanego w celu zaprezentowania możliwości programu Excel VBA i funkcji niestandardowych.

W rozdziałach od 15 „Tworzenie wykresów” do 20 „Automatyzowanie programu Word” zapoznajemy się z procesami tworzenia wykresów, wizualizacji danych, generowania zapytań do stron sieci Web i automatyzowania działań w programie Word.

### **Materiały naukowe potrzebne do tworzenia aplikacji**

W rozdziale 21 „Stosowanie programu Access jako wewnętrznej bazy danych w celu usprawnienia dostępu do danych dla wielu użytkowników” omówiono obsługę odczytywania i zapisywania baz danych programu Access i SQL Server. Metody wykorzystywania baz danych Access umożliwiają tworzenie aplikacji z funkcjami obsługi wielu użytkowników, które są dostępne w programie Access, przy jednoczesnym zachowaniu przyjaznego interfejsu użytkownika programu Excel.

W rozdziale 22 „Zaawansowane techniki stosowania obiektów UserForm” rozwinęta została tematyka stosowania formularzy użytkowników. Rozdział 23 „Interfejs programowania aplikacji (API)” zapoznaje nas z niektórymi sposobami realizacji zadań za pomocą interfejsu Windows API. W rozdziałach od 24 „Obsługa błędów” do 26 „Tworzenie dodatków” omówiono obsługę błędów oraz działanie niestandardowych menu i dodatków. Rozdział 27 „Omówienie sposobów tworzenia dodatków pakietu Office” to krótkie wprowadzenie w proces konstruowania własnych aplikacji JavaScript w programie Excel, a rozdział 28 „Nowości i zmiany w programie Excel 2016” to podsumowanie zmian wprowadzonych w wersji Excel 2016.

### **Czy książka ta uczy programu Excel?**

Firma Microsoft ocenia, że typowi użytkownicy pakietu Office korzystają z 10% funkcji tego oprogramowania. Mamy świadomość, że Czytelnicy tej książki znacznie wykraczają ponad tę średnią, a witryna MrExcel.com ma bardzo inteligentnych użytkowników.

Pomimo tego, ankieta przeprowadzona wśród 8000 czytelników witryny MrExcel.com pokazała, że tylko 42% użytkowników (a jest to grupa ponadprzeciętnych użytkowników!) korzysta z przynajmniej jednej spośród 10 zaawansowanych funkcji programu Excel.

Regularnie prowadzę seminaria Power Excel dla księgowych. Są to podstawowi użytkownicy programu Excel, którzy poświęcają 30 do 40 godzin tygodniowo na pracę z tym programem. Na każdym seminarium pojawiają się dwie kwestie. Pierwsza kwestia to mocne zaskoczenie co najmniej połowy uczestników, jak szybko można zrealizować zadanie za pomocą poszczególnych funkcji, na przykład automatycznego podsumowania czy tabel przestawnych. Druga kwestia to fakt, że zawsze ktoś z uczestników mnie „przebija”. Przykładowo, ktoś zadaje pytanie, odpowiadam na nie, a inna osoba w drugim rzędzie podnosi rękę i daje lepszą odpowiedź.

Wniosek? Sporo wiemy na temat programu Excel. Jednakże oceniam, że we wszystkich rozdziałach, może 58% osób nie stosowało wcześniej tabel przestawnych i może nawet jeszcze mniej używało funkcji filtru „10 pierwszych” w tabeli przestawnej. Mając to na uwadze zdecydowałem, że zanim pokażę, jak zautomatyzować cokolwiek w VBA, krótko omawiam wykonanie tego samego zadania w interfejsie programu Excel. Niniejsza książka nie uczy nas, jak tworzyć tabelę przestawną, ale jedynie zwraca uwagę na tematykę, z którą warto zapoznać się dokładniej.

### Studium przypadku: Miesięczne raporty księgowe

To jest prawdziwa historia. Valerie jest analitykiem biznesowym w dziale księgowości średniej wielkości korporacji. W jej firmie niedawno zainstalowano system ERP (enterprise resource planning), który przekroczył budżet o 16 milionów dolarów. Pod koniec projektu okazało się, że zabrakło funduszy w budżecie IT na generowanie miesięcznych raportów wykorzystywanych w firmie do podsumowywania działań każdego działu.

Jednak Valerie była już bliska podjęcia decyzji o tworzeniu raportów własnymi siłami. Wiedziała, że w tym celu konieczne będzie wyeksportowanie danych księgi głównej z systemu ERP do pliku tekstowego o wartościach rozdzielanych przecinkami. Za pomocą programu Excel, Valerie potrafiła zaimportować dane księgi głównej z systemu ERP do programu Excel.

Tworzenie raportu nie było proste. Podobnie jak w wielu innych firmach, w danych pojawiały się wyjątki. Valerie wiedziała, że niektóre konta w jednym ze źródeł kosztów powinny być przeklasyfikowane jako wydatki oraz że inne konta trzeba całkowicie usunąć z raportu. Pracując uważnie w programie Excel, Valerie wykonała te dostosowania. Utworzyła tabelę przestawną, by wygenerować pierwszą część podsumowania raportu. Wycięła wyniki tabeli przestawnej i wkleiła je do pustego arkusza. Następnie utworzyła nowy raport tabeli przestawnej dla drugiej części podsumowania. Po około trzech godzinach miała zaimportowane dane, utworzone i poukładane do podsumowania pięć tabel przestawnych oraz starannie sformatowany raport.

## Zostać bohaterem

Valerie zaniósła raport menedżerowi, który właśnie usłyszał od działu IT, że generowanie tego zawiłego raportu mogą zabrać dwa, trzy miesiące. Po utworzeniu raportu w programie Excel, Valerie natychmiast stała się bohaterem dnia. W trzy godziny zrobiła coś niemożliwego do wykonania. Valerie była w siódmym niebie, kiedy usłyszała zasłużone „zuch dziewczyna”.

## Kolejne zachwyty

Następnego dnia, menedżer Valerie uczestniczył w comiesięcznym spotkaniu działów. Kiedy inni menedżerowie działów zaczęli narzekać, że nie mogą uzyskać raportów z systemu ERP, menedżer Valerie położył swój raport na stół. Pozostali byli bardzo zdumieni, jak można było wygenerować ten raport? Każdy był zadowolony, że udało się komuś „złamać kod”. Szef firmy poprosił menedżera Valerie, czy nie mógłby przygotowywać raportu dla każdego działu.

## Radość przemienia się w strach

Łatwo domyślić się, co się stało. W tej konkretnej firmie było 46 działów. Oznacza to 46 jednostronicowych podsumowań, które trzeba generować co miesiąc. Każdy raport wymaga importowania danych z systemu ERP, usunięcia pewnych kont, utworzenia pięciu tabel przestawnych i sformatowania raportu w kolorze. Utworzenie pierwszego raportu zajęło Valerie trzy godziny, a po nabyciu wprawy, można oszacować, że wygenerowanie 46 raportów zajmie jej około 40. Nawet jeśli Valerie jeszcze bardziej skróci ten czas, i tak jest to horror. Valerie miała wcześniej inne zadania do wykonania, zanim stała się odpowiedzialna za „spędzenie” 40 godzin na generowaniu tych raportów.

## Ratunkiem jest VBA

Valerie znalazła moją firmę, MrExcel Consulting i wyjaśniła na czym polega problem. W ciągu tygodnia napisałem kilka makr w VBA, które realizowały wszystkie prozaiczne zadania. Na przykład, makra importowały dane, usuwały zbędne konta, tworzyły 5 tabel przestawnych i formatowały raporty. Inaczej mówiąc, 40-to godzinny ręczny proces został zredukowany do kliknięcia dwóch przycisków i skrócony do około 4 minut.

Niewątpliwie w każdej firmie znajdzie się ktoś przyzwyczajony do ręcznego wykonywania zadań, które można zautomatyzować za pomocą VBA. Jestem przekonany, że mogę pójść do każdej firmy, w której jest ponad 20 użytkowników programu Excel i znajdę podobnie zdumiewający przypadek, jaki spotkał Valerie.

# Funkcje VBA a wersje programu Excel w systemie Windows

Kilka lat temu sporo było pogłosek na temat tego, że firma Microsoft może zaprzestać wspomagania VBA. Obecnie jest jednak wiele dowodów na to, że VBA będzie jeszcze długo dostępne w różnych windowsowych wersjach programu do roku 2036. Kiedy program VBA został usunięty z wersji Excel 2008 dla systemu Mac, liczne głosy protestu klientów doprowadziły do dołączenia tych narzędzi w kolejnej wersji programu Excel dla systemu Mac.

Makra XLM zostały zastąpione przez VBA w roku 1993, a 23 lata później są nadal obsługiwane. Firma Microsoft prowadzi prace nad wprowadzeniem alternatywnego do VBA rozwiązania w oparciu o JavaScript, ale wygląda na to, że program Excel będzie obsługiwał VBA przez następne 23 lata.

## Wersje programu Excel

To piąte wydanie książki *Język VBA i makra* zostało opracowane do współpracy z programem Excel 2016. Poprzednie wydania tej książki uwzględniały kod dla wersji od Excel 97 po Excel 2013. W 80% w rozdziałach kod dla programu Excel 2016 jest identyczny z kodem dla poprzednich wersji, chociaż istnieją wyjątki. Na przykład, nowa funkcja automatycznego grupowania w tabelach przestawnych udostępnia możliwości, których nie mamy w programie Excel 2013.

## Różnice dla użytkowników systemu Mac

Pomimo że program Excel dla systemu Windows i program Excel dla systemu Mac są podobne w kontekście interfejsu, istnieje jednak szereg różnic, jeśli będziemy porównywać środowisko VBA. Rzecz jasna, nic z tego, co w rozdziale 23 wykorzystuje interfejs Windows API, nie będzie działać w systemie Mac. Można powiedzieć, że ogólne koncepcje omawiane w tej książce stosują się także do systemu Mac. Ogólną listę różnic w odniesieniu do systemu Mac znaleźć można pod adresem <http://www.mrexcel.com/macvba.html>. Projektowanie za pomocą VBA dla programu Mac Excel 2016 jest dużo trudniejsze, niż w systemie Windows, ze względu na bardzo podstawowe narzędzia edytowania VBA. Firma Microsoft faktycznie zaleca pisanie kodu VBA w programie Excel 2016 dla systemu Windows, a następnie używanie tego kodu VBA w systemie Mac.

## Elementy specjalne i konwencje typograficzne

W książce używane są następujące konwencje typograficzne:

- **Kursywa** – Wskazuje nowe pojęcie podczas jego definiowania, specjalnie wyróżnienie, obce słowa czy frazy oraz litery czy słowa używane jako słowa.
- **Czcionka o stałej szerokości** – Wskazuje część kodu VBA, na przykład nazwę obiektu czy metody.
- **Czcionka wytłuszczona** – Wskazuje na informacje wprowadzane przez użytkownika.

Oprócz tych konwencji typograficznych, używanych jest też kilka elementów specjalnych. W każdym rozdziale jest co najmniej jedno Studium przypadku, która opisuje rzeczywiste rozwiązania typowych problemów. Analizy przypadków pokazują także praktyczne zastosowania tematyki omawianej w rozdziale.

Oprócz ramek Studium przypadku, zobaczymy również ramki Uwaga, Wskazówka i Ostrzeżenie:

---

**— UWAGA —**


---

Uwagi udostępniają dodatkowe, przydatne informacje spoza głównego wątku omawianych kwestii.

---



---

**— WSKAZÓWKA —**


---

Wskazówki opisują szybkie inne sposoby rozwiązywania omawianego problemu, które oszczędzają czas i umożliwiają bardziej efektywną pracę.

---



---

**— OSTRZEŻENIE —**


---

Ostrzeżenia wskazują na potencjalne problemy i pułapki, z którymi możemy się zetknąć. Warto zapoznać się z tymi informacjami – mogą oszczędzić nam wielu godzin pracy i zdenerwowania.

---

## Pliki kodu

W podziękowaniu za zakup tej książki, dołączyliśmy zestaw blisko 50 arkuszy programu Excel, które ilustrują koncepcje omawiane w książce. Ten zestaw plików obejmuje cały kod omówiony w książce, dane przykładowe, dodatkowe uwagi od autorów oraz 25 dodatkowych makr. Pliki kodu można pobrać ze strony dostępnej pod adresem

<http://http://www.ksiazki.promise.pl/aspx/produkt.aspx?pid=112049>

w zakładce „Dodatkowe informacje”.

## Następne kroki

W rozdziale 1 omówiono narzędzia edytowania dostępne w środowisku Visual Basic oraz przedstawiono przyczyny, które powodują, że używanie rejestratora makr nie jest efektywnym sposobem pisania kodu makra VBA.

# Zwiększanie możliwości programu Excel za pomocą języka VBA

# 1

## Możliwości Excela

Język *Visual Basic for Applications* (VBA) w połączeniu z programem Microsoft Excel jest prawdopodobnie narzędziem o największych możliwościach, jakim dysponują typowi użytkownicy komputerów. Język ten znajdziemy na pulpitach 750 milionów użytkowników pakietu Microsoft Office, ale większości z nich nigdy nie przyszło do głowy, jak można wykorzystać język VBA w programie Excel. Za pomocą VBA możemy przyspieszyć realizację dowolnych zadań w Excelu. Przykładowo, jeśli regularnie wykorzystujemy program Excel do tworzenia szeregu miesięcznych wykresów, VBA wykona za nas to zadanie w ciągu kilku sekund.

## Początkowe przeszkody

Istnieją dwie przeszkody utrudniające opanowanie programowania w VBA. Po pierwsze, rejestrator makr w Excelu nie jest doskonały i nie generuje kodu, który można wykorzystywać jako model. Po drugie, dla wielu osób, które wcześniej poznały inny język programowania, na przykład BASIC, składnia języka VBA jest okropnie frustrująca.

## Rejestrator makr nie działa!

Firma Microsoft zaczęła dominować na rynku arkuszy kalkulacyjnych w połowie lat 90-tych. Chociaż w zasadzie całkowicie pomyślnie zakończyło się zadanie stworzenia złożonego arkusza kalkulacyjnego, na który mogliby przejść użytkownicy programu Lotus 1-2-3,

## W tym rozdziale:

|                                                                                          |    |
|------------------------------------------------------------------------------------------|----|
| Możliwości Excela .....                                                                  | 7  |
| Początkowe przeszkody .....                                                              | 7  |
| Rejestrator makr nie działa! .....                                                       | 7  |
| Nikt w zespole programu Excel nie poświęca wiele uwagi rejestratorowi makr ..            | 8  |
| Visual Basic nie przypomina języka BASIC .....                                           | 8  |
| Dobra wiadomość: „wspinaczka” związana z poznaniem języka VBA nie jest trudna .....      | 9  |
| Doskonała wiadomość: program Excel wraz z językiem VBA wart jest włożonego wysiłku ..... | 9  |
| Znajomość narzędzi: karta Deweloper .....                                                | 10 |
| Typy plików, dla których dopuszczane są makra ..                                         | 11 |
| Bezpieczeństwo makr .....                                                                | 12 |
| Przegląd informacji dotyczących rejestrowania, zapisywania i uruchamiania makr .....     | 15 |
| Uruchamianie makra .....                                                                 | 17 |
| Działanie edytora Visual Basic .....                                                     | 21 |
| Mankamenty rejestratora makr .....                                                       | 23 |
| Następne kroki .....                                                                     | 34 |

to w przypadku makr było zupełnie inaczej. Nawet w przypadku osób z doświadczeniami w tworzeniu makr w programie Lotus 1-2-3 próby rejestrowania makr w Excelu zazwyczaj kończyły się niepowodzeniem. Pomimo że język programowania Microsoft VBA ma znacznie większe możliwości niż język makr programu Lotus 1-2-3, podstawową jego wadą jest to, że rejestrator makr nie działa właściwie przy ustawieniach domyślnych.

W programie Lotus 1-2-3 możemy zarejestrować makro jednego dnia, odtworzyć je następnego i zazwyczaj nie będziemy mieli problemów z jego działaniem. Przy próbie wykonania tych samych czynności w programie Microsoft Excel makro może działać dzisiaj, ale jutro już nie. W 1995 roku, kiedy spróbowałem zarejestrować moje pierwsze makro w programie Excel, takim działaniem programu byłem mocno sfrustrowany. W tej książce nauczymy się trzech reguł najefektywniejszego wykorzystywania rejestratora makr.

## Nikt w zespole programu Excel nie poświęca wiele uwagi rejestratorowi makr

Kiedy w firmie Microsoft dodawane są nowe funkcje w programie Excel, poszczególni menedżerowie projektu danej funkcji zakładają, że rejestrator makr podczas wykonywania polecenia rejestruje odpowiednią sekwencję. W poprzedniej dekadzie, rejestrowany kod mógł działać w niektórych sytuacjach, ale najczęściej nie działał we wszystkich sytuacjach. O ile w firmie Microsoft były osoby zajmujące się tworzeniem przydatnego rejestratora makr, rejestrowany kod mógłby być trochę bardziej ogólny, niż jest obecnie.

Kiedyś zapytałem menedżerów projektów, czy określane są zadania dla rejestratora makr. Zadałem pytanie: „Czy próbowaliście zarejestrować kod, który ma rzeczywiście działać lub czy po prostu próbowali uwidocznić obiekty i metody, by osoba rejestrująca kod musiała przeprowadzić dokładniejszą analizę sposobu stosowania poleceń?” Uzyskane odpowiedzi utwierdziły mnie w przekonaniu, że tak naprawdę nikt w firmie Microsoft nie zwraca uwagi na rejestrator makr.

Kiedyś zazwyczaj można było zarejestrować polecenie jedną z pięciu metod, a zarejestrowany kod przeważnie działał. Niestety obecnie, jeśli chcemy zastosować rejestrator makr, musimy często próbować rejestrowania makra siedmioma różnymi metodami, aż znajdziemy zestaw kroków rejestrujących kod, który działa stabilnie.

## Visual Basic nie przypomina języka BASIC

Dwie dekady wcześniej, kod wygenerowany przez rejestrator makr nie przypominał niczego, z czym można było się spotkać do tej pory. To był „Visual Basic” (VB). W różnych okresach miałem przyjemność nauczania się tuzina języków programowania, ale ten dziwnie wyglądający język był zupełnie nieintuicyjny i nie przypominał języka BASIC poznanego jeszcze w szkole.

Na domiar złego, w 1995 roku uważany byłem w pracy za eksperta w dziedzinie arkuszy kalkulacyjnych. W mojej firmie, w tym właśnie czasie nakazano wszystkim przejście z programu Lotusa 1-2-3 do programu Excel, co oznaczało, że musiałem się teraz zmagać

z rejestratorem makr, który nie działał i z językiem, którego nie rozumiałem, czyli z czymś, co trudno nazwać korzystnym splotem wydarzeń.

Jednym z moich założeń podczas pisania tej książki jest to, że Czytelnicy są sprawnymi użytkownikami arkuszy kalkulacyjnych oraz że Czytelnik książki prawdopodobnie zna ponad 90% osób w swoim biurze. Ponadto, założyłem, że nawet jeśli Czytelnik nie jest programistą, zna podstawy języka BASIC. Jednakże, znajomość tego języka nie jest wymaganiem, a prawdę mówiąc w istocie jest przeszkodą w osiągnięciu celu, jakim jest sprawne programowanie w języku VBA. Istnieje również spora szansa, że Czytelnik próbował rejestrować makro w Excelu i także istnieje podobna szansa, że nie był zadowolony z uzyskanych wyników.

## **Dobra wiadomość: „wspinaczka” związana z poznaniem języka VBA nie jest trudna**

Jeśli nawet próba zastosowania rejestratora makr okazała się frustrująca, można to uznać jedynie za niewielką przeszkodę na drodze do pisania złożonych programów w programie Excel. Z tej książki dowiemy się nie tylko, dlaczego nie działa rejestrator makr, ale także, jak zmodyfikować zarejestrowany kod, by stał się użyteczny. Wszystkim Czytelnikom, którzy programowali już w języku BASIC, opiszę język VBA w taki sposób, by mogli łatwo przeanalizować zarejestrowany kod i zrozumieć jego działanie.

## **Doskonała wiadomość: program Excel wraz z językiem VBA wart jest włożonego wysiłku**

Chociaż pewnie brak możliwości zarejestrowania makr w programie Excel spowodował, że wiele osób poczuło się rozczarowanych firmą Microsoft, trzeba jednak podkreślić, że możliwości języka VBA w programie Excel są ogromne. Absolutnie wszystko, co można wykonać za pomocą interfejsu programu Excel, można też niezwykle szybko osiągnąć używając języka VBA. Jeśli ktoś codziennie czy co tydzień ręcznie tworzy takie same raporty, język VBA w programie Excel znacznie uprości wykonywanie takich zadań.

Autorzy tej książki pracują dla firmy MrExcel Consulting i w ramach tej pracy zautomatyzowali tworzenie raportów dla setek klientów. Te historie zazwyczaj są do siebie podobne: lista zleceń dla działu IT jest tak długa, że ich realizacja zajmuje miesiące. Ktoś z działu księgowego lub konstrukcyjnego odkrywa, że można zaimportować pewne dane do programu Excel i utworzyć raporty potrzebne do działania przedsiębiorstwa. „Odkrycie” to pozwala nie czekać już przez wiele miesięcy, aż dział IT napisze program. Problem polega jednak na tym, że po zaimportowaniu danych do programu Excel i zdobyciu uznania u przełożonego za utworzenie raportu, najprawdopodobniej czekać będzie nas dodatkowa i uciążliwa praca związana z tygodniowym czy comiesięcznym generowaniem tego raportu.

I ponownie, doskonałą w tej sytuacji wiadomością jest to, że wystarczy poświęcić kilka godzin na programowanie w języku VBA, by zautomatyzować proces tworzenia raportu

i sprowadzić go do kilku kliknięć myszą. Satysfakcja jest ogromna, tak więc pozostanie ze mną – wyjaśnię kilka podstawowych zasad.

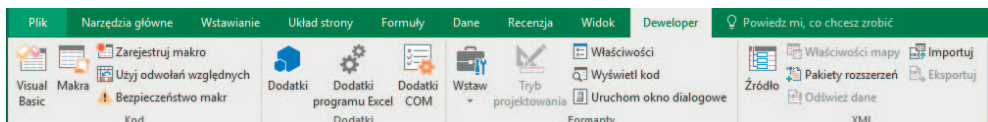
W niniejszym rozdziale wyjaśnimy, dlaczego rejestrator makr nie działa. Przeanalizujemy także przykład zarejestrowanego kodu i pokażemy, dlaczego jednego dnia kod działa, a następnego już nie. W rozdziale znajdują się fragmenty kodu. Zdaję sobie sprawę, że prezentowany w rozdziale kod może nie być zrozumiały dla Czytelnika, ale nie trzeba się tym przejmować. Celem tego rozdziału jest przedstawienie podstawowego problemu dotyczącego rejestratora makr. W rozdziale zaprezentowano również podstawowe elementy środowiska języka Visual Basic.

## Znajomość narzędzi: karta Deweloper

Rozpocznijmy od przeglądu podstawowych narzędzi potrzebnych do korzystania z języka VBA. Domyślnie firma Microsoft ukrywa narzędzia VBA. Aby uzyskać dostęp do karty Deweloper, należy wykonać następujące czynności:

1. Kliknij wstążkę prawym przyciskiem myszy i wybierz polecenie Dostosuj Wstążkę.
2. W okienku z prawej strony zaznacz pole wyboru Deweloper.
3. Kliknij OK, by powrócić do programu Excel.

W programie Excel wyświetlona zostanie karta Deweloper (rysunek 1.1).



Rysunek 1.1 Na karcie Deweloper udostępniono interfejs do uruchamiania i rejestrowania makr.

W grupie Kod na karcie Deweloper znajdują się ikony używane do rejestrowania i odtwarzania makr VBA:

- **Visual Basic** Ikona otwiera narzędzie Visual Basic Editor.
- **Makra** Kliknięcie ikony powoduje wyświetlenie okna dialogowego Makro, gdzie można uruchomić lub edytować makra wymienione na liście.
- **Zarejestruj makro** Kliknięcie ikony rozpoczyna proces rejestrowania makra.
- **Użyj odwołań względnych** Funkcja ta przełącza pomiędzy trybami rejestrowania względnego i bezwzględnego. W przypadku rejestrowania względnego program Excel rejestruje przemieszczenie w dół o trzy komórki. Dla rejestrowania bezwzględnego program Excel rejestruje, że wybrana została komórka A4.
- **Bezpieczeństwo makr** Ikona umożliwia dostęp do modułu Centrum zaufania, gdzie dla używanego komputera można zezwolić na uruchamianie makr lub tego zabronić.

W grupie Dodatki udostępniono narzędzia zarządzania zwykłymi dodatkami i dodatkami COM.

Grupa Formanty (na karcie Deweloper) zawiera menu Wstaw, gdzie znajduje się szereg kontroltek, które można umieszczać na arkuszach. Więcej informacji na ten temat można znaleźć w dalszej części, w podrozdziale „Przypisywanie makra do kontrolki formularza, pola tekstowego lub kształtu”. Przycisk Uruchom okno dialogowe umożliwia wyświetlenie okna dialogowego zdefiniowanego przez użytkownika lub formularza zaprojektowanego za pomocą języka VBA. Więcej informacji na temat formularzy użytkownika znaleźć można w rozdziale 10 „Obiekty UserForm – wprowadzenie”.

W grupie XML na karcie Deweloper umieszczono narzędzia do importowania i eksportowania dokumentów XML.

Grupa Modyfikowanie pozwala określić, czy dla nowych dokumentów zawsze będzie wyświetlane narzędzie Panel dokumentów. W panelu dokumentów użytkownicy mogą wprowadzać słowa kluczowe i opisy dokumentów. Jeśli mamy zainstalowane aplikacje SharePoint i InfoPath, możemy definiować niestandardowe pola, które widoczne będą w panelu dokumentów.

## Typy plików, dla których dopuszczane są makra

Program Excel 2016 obsługuje cztery typy plików. Makra nie mogą być przechowywane w plikach .xlsx, a jak wiadomo, ten rodzaj plików jest typem domyślnym! Musimy używać funkcji Zapisz jako, dla wszystkich naszych arkuszy zawierających makra lub możemy zmienić domyślny typ pliku używany przez program Excel 2016.

Poniżej wymieniono dostępne typy plików:

- **Skoroszyt programu Excel (.xlsx)** Pliki są przechowywane jako szereg obiektów XML, a następnie są pakowane w postaci pojedynczego pliku. Dzięki temu powstają pliki o znacznie mniejszych rozmiarach, a także możliwa jest edycja lub tworzenie skoroszytów programu Excel w innych aplikacjach (nawet w Notatniku!). Niestety makra nie mogą być przechowywane w plikach z rozszerzeniem .xlsx.
- **Skoroszyt programu Excel z obsługą makr (.xlsm)** Jest to podobny typ pliku do domyślnego .xlsx, przy czym włączona jest obsługa makr. Podstawowe założenie polega na tym, żeby użytkownik mając plik .xlsx nie musiał obawiać się złośliwych makr. Jeśli jednak użytkownik widzi plik .xlsm, powinien zwrócić uwagę na to, że do pliku mogą być dołączone makra.
- **Skoroszyt binarny programu Excel (.xlsb)** Jest to binarny format opracowany do obsługi, zawierających ponad milion wierszy (możliwość wprowadzona w programie Excel 2007). Starsze wersje programu Excel przechowują swoje pliki we własnych formatach binarnych. Pomimo że formaty binarne mogą ładować się szybciej, są mniej odporne na uszkodzenia, a utrata kilku bitów może zniszczyć cały plik. W tym formacie makra są obsługiwane.
- **Skoroszyt programu Excel 97-2003 (.xls)** Format ten tworzy pliki, które mogą być odczytywane przez użytkowników starszych wersji programu Excel. W tym formacie binarnym dopuszczona jest obsługa makr; jeśli jednak zapiszemy plik w tym formacie, utracimy dostęp do komórek spoza zakresu A1:IV65536. Ponadto, jeśli otworzymy

plik w programie Excel 2003, utracimy dostęp do elementów, korzystających z funkcji wprowadzonych w programie Excel 2007 lub nowszym.

Aby uniknąć konieczności wybierania typu skrótytu z włączoną obsługą makr (w oknie dialogowym Zapisywanie jako), możemy dostosować naszą kopię programu Excel tak, by nowe pliki były zawsze zapisywane w formacie .xlsm. W tym celu:

1. Kliknij menu Plik i wybierz menu Opcje.
2. W oknie dialogowym Opcje programu Excel, w okienku z lewej strony zaznacz kategorię Zapisywanie.
3. Wyświetl listę rozwijaną Zapisz pliki w następującym formacie i wybierz opcję Skrótytu programu Excel z obsługą makr. Kliknij OK.

---

#### — UWAGA —

Chociaż przeważnie nie boimy się używać makr, spotkałem osoby, które denerwują się, kiedy widzą plik z rozszerzeniem .xlsm, a naprawdę byli zirytowani, kiedy wysłałem im plik .xlsm, który nie miał żadnego makra. Ich reakcja przypomina wypowiedź króla Artura „You got me all worked up!” (zmusiłeś mnie do mnóstwa niepotrzebnej pracy) z komedii „Monty Python i Święty Grail”. Poczta Gmail portalu Google dołączyła do tego poglądu i odrzuciła prezentowanie podglądu załączników wysyłanych w formacie .xlsm.

Jeśli spotkamy kogoś, kto obawia się plików typu .xlsm, warto przypomnieć te kwestie:

- Każdy skrótyt utworzony w ostatnich 30 latach mógł mieć makra, ale w rzeczywistości większość ich nie ma.
- Jeśli ktoś chce uniknąć plików z makrami, powinien stosować ustawienia zabezpieczeń, by zapobiec uruchamianiu makr. Osoba taka będzie nadal mogła otwierać plik .xlsm, by uzyskać dane zawarte w arkuszu.

Mam nadzieję, że dzięki tym argumentom pokonamy wszystkie obawy dotyczące plików .xlsm i staną się one domyślnym typem.

---

## Bezpieczeństwo makr

Po tym, jak makra VBA w programie Word zaczęto wykorzystywać jako metodę przesyłania wirusa Melissa, firma Microsoft zmieniła domyślne ustawienia zabezpieczeń tak, by blokować uruchamianie makr. Z tego powodu, zanim rozpoczniemy omawianie sposobów rejestrowania makr, warto przyjrzeć się, jak dostosować ustawienia domyślne.

W programie Excel 2016 można globalnie zmodyfikować ustawienia zabezpieczeń lub kontrolować ustawienia makr dla wskazanych skrótytów poprzez przechowywanie ich w zaufanej lokalizacji. Makra zostaną automatycznie włączone dla wszystkich skrótytów zapisanych w lokalizacji oznaczonej jako zaufana.

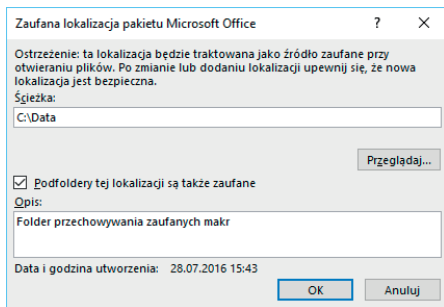
Ustawienia bezpieczeństwa makr znajdziemy, klikając ikonę Bezpieczeństwo makr na karcie Deweloper, co spowoduje wyświetlenie kategorii Ustawienia makr w oknie Centrum zaufania. Okienko nawigacji z lewej strony pozwala uzyskać dostęp do listy Zaufane lokalizacje.

## Dodawanie zaufanej lokalizacji

Skoroszyty zawierające makra możemy przechowywać w folderze oznaczonym jako lokalizacja zaufana. Wszystkie skoroszyty zapisane w zaufanym folderze będą miały włączoną obsługę makr. Firma Microsoft zaleca, by zaufane lokalizacje były definiowane na lokalnych dyskach twardych. Zgodnie z ustawieniami domyślnymi nie możemy ufać lokalizacjom na dyskach sieciowych.

Aby zdefiniować zaufaną lokalizację, wykonujemy poniższą procedurę:

1. Kliknij przycisk Bezpieczeństwo na karcie Deweloper.
2. W okienku nawigacji z lewej strony okna Centrum zaufania, kliknij Zaufane lokalizacje.
3. Jeśli zaufane lokalizacje mają znajdować się na dyskach sieciowych, zaznacz opcję Zezwalaj na zaufane lokalizacje w mojej sieci (niezalecane).
4. Kliknij przycisk Dodaj nową lokalizację. Wyświetlone zostaje okno dialogowe Zaufana lokalizacja pakietu Microsoft Office (rysunek 1.2).



Rysunek 1.2 Zarządzanie zaufanymi folderami w oknie Centrum zaufania.

5. Kliknij przycisk Przeglądaj. Program Excel wyświetli okno dialogowe Przeglądaj.
6. Przejdź do folderu nadrzędnego dla folderu, który ma stać się lokalizacją zaufaną. Kliknij zaufany folder. Chociaż nazwa folderu nie jest wyświetlana w polu Nazwa folderu, kliknij OK. Prawidłowa nazwa folderu będzie widoczna w oknie dialogowym Przeglądaj.
7. Aby podfoldery wybranego folderu również były lokalizacjami zaufanymi, zaznacz pole wyboru Podfoldery tej lokalizacji są także zaufane.
8. Kliknij OK, by dodać folder do listy Zaufane lokalizacje.

### OSTRZEŻENIE

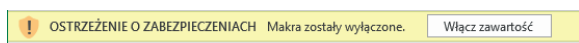
Podczas wybierania zaufanych lokalizacji należy zachować ostrożność. Po kliknięciu załącznika wiadomości e-mail, będącego plikiem programu Excel, program Outlook zapisze ten plik w folderze tymczasowym na dysku C:. Raczej nie będziemy chcieli dodawać całego dysku C:\ i wszystkich jego podfolderów do listy Zaufane lokalizacje.

## Zastosowanie ustawień makr w celu włączenia obsługi makr poza zaufanymi lokalizacjami

Do wszystkich makr zapisanych poza zaufanymi lokalizacjami program Excel stosuje ustawienia makr. Zmienione zostały nazwy ustawień Niskie, Średnie, Wysokie i Bardzo wysokie, które znamy z programu Excels 2003.

Aby uzyskać dostęp do ustawień makr, należy kliknąć polecenie Bezpieczeństwo makr (na karcie Deweloper). Excel wyświetli kategorię Ustawienia makr okna dialogowego Centrum zaufania. Należy zaznaczyć drugą opcję: Wyłącz wszystkie makra i wyświetl powiadomienie. Poniżej przedstawiono opis każdej opcji:

- **Wyłącz wszystkie makra bez powiadomienia** Ustawienie to blokuje uruchamianie wszystkich makr i jest przeznaczone dla osób, które nigdy nie zamierzają korzystać z makr. Ponieważ książka uczy, jak używać makr, nie będziemy używać tej opcji. Ustawienie to, ogólnie rzecz biorąc, jest odpowiednikiem opcji Bardzo wysokie w programie Excel 2003. W przypadku tego ustawienia mogą działać jedynie makra zapisane w zaufanych lokalizacjach.
- **Wyłącz wszystkie makra i wyświetl powiadomienie** W opisie działania ustawienia dołączono słowa „wyświetl powiadomienie”. Oznacza to, że użytkownikowi wyświetlone zostanie powiadomienie, kiedy otworzy plik zawierający makra i użytkownik zdecyduje, czy chce włączyć zawartość. Jeśli powiadomienie zostanie zignorowane, makra pozostaną wyłączone. Jest to ustawienie podobne do opcji Średnie w programie Excel 2003 i jest to zalecane ustawienie. W programie Excel 2016, w obszarze komunikatów wyświetlane jest powiadomienie, informujące o tym, że makra zostały wyłączone. Klikając tę opcję, użytkownik może włączyć zawartość (rysunek 1.3).



Rysunek 1.3. Przycisk Włącz zawartość widoczny w przypadku użycia opcji Wyłącz wszystkie makra i wyświetl powiadomienie.

- **Wyłącz wszystkie makra oprócz makr podpisanych cyfrowo** Ustawienie to wymaga użycia narzędzia do tworzenia podpisów cyfrowych firmy VeriSign lub innego dostawcy. Jest to właściwy wybór dla osób zamierzających sprzedawać dodatki innym użytkownikom, ale jest trochę uciążliwym rozwiązaniem w przypadku pisania makr na własny użytek.
- **Włącz wszystkie makra (niezalecane, może zostać uruchomiony niebezpieczny kod)** Ustawienie to jest podobne do ustawienia Niskie w programie Excel 2003. Chociaż ustawienie sprawia najmniej kłopotów, jednak naraża komputer na ataki złośliwych wirusów, podobnych w działaniu do wirusa Melissa. Firma Microsoft nie zaleca używania tego ustawienia.

## Stosowanie opcji Wyłącz wszystkie makra i wyświetl powiadomienie

Zalecane jest stosowanie ustawienia Wyłącz wszystkie makra i wyświetl powiadomienie. W przypadku użycia tej opcji po otwarciu skoroszytu zawierającego makra, bezpośrednio nad paskiem formuły wyświetli się ostrzeżenie o zabezpieczeniach. Jeśli spodziewamy się, że ten skoroszyt zawiera makra, należy kliknąć przycisk Włącz zawartość. Jeśli nie chcemy włączać makr w otwieranym skoroszycie, możemy zamknąć ostrzeżenie o zabezpieczeniach poprzez kliknięcie ikonki X z prawej strony paska tytułu.

Jeśli zapomnimy wyłączyć makra i spróbujemy je uruchomić, program Excel poinformuje nas, że nie można uruchomić makra, ponieważ wszystkie zostały wyłączone. W takiej sytuacji najlepiej jest zamknąć skoroszyt i otworzyć go jeszcze raz.

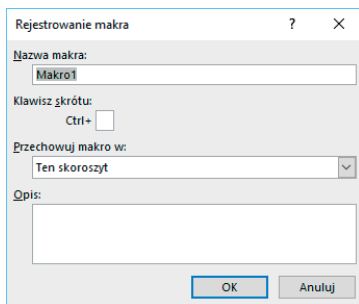
### — OSTRZEŻENIE —

Jeśli włączymy makra w skoroszycie przechowywanym na lokalnym dysku twardym, a następnie go zapiszemy, program Excel zapamięta, że w tym skoroszycie włączaliśmy makra. Przy kolejnym otwieraniu tego skoroszytu obsługa makr będzie automatycznie włączona.

## Przegląd informacji dotyczących rejestrowania, zapisywania i uruchamiania makr

Rejestrowanie makr jest przydatne dla osób, które nie mają odpowiedniego doświadczenia, by samodzielnie napisać ich kod. Po zdobyciu potrzebnej wiedzy i doświadczenia coraz rzadziej korzysta się z możliwości rejestrowania makr.

Aby rozpocząć rejestrację makra, na karcie Deweloper wybieramy polecenie Zarejestruj makro. Przed rozpoczęciem rejestrowania program Excel wyświetla okno dialogowe Rejestrowanie makra (rysunek 1.4).



Rysunek 1.4. W oknie dialogowym Rejestrowanie makra, do rejestrowanego makra przypisujemy nazwę i klawisz skrótu

## Wypełnianie okna dialogowego Rejestrowanie makra

W polu Nazwa makra wpisujemy jego nazwę. Trzeba pamiętać, by nie używać spacji, przykładowo należy użyć nazwy **Makro1**, a nie **Makro 1**. Ponieważ makro może być bardzo duże, warto stosować nazwy opisowe, na przykład nazwa FormatowanieRaportu jest znacznie bardziej przydatna niż Makro1.

W drugim polu okna dialogowego Rejestrowanie makra definiujemy klawisz skrót. Jeśli w tym polu wpiszymy małą literę j, a następnie naciśniemy klawisze Ctrl+J, makro to zostanie uruchomione. Warto pamiętać jednak, że większość skrótów z literami, począwszy od Ctrl+A, a skończywszy na Ctrl+Z (za wyjątkiem Ctrl+J) jest już w programie Excel wykorzystywanych do innych zadań. Jednym z rozwiązań jest przypisywanie do makra połączenia klawiszy Ctrl+Shift+A aż do Ctrl+Shift+Z. Aby przypisać do makra skrót Ctrl+Shift+A, w polu skrótu wpisujemy Shift+A.

---

### —OSTRZEŻENIE—

Przypisanie skrótów do makr zmienia dotychczasowe przypisanie klawiszy. Przykładowo, jeśli przypiszemy do makra kombinację Ctrl+C, program Excel zamiast wykonać standardowe kopiowanie, uruchomi makro.

---

W oknie dialogowym Rejestrowanie makra można zdecydować o tym, gdzie makro ma być zapisane po zarejestrowaniu. Dostępne opcje to: Skoroszyt makr osobistych, Nowy skoroszyt oraz Ten skoroszyt. Zaleca się, by zapisywać makra związane z danym skoroszytem przy użyciu opcji Ten skoroszyt.

Skoroszyt makr osobistych (Personal.xlsm) nie jest widocznym skoroszytem – skoroszyt jest tworzony, jeśli do zapisu wybierzemy opcję Skoroszyt makr osobistych. Skoroszyt ten jest używany do zapisywania makra w skoroszycie otwieranym automatycznie podczas uruchamiania programu Excel i dzięki temu od razu będzie można korzystać z makra. Po uruchomieniu programu Excel skoroszyt jest ukrywany. Aby wyświetlić ten skoroszyt, należy wybrać opcję Odkryj okno na karcie Widok.

---

### —WSKAZÓWKA—

Nie zaleca się używania skoroszytu makr osobistych dla wszystkich zapisywanych makr. Należy zapisywać w nim tylko te makra, które pomagają w wykonywaniu zadań ogólnego przeznaczenia, a nie zadań, które wykonywane są dla określonego arkusza lub skoroszytu.

---

W czwartym polu okna dialogowego Rejestrowanie makra wpisujemy opis. Opis ten zostanie dodany w formie komentarza na początku makra.

Po wybraniu lokalizacji, w której ma być zapisane makro, należy kliknąć OK. Teraz trzeba zarejestrować makro. Na przykład, w aktywnej komórce wpisujemy Hello World i naciskamy Ctrl+Enter, by zaakceptować wpis i pozostać w tej samej komórce. Po zakończeniu rejestrowania makra klikamy ikonę Zatrzymaj rejestrowanie (na karcie Deweloper).

---

**WSKAZÓWKA**

---

Ikona Zatrzymaj rejestrowanie jest również dostępna w lewym dolnym rogu okna programu Excel i wygląda jak niewielki niebieski kwadrat. Znajduje się z prawej strony słowa *Gotowy* na pasku stanu. Użycie tego przycisku Zatrzymaj może być czasami wygodniejsze niż powrót karty Deweloper. Po zarejestrowaniu pierwszego makra w tym obszarze zazwyczaj widoczna jest ikona Rejestruj makro.

---

## Uruchamianie makra

Jeśli przypisaliśmy klawisz skrótu do makra, możemy je uruchomić poprzez wciśnięcie tej kombinacji klawiszy. Makra można także przypisywać do przycisków na wstążce lub na pasku narzędzi Szybki dostęp, do kontrolek formularza, do obiektów graficznych lub też możemy je uruchamiać na pasku narzędzi programu Visual Basic.

### Tworzenie przycisku makra na wstążce

W celu uruchomienia makra możemy dodać ikonę do nowej grupy na wstążce. Jest to odpowiednie działanie w przypadku makr przechowywanych w skoroszytcie makr osobistych. Ikony dodane do wstążki pozostają dostępne, nawet jeśli skoroszyt z makrem nie jest otwarty. Jeśli klikniemy ikonę, kiedy skoroszyt nie jest otwarty, program Excel otworzy skoroszyt i uruchomi makro. Aby dodać przycisk makra na wstążce, wykonujemy następujące instrukcje:

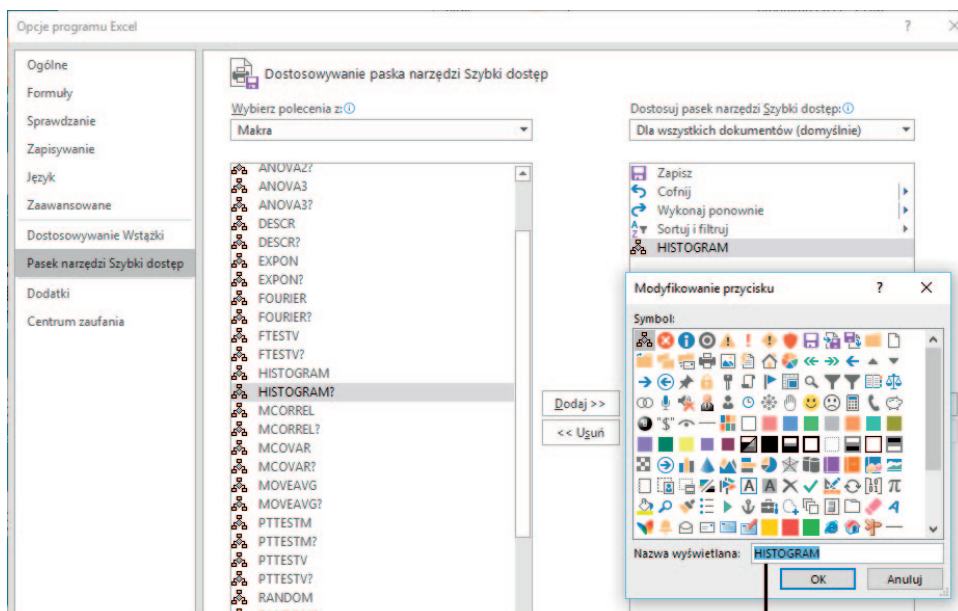
1. Kliknij wstążkę prawym przyciskiem myszy, a następnie wybierz polecenie *Dostosuj Wstążkę*.
2. Na liście z prawej strony wybierz nazwę karty, do której ma być dodana ikona.
3. Kliknij przycisk *Nowa grupa* poniżej listy z prawej strony. Program Excel doda nową pozycję nazwaną *Nowa grupa* (Niestandardowa) na końcu listy grup wybranej karty.
4. Aby przenieść grupę na karcie wstążki w lewo, kilkakrotnie kliknij ikonę strzałki w górę (z prawej strony okna dialogowego).
5. Aby zmienić nazwę grupy, kliknij przycisk *Zmień nazwę*. Wpisz nową nazwę, na przykład *Makra raportów*. Kliknij OK. Excel wyświetli grupę na liście jako *Makra raportów* (Niestandardowa). Należy zwrócić uwagę, że słowo *Niestandardowa* nie pojawi się na wstążce.
6. Otwórz listę rozwijaną w lewym górnym narożniku i wybierz opcję *Makra* (czwarta kategoria na liście). W oknie listy program Excel wyświetli listę dostępnych makr.
7. Wybierz makro z listy po lewej stronie. Kliknij przycisk *Dodaj* znajdujący się po środku okna dialogowego. Program Excel przeniesie makro na listę z prawej strony do wybranej grupy. Dla wszystkich makr stosowana jest ogólna ikona VBA.
8. Kliknij makro na liście z prawej strony. Kliknij przycisk *Zmień nazwę*, poniżej tej listy. Program Excel wyświetli listę 180 ikon do wyboru. Wybierz ikonę. Możesz również wpisać opisową etykietę dla tej ikony, na przykład *Formatuj raport*.

9. Kliknij OK, aby zamknąć okno Opcje programu Excel. Na wybranej karcie wstążki pojawi się nowy przycisk.
10. Grupę Makra raportów możesz przenieść w inne miejsce na karcie wstążki. W tym celu kliknij pozycję Makra raportów (Niestandardowa) i użyj strzałek w górę i w dół z prawej strony okna dialogowego.
11. Kliknij OK, by zamknąć okno Opcje programu Excel. Na wybranej karcie wstążki pojawi się nowy przycisk.

## Tworzenie przycisku makra na pasku narzędzi Szybki dostęp

W celu uruchomienia makra można dodać ikonę do paska narzędzi Szybki dostęp. Jeśli makro zostało zapisane w skoroszybie makr osobistych, do makra można przypisać przycisk, który będzie się stale wyświetlał na pasku narzędzi Szybki dostęp. Jeśli makro jest zapisane w bieżącym skoroszybie, można określić, że ikona będzie wyświetlana tylko wtedy, gdy skoroszyt jest otwarty. Aby dodać przycisk makra do paska narzędzi Szybki dostęp, wykonujemy poniższe czynności:

1. Kliknij prawym przyciskiem myszy pasek narzędzi Szybki dostęp, a następnie wybierz polecenie Dostosuj pasek narzędzi Szybki dostęp.
2. Jeśli makro ma być dostępne tylko wtedy, gdy jest otwarty bieżący skoroszyt, rozwiń listę w prawym górnym narożniku i zmień opcję Dla wszystkich dokumentów (Domyślnie) na opcję Dla *NazwaPliku.xlsm*. Ikony powiązane z bieżącym skoroszytem wyświetlane są na końcu paska narzędzi Szybki dostęp.
3. Rozwiń listę w górnej lewej części okna i wybierz pozycję Makra (czwarta kategoria na liście). Program Excel wyświetli listę dostępnych makr.
4. W oknie z lewej strony wybierz makro z listy. Kliknij przycisk Dodaj znajdujący się na środku okna dialogowego. Program Excel przeniesie makro na listę po prawej stronie. Dla wszystkich makr w programie Excel stosowana jest ogólna ikona VBA.
5. Kliknij makro na liście z prawej strony. Kliknij przycisk Modyfikuj (poniżej okienka listy z prawej strony). Excel wyświetli listę 180 dostępnych ikon (rysunek 1.5). Wybierz ikonę z listy. W polu Nazwa wyświetlania zastąp nazwę makra nazwą skrótową, która widoczna będzie w etykietce ekranowej ikony.
6. Kliknij OK, by zamknąć okno dialogowe Modyfikowanie przycisku.
7. Kliknij OK, by zamknąć okno dialogowe Opcje programu Excel. Na pasku narzędzi Szybki dostęp wyświetli się nowy przycisk.



W tym polu wprowadź odpowiedź

Rysunek 1.5 Dodawanie przycisku makra do paska narzędzi Szybki dostęp

## Przypisywanie makra do kontrolki formularza, pola tekstowego lub kształtu

Jeśli chcemy utworzyć makro specyficzne dla danego skoroszytu, zapisujemy je w skroścyce i dołączamy do kontrolki formularza lub dowolnego obiektu arkusza.

Aby dołączyć makro do kontrolki formularza na arkuszu:

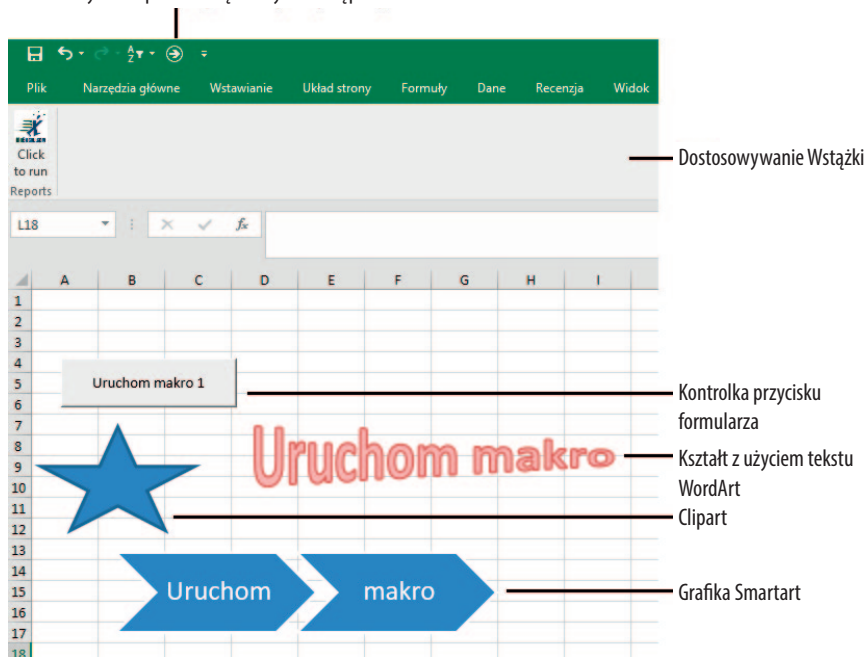
1. Na karcie Deweloper kliknij przycisk Wstaw, by wyświetlić listę rozwijaną tego przycisku. Na tej liście program Excel udostępnia 12 kontrolkek formularza i 12 kontrolkek ActiveX. Na początku prezentowane są kontrolki formularza, a u dołu kontrolki ActiveX. Większość ikon na tej liście w części dotyczącej kontrolkek ActiveX wygląda identycznie jak ikonki kontrolkek formularza. Kliknij ikonę Przycisk (formant formularza) w lewym górnym narożniku listy rozwijanej Wstaw.
2. Przenieś kursor nad arkusz; kursor zmieni się na znak plus.
3. Narysuj przycisk na arkuszu. W tym celu kliknij i przytrzymaj lewy przycisk myszy, rysując jednocześnie kształt pola. Po zakończeniu rysowania zwolnij przycisk myszy.
4. Wybierz makro w oknie dialogowym Przypisywanie makra i kliknij OK. Spowoduje to utworzenie przycisku wraz z ogólnym opisem typu Przycisk 1.
5. Wpisz nową etykietę przycisku. Podczas pisania ramka zaznaczenia wokół przycisku zmienia się z kropek na skośne linie oznaczające tryb edycji tekstu. W trybie edycji tekstu nie można zmienić koloru przycisku. Aby wyjść z trybu edycji tekstu, należy kliknąć skośne linie (co spowoduje, że zmienią się w kropki) lub ponownie

nacisnąć klawisz Ctrl i kliknąć przycisk. Jeśli przypadkowo klikniemy poza przyciskiem, w celu ponownego zaznaczenia przycisku, klikamy przycisk przy naciśniętym klawiszu Ctrl. Następnie należy przeciągnąć kursor nad tekstem, aby go zaznaczyć.

6. Kliknij prawym przyciskiem myszy ramkę w postaci kropek wokół przycisku i wybierz polecenie Formatuj formant. Wyświetlone zostaje okno dialogowe Formatowanie formantu, składające się z siedmiu zakładek. Jeśli okno dialogowe Formatowanie formantu zawiera tylko zakładkę Czcionka, oznacza to, że nadal aktywny jest tryb edycji tekstu. W takim przypadku zamknij okno dialogowe, naciśnij Ctrl i kliknij przycisk, a następnie powtórz ten krok.
7. Użyj opcji w oknie dialogowym Formatowanie formantu, by zmienić rozmiar czcionki, kolor czcionki, marginesy oraz inne ustawienia kontrolki. Po zakończeniu formatowania kliknij OK, by zamknąć okno dialogowe. Kliknij komórkę, by usunąć zaznaczenie przycisku.
8. Kliknij nowy przycisk, aby uruchomić makro.

Makra można przypisywać do dowolnych obiektów arkusza, takich jak grafiki clipart, kształty, obiekty SmartArt czy pola tekstowe. Na rysunku 1.6 górny przycisk ma kształt standardowego przycisku kontrolki formularza. Pozostałe obrazy to obiekt clipart, tekst WordArt i grafika SmartArt. W celu przypisania makra do dowolnego obiektu klikamy obiekt prawym przyciskiem myszy i wybieramy polecenie Przypisz makro.

Dostosowywanie paska narzędzi Szybki dostęp

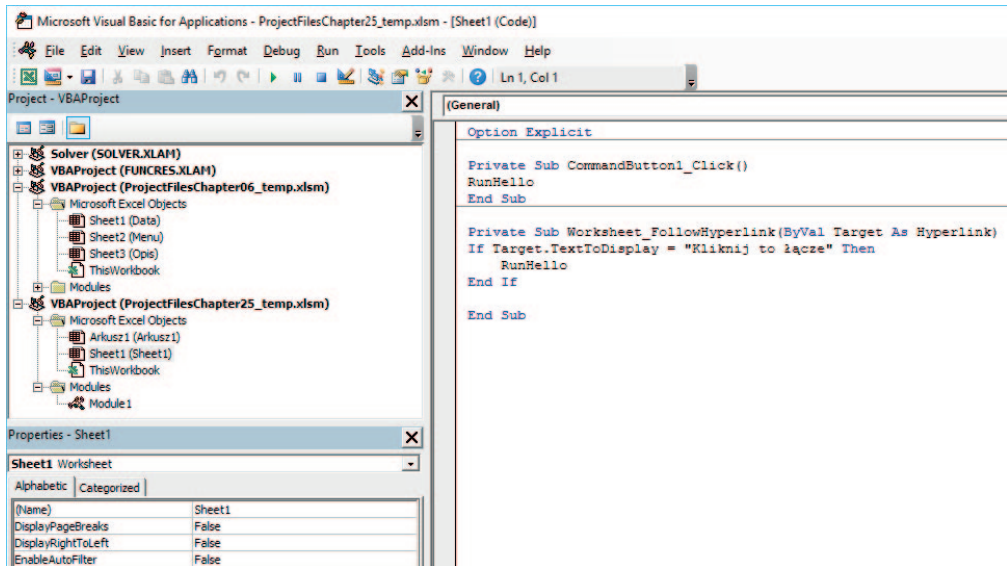


**Rysunek 1.6.** Przypisywanie makra do kontrolki formularza lub obiektu zapisanego w tym samym skoroszybie co makro. Makro można przypisać do dowolnego spośród tych obiektów

## Działanie edytora Visual Basic

Za pomocą narzędzia VB Editor przeprowadzamy edycję zarejestrowanego makra. Aby uruchomić edytor, naciskamy klawisze Alt+F11 lub używamy ikony Visual Basic na karcie Deweloper.

Na rysunku 1.7 pokazano przykład typowego ekranu edytora VB, na którym widoczne są trzy okna: Project Explorer, Properties i Programming (okno eksploratora projektu, właściwości i kodu). Nie należy się przejmować tym, że na komputerze mamy trochę inny ekran, niż okno prezentowane na rysunku, ponieważ podczas omawiania edytora dowiemy się, jak wyświetlać potrzebne okna.



Rysunek 1.7 Okno VB Editor

## Ustawienia narzędzia VB Editor

Szereg opcji edytora VB umożliwia dostosowanie narzędzia, by łatwiej było nam pisać makra.

Klika przydatnych ustawień udostępnionych jest na karcie Tools/Options/Editor. Wszystkie opcje, poza jedną, mają odpowiednie domyślne ustawienie. Jedno ustawienie wymaga podjęcia decyzji ze strony użytkownika. Chodzi o opcję Require Variable Declaration (wymagaj deklarowania zmiennych). Zgodnie z ustawieniami domyślnymi program Excel nie wymaga deklarowania zmiennych. Osobiście preferuję to ustawienie, ponieważ pozwala ono zaoszczędzić czas podczas tworzenia programów. Współautorka tej książki jest jednak odmiennego zdania. Zmiana ta powoduje zatrzymanie kompilatora, jeśli napotka zmienną, której nie rozpoznaje, co zmniejsza liczbę pomyłek w nazwach zmiennych. Włączenie lub wyłączenie tej opcji zależy od osobistych preferencji użytkownika.

## Eksplorator projektu

W oknie eksploratora projektu wymienione są wszystkie otwarte skoroszyty i załadowane dodatki. Jeśli klikniemy ikonę +, obok pozycji VBAProject, zobaczymy folder Microsoft Excel Objects. Mogą tam być również foldery formularzy, modułów klas i modułów standardowych. Każdy folder zawiera jeden lub kilka komponentów.

Kliknięcie komponentu prawym przyciskiem myszy i wybranie polecenia View Code lub dwukrotne kliknięcie komponentu powoduje wyświetlenie kodu w oknie Programming. Wyjątek stanowią formularze użytkowników – w tym przypadku dwukrotne kliknięcie powoduje wyświetlenie formularza w widoku projektu (Design).

Aby wyświetlić okno Project Explorer, należy z menu wybrać polecenie View/Project Explorer, nacisnąć klawisze Ctrl+R lub znaleźć i kliknąć dziwną ikonę eksploratora projektu na pasku narzędzi, poniżej menu Tools, znajdującą się pomiędzy ikonami Design Mode i Properties Window.

Aby wstawić moduł, klikamy projekt prawym przyciskiem myszy, wybieramy polecenie Insert, a następnie wskazujemy typ modułu. Poniżej wymieniono dostępne moduły:

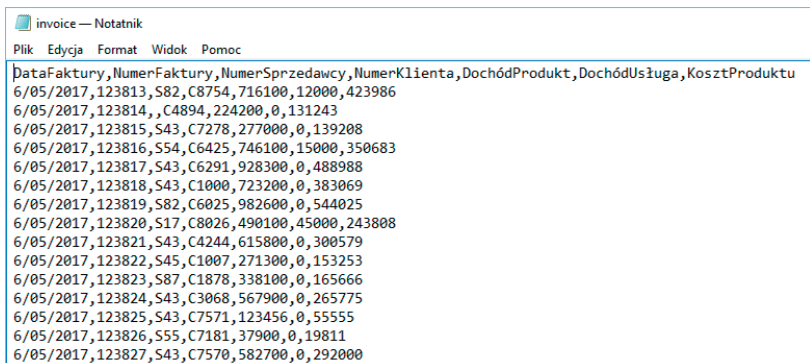
- **Obiekty Microsoft Excel** Domyślnie projekt zawiera moduły arkuszy dla każdego arkusza w skoroszytcie oraz jeden moduł ThisWorkbook (Ten skoroszyt). Kod specyficzny dla arkusza, taki jak kontrolki czy zdarzenia związane z arkuszem, umieszczany jest w odpowiadającym mu arkuszu. Kod obsługi zdarzeń skoroszytu umieszczony jest w module ThisWorkbook. Więcej informacji na temat zdarzeń można znaleźć w rozdziale 7 „Programowanie zdarzeń”.
- **Formularze** Program Excel pozwala zaprojektować własne formularze do komunikowania się z użytkownikami. Więcej informacji na temat formularzy można znaleźć w rozdziale 10 „Obiekty UserForm – wprowadzenie”.
- **Moduły** Podczas rejestrowania makra program Excel automatycznie tworzy moduł, w którym umieszcza kod. Większość tworzonego przez nas kodu umieszczana jest w modułach tego typu.
- **Moduły klas** Moduły klas to w programie Excel metoda tworzenia własnych obiektów. Moduły klas pozwalają również na współdzielenie fragmentów kodu pomiędzy programistami bez konieczności poznania sposobu działania kodu. Więcej informacji na temat modułów klas znaleźć można w rozdziale 9 „Tworzenie klas, rekordów i kolekcji”.

## Okno Properties

Okno Properties umożliwia edycję właściwości różnych komponentów, takich jak arkusze, skoroszyty, moduły i kontrolki formularzy. Lista właściwości może być różna, w zależności od wybranego komponentu. Aby wyświetlić to okno, z menu wybieramy polecenie View/Properties Window, wciskamy klawisz F4 lub klikamy ikonę okna Properties na pasku narzędzi.

## Mankamenty rejestratora makr

Założmy, że pracujemy w dziale księgowości. Każdego dnia otrzymujemy plik tekstowy z systemu firmy z listą wszystkich faktur wystawionych poprzedniego dnia. W tym pliku tekstowym poszczególne pola są rozdzielone przecinkami. Kolumny w pliku to DataFaktury, NumerFaktury, NumerSprzedawcy, NumerKlienta, DochódProdukt, DochódUsługa oraz KosztProduktu (rysunek 1.8).



```
invoice — Notatnik
Plik Edycja Format Widok Pomoc
DataFaktury,NumerFaktury,NumerSprzedawcy,NumerKlienta,DochódProdukt,DochódUsługa,KosztProduktu
6/05/2017,123813,S82,C8754,716100,12000,423986
6/05/2017,123814,,C4894,224200,0,131243
6/05/2017,123815,S43,C7278,277000,0,139208
6/05/2017,123816,S54,C6425,746100,15000,350683
6/05/2017,123817,S43,C6291,928300,0,488988
6/05/2017,123818,S43,C1000,723200,0,383069
6/05/2017,123819,S82,C6025,982600,0,544025
6/05/2017,123820,S17,C8026,490100,45000,243808
6/05/2017,123821,S43,C4244,615800,0,300579
6/05/2017,123822,S45,C1007,271300,0,153253
6/05/2017,123823,S87,C1878,338100,0,165666
6/05/2017,123824,S43,C3068,567900,0,265775
6/05/2017,123825,S43,C7571,123456,0,55555
6/05/2017,123826,S55,C7181,37900,0,19811
6/05/2017,123827,S43,C7570,582700,0,292000
```

Rysunek 1.8 Plik Invoice.txt

Każdego ranka ręcznie importujemy ten plik do programu Excel. Dodajemy wiersz podsumowania, pogrubiamy nagłówki, a następnie drukujemy raport i przekazujemy go kilku menedżerom.

Wydaje się, że jest to prosty proces, idealnie nadający się do wykorzystania rejestratora makr. Jednak z powodu pewnych problemów z rejestratorem makr pierwsze próby wykonania tego procesu nie muszą zakończyć się sukcesem. W zamieszczonej poniżej analizie przypadku wyjaśniamy, jak pokonać te problemy.

### Studium przypadku: Przygotowanie do rejestracji makra

Zadanie opisane w poprzednim fragmencie idealnie nadaje się do utworzenia makra. Przed zarejestrowaniem makra należy jednak przeanalizować wykonywane czynności. W naszym przypadku są to następujące operacje:

1. Kliknięcie menu Plik i wybranie polecenia Otwórz.
2. Przejście do folderu, w którym jest zapisany plik invoice.txt.
3. Z listy rozwijanej Pliki typu wybranie opcji Wszystkie pliki.
4. Wybranie pliku invoice.txt.
5. Kliknięcie przycisku Otwórz.
6. W oknie Kreator importu tekstu – krok 1 z 3, w sekcji Typ danych źródłowych zaznaczenie opcji Rozdzielany.
7. Kliknięcie przycisku Dalej.

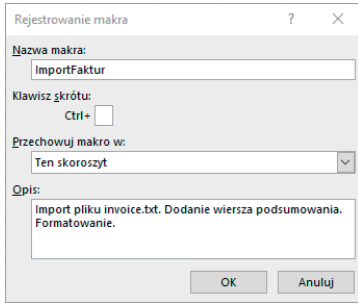
8. W oknie Kreator importu tekstu – krok 2 z 3, w ramce Ograniczniki usunięcie zaznaczenia pola wyboru Tabulator i zaznaczenie pola wyboru Przecinek.
9. Kliknięcie Dalej.
10. W oknie Kreator importu tekstu – krok 3 z 3, w ramce Format danych w kolumnie wybranie opcji Data: MDR.
11. Kliknięcie przycisku Zakończ, aby zaimportować plik.
12. Naciśnięcie klawisza End i strzałka w dół, by przejść do ostatniego wiersza danych.
13. Ponownie wciśnięcie strzałki w dół, by przejść do wiersza podsumowania.
14. Wpisanie słowa Razem.
15. Cztery razy naciśnięcie klawisza strzałka w prawo, by przejść do kolumny E wiersza podsumowania.
16. Kliknięcie przycisku Autosumowanie, a następnie naciśnięcie klawiszy Ctrl+Enter, by dodać podsumowanie do kolumny DochódProdukt i jednocześnie pozostać w tej komórce.
17. Przeciągnięcie uchwyty automatycznego wypełniania z kolumny E do kolumny G w celu skopiowania formuły podsumowującej do kolumn F i G.
18. Zaznaczenie pierwszego wiersza i kliknięcie ikony Pogrubienie na karcie Narzędzia główne, by wyróżnić czcionkę nagłówek.
19. Zaznaczenie wiersza podsumowania i kliknięcie ikony Pogrubienie, by wyróżnić zawartość tego wiersza.
20. Wciśnięcie Ctrl+A, by zaznaczyć bieżący obszar.
21. Na karcie Narzędzia główne wybieranie narzędzia Formatuj i opcji Autodopasowanie szerokości kolumn.

Po zebraniu czynności, które należy wykonać, możemy przystąpić do rejestracji pierwszego makra. Otwieramy pusty skoroszyt i zapisujemy go na przykład pod nazwą MacroToImportInvoices.xlsm. Na karcie Deweloper klikamy teraz przycisk Zarejestruj makro.

W oknie Rejestrowanie makra domyślna nazwa makra to Makro1. Zmieniamy tę nazwę na bardziej opisową, przykładowo ImportFaktur. Upewniamy się, że do zapisu makra wybrana jest opcja Ten skoroszyt. Ponieważ później przyda się łatwy sposób uruchamiania tego makra, w polu Klawisz skrótu wpisujemy literę i. W polu Opis dodajemy krótki opis działania makra (rysunek 1.9). Po wykonaniu tych czynności klikamy OK.

## Rejestrowanie makra

Teraz rejestrator makr zarejestruje każdy nasz ruch. Z tego względu należy starać się wykonywać wszystkie czynności po kolei, bez żadnych dodatkowych działań. Jeśli przypadkowo przejdziemy do kolumny F, a następnie z powrotem do kolumny E w celu wprowadzenia pierwszej sumy, zarejestrowane makro będzie codziennie „ślepo” powielało tę samą



**Rysunek 1.9** Przed zarejestrowaniem makra należy wypełnić pola w oknie dialogowym Rejestrowanie makra.

pomyłkę. Zarejestrowane makra działają szybko, ale nie warto przyglądać się, jak każdorazowo odtwarzane są wszystkich nasze pomyłki.

Należy zatem uważnie wykonać wszystkie działania niezbędne do utworzenia raportu. Po wykonaniu ostatniej czynności klikamy przycisk Zatrzymaj rejestrowanie (na karcie Deweloper).

## Analiza kodu w oknie programowania

Spróbujmy przyjrzeć się kodowi, który przed chwilą zarejestrowaliśmy w analizie przypadku. Nic nie szkodzi, jeśli na razie kod ten nie jest zrozumiały.

Aby otworzyć edytor VB, naciskamy klawisze Alt+F11. Dla pozycji VBA Project (Makro-DoImportowaniaFaktur.xlsm) odnajdujemy komponent Module1. Trzeba go kliknąć prawym przyciskiem myszy i wybrać polecenie View Code. Warto zwrócić uwagę, że niektóre wiersze rozpoczynają się od apostrofu; są to komentarze, które są ignorowane przez program. Rejestrator makr rozpoczyna makra od kilku komentarzy. Do tego celu wykorzystuje opis, który wprowadziliśmy w oknie dialogowym Rejestrowanie makra. Komentarz dotyczący klawisza skrótów ma przypominać, jaki klawisz skrót wybraliśmy dla makra.

---

### — UWAGA —

Komentarz *nie* powoduje przypisania klawisza skrótów. Jeśli zmodyfikujemy komentarz i wprowadzimy tam ciąg Ctrl+J, nie spowoduje to zmiany klawisza skrótów przypisanego do makra. W celu zmiany klawisza skrótów należy zmienić ustawienie w oknie dialogowym Makro w Excelu lub uruchomić poniższy kod:

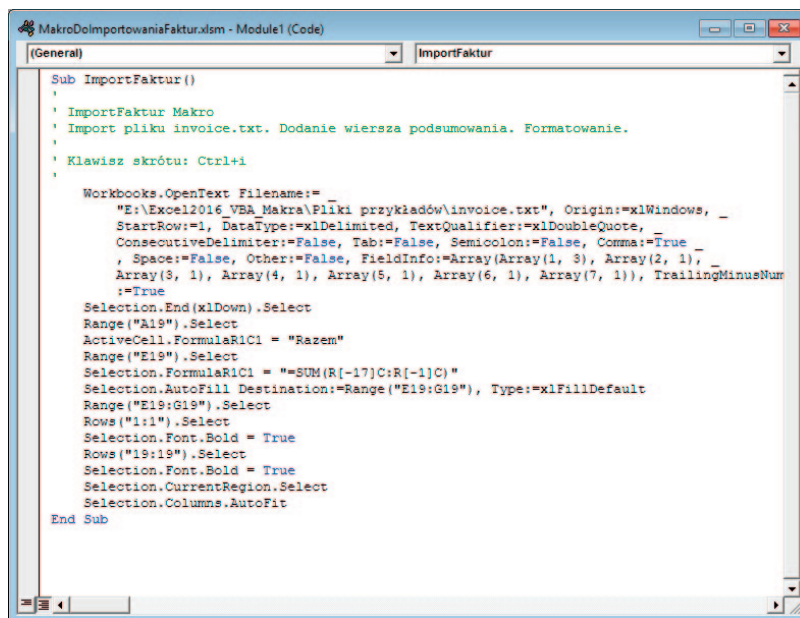
```
Application.MacroOptions Macro:="ImportFaktur", _
    Description:="", ShortcutKey:="j"
```

---

Zarejestrowane makro jest zazwyczaj dość staranne (rysunek 1.10). W każdym wierszu kodu (poza wierszami komentarza) jest wcięcie o szerokości czterech znaków. W celu kontynuacji wiersza kodu w następnym wierszu należy na końcu wiersza umieścić spację i znak podkreślenia. Nie zapominajmy o spacji przez znakiem podkreślania – jej brak spowoduje błąd.

**— UWAGA —**

Ze względu na fizyczne rozmiary tej książki w pojedynczym wierszu nie zmieści się 100 znaków. Dlatego wiersze będą dzielone przy szerokości około 80 znaków, by mogły zmieścić się na stronie. Z tych też powodów makro, które zarejestrowaliśmy na komputerze, może się nieco różnić od prezentowanego w książce.



Rysunek 1.10 Zarejestrowane makro ma schludny wygląd i estetyczne wcięcia

Zwróćmy uwagę, że poniższe siedem wierszy zarejestrowanego kodu to w rzeczywistości tylko jeden wiersz, który został podzielony w celu poprawy czytelności:

```
Workbooks.OpenText Filename:= _
    "E:\Excel2016_VBA_Makra\Pliki przykladow\invoice.txt", _
    Origin:=xlWindows, StartRow:=1, DataType:=xlDelimited, _
    TextQualifier:=xlDoubleQuote, ConsecutiveDelimiter:=False, _
    Tab:=False, Semicolon:=False, Comma:=True, Space:=False, _
    Other:=False, FieldInfo:=Array(Array(1, 3), Array(2, 1), _
    Array(3, 1), Array(4, 1), Array(5, 1), Array(6, 1), Array(7, 1)), _
    TrailingMinusNumbers:=True
```

Uwzględniając, że powyższy kod to jeden wiersz, widzimy, że rejestrator makr zarejestrował proces składający się z 21 kroków w 14 wierszach kodu. Robi wrażenie.

**— UWAGA —**

Każde działanie wykonywane za pomocą interfejsu użytkownika programu Excel może odpowiadać jednemu wierszowi lub kilku wierszom zarejestrowanego kodu. Niektóre działania mogą generować kilkanaście wierszy kodu.

## Sprawdzanie każdego makra

Zawsze warto sprawdzać makra. W celu przetestowania nowego makra powrócimy do standardowego interfejsu programu Excel, naciskając klawisze Alt+F11. Zamykamy plik invoice.txt bez zapisywania zmian. Arkusz MacroToImportInvoices.xlsm jest w dalszym ciągu otwarty.

Naciskamy klawisze Ctrl+I, by uruchomić zarejestrowane makro. Makro powinno zadziałać dobrze, jeśli prawidłowo wykonaliśmy poszczególne kroki procedury. Dane zostały zaimportowane, podsumowania dodane, formatowanie zastosowane, a szerokość kolumn powiększona. Wydaje się, że to doskonale rozwiązanie (rysunek 1.11).

|    | A           | B            | C               | D            | E             | F            | G             |
|----|-------------|--------------|-----------------|--------------|---------------|--------------|---------------|
| 1  | DataFaktury | NumerFaktury | NumerSprzedawcy | NumerKlienta | DochódProdukt | DochódUsługa | KosztProduktu |
| 2  | 05.06.2017  | 123813       | S82             | C8754        | 716100        | 12000        | 423986        |
| 3  | 05.06.2017  | 123814       |                 | C4894        | 224200        | 0            | 131243        |
| 4  | 05.06.2017  | 123815       | S43             | C7278        | 277000        | 0            | 139208        |
| 5  | 05.06.2017  | 123816       | S54             | C6425        | 746100        | 15000        | 350683        |
| 6  | 05.06.2017  | 123817       | S43             | C6291        | 928300        | 0            | 488988        |
| 7  | 05.06.2017  | 123818       | S43             | C1000        | 723200        | 0            | 383069        |
| 8  | 05.06.2017  | 123819       | S82             | C6025        | 982600        | 0            | 544025        |
| 9  | 05.06.2017  | 123820       | S17             | C8026        | 490100        | 45000        | 243808        |
| 10 | 05.06.2017  | 123821       | S43             | C4244        | 615800        | 0            | 300579        |
| 11 | Razem       | 123822       | S45             | C1007        | 5703400       | 72000        | 3005589       |

Rysunek 1.11 Makro formatuje dane w arkuszu

## Uruchomienie tego samego makra innego dnia generuje nieoczekiwane wyniki

Po przetestowaniu makra należy je zapisać, by można go było użyć następnego dnia. Następnego dnia po przyjsciu do pracy otrzymaliśmy nowy plik invoice.txt z systemu. Otwieramy makro, wciskamy Ctrl+I, by je uruchomić i... mamy katastrofę. W pliku z 5 czerwca jest 9 faktur, a 6 czerwca faktur jest 17. Zarejestrowane makro „ślepo” dodało wiersz z podsumowaniem w wierszu 11, ponieważ właśnie w tym wierszu umieściliśmy podsumowanie w momencie, gdy było ono rejestrowane (rysunek 1.12).

|    | A           | B            | C               | D            | E             | F            | G             |
|----|-------------|--------------|-----------------|--------------|---------------|--------------|---------------|
| 1  | DataFaktury | NumerFaktury | NumerSprzedawcy | NumerKlienta | DochódProdukt | DochódUsługa | KosztProduktu |
| 2  | 05.06.2017  | 123813       | S82             | C8754        | 716100        | 12000        | 423986        |
| 3  | 05.06.2017  | 123814       |                 | C4894        | 224200        | 0            | 131243        |
| 4  | 05.06.2017  | 123815       | S43             | C7278        | 277000        | 0            | 139208        |
| 5  | 05.06.2017  | 123816       | S54             | C6425        | 746100        | 15000        | 350683        |
| 6  | 05.06.2017  | 123817       | S43             | C6291        | 928300        | 0            | 488988        |
| 7  | 05.06.2017  | 123818       | S43             | C1000        | 723200        | 0            | 383069        |
| 8  | 05.06.2017  | 123819       | S82             | C6025        | 982600        | 0            | 544025        |
| 9  | 05.06.2017  | 123820       | S17             | C8026        | 490100        | 45000        | 243808        |
| 10 | 05.06.2017  | 123821       | S43             | C4244        | 615800        | 0            | 300579        |
| 11 | Razem       | 123822       | S45             | C1007        | 5703400       | 72000        | 3005589       |
| 12 | 05.06.2017  | 123823       | S87             | C1878        | 338100        | 0            | 165666        |
| 13 | 05.06.2017  | 123824       | S43             | C3068        | 567900        | 0            | 265775        |
| 14 | 05.06.2017  | 123825       | S43             | C7571        | 123456        | 0            | 55555         |
| 15 | 05.06.2017  | 123826       | S55             | C7181        | 37900         | 0            | 19811         |
| 16 | 05.06.2017  | 123827       | S43             | C7570        | 582700        | 0            | 292000        |
| 17 | 05.06.2017  | 123828       | S87             | C5302        | 495000        | 0            | 241504        |
| 18 | 05.06.2017  | 123828       | S87             | C5302        | 495000        | 0            | 241504        |

Rysunek 1.12 Makro miało dodawać podsumowanie na końcu pliku, ale rejestrator tak zarejestrował makro, że podsumowanie zawsze wyświetla się w wierszu 11.

## Możliwe rozwiązanie: wykorzystywanie odwołań względnych podczas rejestrowania

Domyślnie rejestrator makr rejestruje wszystkie działania jako odwołania *bezwzględne*. Jeśli podczas rejestrowania przechodzimy do wiersza 11, w momencie uruchamiania makro zawsze będzie przechodziło do wiersza 11. Bardzo rzadko takie działanie będzie odpowiednie w przypadku zmiennej liczby wierszy danych. Lepszym rozwiązaniem podczas rejestrowania jest wykorzystanie odwołań względnych.

W makrach zarejestrowanych w trybie odwołań bezwzględnych są zapisywane rzeczywiste adresy wskaźnika komórki (przykładowo A11). W makrach zarejestrowanych w trybie odwołań względnych zapisane są informacje o tym, że wskaźnik komórki powinien przemieścić się o określoną liczbę wierszy i kolumn w odniesieniu do bieżącego położenia. Na przykład, jeśli wskaźnik komórki znajduje się w komórce A1, kod `ActiveCell.Offset(16, 1).Select` przeniesie wskaźnik do komórki B17, czyli do komórki, która znajduje się o 16 wierszy w dół i jedną kolumnę w prawo.

Pomimo że rejestrowanie w trybie odwołań względnych jest odpowiednie dla większości sytuacji, czasami podczas rejestrowania może zachodzić jednak potrzeba użycia odwołania bezwzględnego. Ten przykład jest dobrą ilustracją takiej sytuacji: Po dodaniu wiersza podsumowania do zestawu danych trzeba powrócić do wiersza 1. Jeśli w trybie odwołań względnych po prostu klikniemy wiersz 1, program Excel zarejestruje operację zaznaczenia wiersza, który znajduje się 10 wierszy powyżej wiersza bieżącego. Taka operacja zadziała w przypadku pierwszego pliku `invoice.txt`, ale nie będzie działać w przypadku dłuższego lub krótszego pliku. Poniżej wymieniono dwie metody obejścia problemu:

- Wyłączenie rejestrowania w trybie odwołań względnych, kliknięcie wiersza 1, a następnie ponowne włączenie rejestrowania względnego.
- Zachowanie włączonego rejestrowania w trybie odwołań względnych. Wyświetlenie okna dialogowego *Przechodzenie do*, poprzez naciśnięcie klawisza F5. Wpisanie A1 i kliknięcie OK. Okno dialogowe *Przechodzenie do* rejestrowane jest w typowy sposób i następuje przejście do adresu bezwzględnego, nawet jeśli włączone jest rejestrowanie w trybie odwołań względnych. Odmiana tej metody wykorzystana została w poniższej analizie przypadku.

Spróbujmy ponownie przeprowadzić analizę tego samego przypadku – tym razem z wykorzystaniem odwołań względnych. Rozwiązanie będzie znacznie zbliżone do rozwiązania poprawnego.

### Studium przypadku: Rejestrowanie makra przy użyciu odwołań względnych

Spróbujmy zarejestrować makro jeszcze raz, tym razem z wykorzystaniem odwołań względnych.

*Uwaga:* jeśli stosowaliśmy pliki przykładów, należy wykonać następujące działania:

1. Zamknięcie pliku `invoice.txt` w programie Excel.
2. Zmiana nazwy pliku `invoice.txt` na `invoice2.txt`.

3. Zmiana nazwy pliku invoice1.txt na invoice.txt.
4. Powrót do skoroszytu MacroTolImportInvoices.xlsm.

Na karcie Deweloper wybieramy przycisk Użyj odwołań względnych, by zmienić tryb rejestrowania. Ustawienie to jest aktywne dopóki nie zostanie wyłączone lub do zamknięcia programu Excel.

W skoroszytcie MacroTolImportInvoices.xlsm rejestrujemy nowe makro poprzez wybranie polecenia Zarejestruj makro na karcie Deweloper. Nadajemy mu nazwę ImportFakturOdwWzgl i przypisujemy inny klawisz skrótu, np.: Ctrl+Shift+J.

Należy powtórzyć kroki 1 do 11 w poprzednim przykładzie analizy, by zaimportować plik, a następnie wykonać poniższe instrukcje:

1. Naciśnięcie Ctrl+strzałka w dół, by przejść do ostatniego wiersza danych.
2. Ponownie naciśnięcie klawisza strzałki w dół, by przejść do wiersza podsumowania.
3. Wpisanie słowa Razem.
4. Cztery razy naciśnięcie strzałki w prawo, by przejść do kolumny E wiersza podsumowania.
5. Przytrzymanie klawisza Shift naciskając jednocześnie dwa razy strzałkę w prawo, by zaznaczyć zakres E11:G11.
6. Kliknięcie przycisku Autosumowanie.
7. Naciśnięcie klawiszy Shift+spacja, by zaznaczyć cały wiersz. Naciśnięcie klawiszy Ctrl+B, by do wiersza zastosować formatowanie: pogrubienie.
8. Naciśnięcie F5, by wyświetlić okno dialogowe Przechodzenie do.
9. W oknie dialogowym Przechodzenie do, wpisz A1:G1 i kliknij przycisk OK. Nawet jeśli włączone jest rejestrowanie względne, każda nawigacja wykonywana za pośrednictwem okna dialogowego Przechodzenie do zostanie zarejestrowana jako odwołanie bezwzględne. Naciśnięcie klawiszy Ctrl+Home, by przejść do komórki A1.
10. Kliknięcie ikony Pogrubienie, by nagłówki były sformatowane za pomocą pogrubienia.
11. Naciśnięcie Ctrl+\*, by zaznaczyć wszystkie dane w bieżącym regionie.
12. Na karcie Narzędzia główne, wybranie opcji Autodopasowanie szerokości kolumn (w menu Formatuj).
13. Zatrzymanie rejestrowania.

Naciskamy klawisze Alt+F11, by przejść do edytora VB i przejrzeć kod. Nowe makro widoczne jest w Module1, poniżej poprzedniego makra.

Jeśli zamkniemy program Excel pomiędzy rejestrowaniem pierwszego i drugiego makra, dla nowo zarejestrowanego modułu program Excel wstawi nowy moduł nazywany Module2.

```

Sub ImportFakturOdWzgl()
'
' ImportFakturOdWzgl Makro
' Import pliku invoice.txt. Dodanie wiersza podsumowania.
' Formatowanie.
' Klawisz skrót: Ctrl+Shift+J
ChDir "E:\Excel2016_VBA_Makra\Pliki przykładów"
Workbooks.OpenText Filename:= _
    "E:\Excel2016_VBA_Makra\Pliki przykładów\invoice.txt", _
    Origin:=xlWindows, _
    StartRow:=1, DataType:=xlDelimited, _
    TextQualifier:=xlDoubleQuote, _
    ConsecutiveDelimiter:=False, Tab:=False, Semicolon:=False, _
    Comma:=True, Space:=False, Other:=False, _
    FieldInfo:=Array(Array(1, 3), Array(2, 1), Array(3, 1), _
    Array(4, 1), Array(5, 1), Array(6, 1), Array(7, 1)), _
    TrailingMinusNumbers:=True

Selection.End(xlDown).Select
ActiveCell.Offset(1, 0).Range("A1").Select
ActiveCell.FormulaR1C1 = "Razem"
ActiveCell.Offset(0, 4).Range("A1:C1").Select
Selection.FormulaR1C1 = "=SUM(R[-17]C:R[-1]C)"
ActiveCell.Rows("1:1").EntireRow.Select
ActiveCell.Activate
Selection.Font.Bold = True
Application.Goto Reference:="R1C1:R1C7"
Range("A1").Select
Selection.Font.Bold = True
Selection.CurrentRegion.Select
Selection.Columns.AutoFit
End Sub

```

W celu przetestowania makra zamykamy plik invoice.txt bez zapisywania zmian, a następnie uruchamiamy makro za pomocą klawiszy Ctrl+Shift+J. Wszystko wygląda poprawnie – uzyskaliśmy te same wyniki za pomocą makra utworzonego przez rejestrator.

Następny test polega na sprawdzeniu, czy program zadziała następnego dnia, w którym może być więcej wierszy. Jeśli wykorzystywaliśmy pliki przykładów, zamykamy plik invoice.txt w programie Excel. Zmieniamy nazwę pliku invoice.txt na invoice2.txt i zmieniamy nazwę pliku invoice4.txt na invoice.txt.

Otwieramy plik MacroToImportInvoices i uruchamiamy makro za pomocą skrótów Ctrl+Shift+J. Tym razem wszystko wygląda dobrze, a sumy umieszczone są na właściwych miejscach. Spójrzmy na rysunek 1.13. Czy jest tam coś niezwykłego?

Jeśli nie jesteśmy uważni, możemy wydrukować ten raport dla menedżera. I w takiej sytuacji możemy mieć kłopoty. Kiedy spojrzymy na komórkę E19, widzimy, że program Excel wstawił zielony trójkąt, by zwrócić uwagę na komórkę. Jeśli spróbowałibyśmy sprawdzić to w programie Excel 95 lub Excel 97, kiedy nie działały jeszcze funkcje znaczników inteligentnych (SmartTag), nic nie wskazywałoby, że coś nie jest w porządku.

Kiedy przeniesiemy wskaźnik położenia do komórki E19, w pobliżu komórki pojawi się wskaźnik ostrzeżenia, który informuje, że w formule nie powiodło się dołączenie

przylegających komórek. Przeglądając się paskowi formuły, zauważymy, że makro posu-  
mowało tylko wiersze od 10 do 18. Ani rejestrowanie z użyciem odwołań względnych, ani  
z użyciem odwołań bezwzględnych nie jest na tyle „sprytne”, by powielić logikę przycisku  
Autosumowanie.

Załóżmy, że w danym dniu mamy mniej rekordów faktur. Program Excel „nagrodziłby”  
nas nielogiczną formułą =SUM(E6:E1048574), jak ilustruje to rysunek 1.14. Ponieważ  
formuła ta byłaby w komórce E7, w pasku stanu pojawiłoby się ostrzeżenie o odwołaniu  
cyklicznym.

*Uwaga:* Aby wypróbować to samemu, zamykamy plik invoice.txt w programie Excel.  
Zmieniamy nazwę pliku invoice.txt na invoice2.txt i zmieniamy nazwę pliku invoice4.txt  
na invoice.txt.

| DataFaktury | NumerFaktury | NumerSprzedawcy | NumerKlienta | DochódProdukt | DochódUsługa | KosztProduktu |
|-------------|--------------|-----------------|--------------|---------------|--------------|---------------|
| 05.06.2017  | 123813       | S82             | C8754        | 716100        | 12000        | 423986        |
| 05.06.2017  | 123814       |                 | C4894        | 224200        | 0            | 131243        |
| 05.06.2017  | 123815       | S43             | C7278        | 277000        | 0            | 139208        |
| 05.06.2017  | 123816       | S54             | C6425        | 746100        | 15000        | 350683        |
| 05.06.2017  | 123817       | S43             | C6291        | 928300        | 0            | 488988        |
| 05.06.2017  | 123818       | S43             | C1000        | 723200        | 0            | 383069        |
| 05.06.2017  | 123819       | S82             | C6025        | 982600        | 0            | 544025        |
| 05.06.2017  | 123820       | S17             | C8026        | 490100        | 45000        | 243808        |
| 05.06.2017  | 123821       | S43             | C4244        | 615800        | 0            | 300579        |
| 05.06.2017  | 123822       | S45             | C1007        | 271300        | 0            | 153253        |
| 05.06.2017  | 123823       | S87             | C1878        | 338100        | 0            | 165666        |
| 05.06.2017  | 123824       | S43             | C3068        | 567900        | 0            | 265775        |
| 05.06.2017  | 123825       | S43             | C7571        | 123456        | 0            | 55555         |
| 05.06.2017  | 123826       | S55             | C7181        | 37900         | 0            | 19811         |
| 05.06.2017  | 123827       | S43             | C7570        | 582700        | 0            | 292000        |
| 05.06.2017  | 123828       | S87             | C5302        | 495000        | 0            | 241504        |
| 05.06.2017  | 123828       | S87             | C5302        | 495000        | 0            | 241504        |
|             |              |                 |              | 3527156       | 0            | 1735647       |

**Rysunek 1.13** Rezultat uzyskany po uruchomieniu makra w trybie odwołań względnych.

| DataFaktury | NumerFaktury | NumerSprzedawcy | NumerKlienta | DochódProdukt | DochódUsługa | KosztProduktu |
|-------------|--------------|-----------------|--------------|---------------|--------------|---------------|
| 08.06.2017  | 123850       |                 | C1654        | 161000        | 0            | 90761         |
| 08.06.2017  | 123851       |                 | C6460        | 275500        | 10000        | 146341        |
| 08.06.2017  | 123852       |                 | C5143        | 925400        | 0            | 473515        |
| 08.06.2017  | 123853       |                 | C7868        | 148200        | 0            | 75700         |
| 08.06.2017  | 123854       |                 | C3310        | 890200        | 0            | 468333        |
| Razem       |              |                 |              | 0             | 0            | 0             |

**Rysunek 1.14** Wynik działania makro w trybie odwołań względnych przy mniejszej liczbie rekordów faktur.

Jeśli ktoś spróbował używać rejestratora makr, prawdopodobnie napotkał problemy po-  
dobne do tych, które opisano w analizie ostatnich dwóch przypadków. Chociaż jest to fru-  
strujące, pozytywne jest to, że rejestrator makr pozwala zrealizować 95% zadania związa-  
nego z utworzeniem poprawnego makra.

Naszym zadaniem jest rozpoznanie miejsc, w których działanie rejestratora makr może  
się nie powieść, a następnie zmodyfikowanie kodu VBA w celu poprawienia jednego czy  
kilku wierszy i uzyskania poprawnie działającego makra. Korzystając z własnej inwencji,  
możemy tworzyć ciekawe makra, które przyspieszą nasze codzienne działania.

Jeśli ktoś jest podobny do mnie, to pewnie teraz w duchu przeklina firmę Microsoft.  
Zmarnowaliśmy sporo czasu w ciągu ostatnich kilku dni i żadne makro nie zadziałało.  
Na domiar złego, z tego typu sytuacjami bez trudu radził sobie rejestrator makr Lotus

1-2-3 powstały w 1983 roku. 33 lata temu Mitch Kapor rozwiązał problem, z którym firma Microsoft nie może sobie poradzić dzisiaj.

Czy wiecie, że do wersji Excel 97 program Microsoft Excel „potajemnie” obsługiwał makra Lotusa. Fakt ten odkryłem zaraz po tym, jak firma Microsoft zaprzestała obsługi programu Excel 97. Wiele firm przeszło wówczas na Excel XP, który już nie obsługiwał makr Lotus 1-2-3. Sporo firm zleciło nam wtedy zadanie konwersji starych makr Lotusa 1-2-3 na Excel VBA. Interesujące jest to, że w wersjach Excel 5, Excel 95 i Excel 97 firma Microsoft oferowała interpreter, który potrafił poprawnie obsłużyć makra programu Lotus, które bezbłędnie rozwiązują ten problem, podczas gdy ich własny rejestrator tego nie robił (i nadal nie potrafi).

## Podczas rejestrowania nigdy nie używaj przycisku Autosumowanie lub Szybka analiza

Opisany problem dotyczący rejestrowania funkcji Autosumowanie można jednak rozwiązać, przy czym należy zapamiętać, że rejestrator makr nigdy poprawnie nie zarejestruje działania przycisku Autosumowanie.

Jeśli rozpoczniemy działanie w komórce E99 i klikniemy przycisk Autosumowanie, program Excel zacznie skanowanie od komórki E98 w górę do czasu, aż odnajdzie komórkę tekstową, pustą komórkę lub formułę. Następnie zaproponuje formułę, która sumuje wszystkie komórki pomiędzy komórką bieżącą a tą znalezioną.

Rejestrator makr rejestruje jednak określony wynik tego wyszukiwania w momencie, w którym zarejestrowano makro. Zamiast zarejestrować coś w stylu „wykonaj zwykłą logikę przycisku Autosumowanie”, rejestrator makr wstawia pojedynczy wiersz kodu sumujący poprzednie 98 komórek.

W programie Excel 2013 wprowadzono funkcję Szybka analiza. Zaznaczamy zakres E2:G99; otwieramy ikonę Szybka analiza, która wyświetlana jest poniżej prostokątnego zaznaczenia z prawej jego strony; wybieramy opcję Suma; i uzyskujemy poprawne sumy w wierszu 100. Rejestrator makr na sztywno zakoduje formuły, by zawsze wyświetlane były w wierszu 100 i zawsze sumowały wiersze od 2 do 99.

Dość dziwnym obejściem tego problemu jest wpisanie funkcji SUMA, która wykorzystuje połączenie odwołań względnych i bezwzględnych. Jeśli wpiszemy =SUMA(E\$2:E10) w czasie działania rejestratora makr, program Excel poprawnie doda kod, który zawsze będzie sumował komórki, poczynwszy od ustalonego wiersza 2 w dół do względnego odwołania, które jest bezpośrednio nad komórką bieżącą.

Poniżej zaprezentowano uzyskany kod:

```
Sub FormatInvoice3()
Sub FormatInvoice3()
' Makro FormatInvoice3
' Importowanie. Podsumowanie. Formatowanie.
' Skrót klawiszowy: Ctrl+K
Workbooks.OpenText Filename:="C:\Data\invoice.txt", _
Origin:=437, StartRow:=1, DataType:=xlDelimited, _
TextQualifier:=xlDoubleQuote, ConsecutiveDelimiter:=False, _
Tab:=False, Semicolon:=False, Comma:=True, Space:=False, _
Other:=False, FieldInfo:=Array(Array(1, 3), Array(2, 1), _
```

```

        Array(3, 1), Array(4, 1), Array(5, 1), Array(6, 1), _
        Array(7, 1)), TrailingMinusNumbers:=True
    Selection.End(xlDown).Select
    ActiveCell.Offset(1, 0).Range("A1").Select
    ActiveCell.FormulaR1C1 = "Total"
    ActiveCell.Offset(0, 4).Range("A1").Select
    Selection.FormulaR1C1 = "=SUM(R2C:R[-1]C)"
    Selection.AutoFill Destination:=ActiveCell.Range("A1:C1"), _
        Type:=xlFillDefault
    ActiveCell.Range("A1:C1").Select
    ActiveCell.Rows("1:1").EntireRow.Select
    ActiveCell.Activate
    Selection.Font.Bold = True
    Application.Goto Reference:="R1C1:R1C7"
    Selection.Font.Bold = True
    Selection.CurrentRegion.Select
    Selection.Columns.AutoFit
End Sub

```

To trzecie makro działa poprawnie dla zbioru danych dowolnego rozmiaru.

## Cztery wskazówki dotyczące używania rejestratora makr

Bardzo rzadko zdarza się sytuacja, by 100% z zarejestrowanych makr zadziałało. Będziemy jednak znacznie bliżej tego celu, jeśli zastosujemy się do czterech wskazówek prezentowanych poniżej.

### Wskazówka 1: Włączaj ustawienie Użyj odwołań względnych

Firma Microsoft powinna nadać temu ustawieniu status ustawienia domyślnego. Włącz ustawienie i nie wyłączaj go podczas rejestrowania makr.

### Wskazówka 2: Używaj specjalnych klawiszy nawigacyjnych do przechodzenia na koniec zbioru danych

Jeśli jesteś na początku zbioru danych i chcesz przejść do ostatniej komórki z danymi, naciśnij klawisze Ctrl+strzałka w dół lub naciśnij klawisz End, a następnie strzałkę w dół.

Podobnie, aby przejść do ostatniej kolumny w bieżącym wierszu zbioru danych, naciśnij klawisze Ctrl+strzałka w prawo lub naciśnij klawisz End, a następnie strzałkę w prawo.

Dzięki użyciu tych klawiszy nawigacyjnych można przechodzić na koniec zbioru danych niezależnie od tego, z ilu wierszy lub kolumn składa się zbiór danych określonego dnia.

Używaj klawiszy Ctrl+\*, by zaznaczać bieżący region wokół komórki aktywnej. Zakładając, że dane nie zawierają pustych wierszy lub kolumn, ta kombinacja klawiszy powoduje zaznaczenie całego zestawu danych.

### Wskazówka 3: Podczas rejestrowania makra nigdy nie używaj przycisku Autosumowanie

Rejestrator makr nie potrafi zarejestrować „istoty” działania przycisku Autosumowanie. Zamiast tego koduje „na sztywno” formułę, wynikającą z naciśnięcia tego przycisku. Formuła ta nie zadziała poprawnie, jeśli zbiór danych będzie zawierał mniej lub więcej rekordów.

Zamiast tego należy wpisać formułę z pojedynczym znakiem dolara, na przykład `=SUMA(E$2:E10)`. W takim przypadku, rejestrator makr zarejestruje pierwsze odwołanie `E$2` jako ustalone i rozpocznie określać zakres sumowania bezpośrednio pod nagłówkami w pierwszym wierszu. Zakładając, że bieżącą komórką jest `E11`, rejestrator makr rozpozna `E10` jako odwołanie względne wskazujące bezpośrednio powyżej bieżącej komórki.

#### **Wskazówka 4: Próbuje rejestrować makra innymi metodami, jeśli zastosowana metoda nie działa**

W programie Excel najczęściej istnieje wiele sposobów wykonywania tych samych zadań. Jeśli napotkamy błędny kod generowany przez jeden sposób, warto spróbować innego. Ponieważ w zespole menedżerów projektów programu Excel jest 16 osób, prawdopodobnie każda metoda była programowana przez inną grupę. W jednej z analiz przypadków w tym rozdziale, jedno zadanie wiązało się z zastosowaniem do wszystkich komórek funkcji Autodopasowanie szerokości kolumn. Niektóre osoby mogły nacisnąć klawisze `Ctrl+A`, by zaznaczyć wszystkie komórki, a inne mogły nacisnąć klawisze `Ctrl+*`. Począwszy od wersji Excel 2007, nie działa kod generowany przez naciśnięcie klawiszy `Ctrl+A` przy włączonym trybie odwołań względnych. Kod związany z sekwencją `Ctrl+*` jest bardzo stary i nadal działa we wszystkich przypadkach.

## **Następne kroki**

W rozdziale 2 „Skoro nazywa się BASIC, dlaczego nie wygląda znajomo?”, przeanalizujemy trzy makra, które zarejestrowaliśmy w tym rozdziale i postaramy się je zrozumieć. Kiedy już nauczymy się czytać kod VBA, naturalnym krokiem będzie poprawienie zarejestrowanego kodu lub napisanie poprawnego od początku. Warto przeczytać kolejny rozdział – wkrótce przekonamy się, że VBA to dobre rozwiązanie i nauczymy się pisać dobrze działający kod.

# Skoro nazywa się BASIC, dlaczego nie wygląda znajomo?

## 2

### Nie rozumiem tego kodu!

Jak już nadmieniono w rozdziale 1 „Zwiększanie możliwości programu Excel za pomocą języka VBA”, jeśli ktoś zapoznał się z językiem proceduralnym, takim jak BASIC czy COBOL, może być trochę zdezorientowany przy przeglądaniu kodu w języku VBA. Wprawdzie skrót VBA rozwija się jako *Visual Basic for Applications*, jest to jednak obiektowa wersja języka BASIC. Poniżej zamieszczono przykład kodu w języku VBA:

```
Selection.End(xlDown).Select
Range("A11").Select
ActiveCell.FormulaR1C1 = "Razem"
Range("E11").Select
Selection.FormulaR1C1 = _
    "=SUM(R[-9]C:R[-1]C)"
Selection.AutoFill _
    Destination:=Range("E11:G11"), _
    Type:=xlFillDefault
```

Prawdopodobnie kod ten nie ma żadnego sensu dla osób, które poznały jedynie języki proceduralne. Niestety, najczęściej pierwszym wprowadzeniem do programowania w szkole (zakładając, że ktoś ma ponad 40 lat) był język proceduralny. A to przykład kodu napisanego w języku BASIC:

```
For x = 1 to 10
    Print Rpt$(" ",x);
    Print "*"
Next x
```

### W tym rozdziale:

|                                                                                    |    |
|------------------------------------------------------------------------------------|----|
| Nie rozumiem tego kodu! .....                                                      | 35 |
| „Części mowy” języka VBA .....                                                     | 36 |
| Język VBA nie jest bardzo trudny .....                                             | 40 |
| Analiza kodu zarejestrowanego makra:<br>korzystanie z edytora VB i tematów pomocy. | 41 |
| Stosowanie narzędzi debugowania<br>do analizy zarejestrowanego kodu. ....          | 47 |
| Narzędzie Object Browser:<br>odwołanie ostateczne .....                            | 54 |
| Siedem wskazówek poprawiania<br>zarejestrowanego kodu .....                        | 55 |
| Następne kroki .....                                                               | 62 |

Jeśli uruchomimy ten kod, na ekranie uzyskamy piramidę gwiazdek:

```

*
 *
  *
   *
    *
     *
      *

```

Jeśli ktoś poznał programowanie proceduralne, prawdopodobnie przygląda się wcześniej-szemu kodowi i zastanawia się, co tu się dzieje, ponieważ języki proceduralne bardziej przypominają język angielski niż jest to w przypadku języków obiektowych. Instrukcja `Print "Hello World"` jest zgodna z formatem czasownik-obiekt, który jest zgodny z ogólną zasadą mówienia. Oderwijmy się na chwilę od programowania i przyjrzyjmy się konkretnemu przykładowi.

## „Części mowy” języka VBA

Jeśli chcielibyśmy napisać kod dla instrukcji gry w piłkę za pomocą języka BASIC, instrukcja kopnięcia piłki mogłaby wyglądać następująco:

```
"Kick the Ball"
```

To tak samo, jak mówimy! I coś takiego ma sens. Mamy czasownik kopnij (`kick`), a następnie rzeczownik piłka (`ball`). Kod języka BASIC w poprzednim podrozdziale miał czasownik drukuj (`Print`) i rzeczownik gwiazdka (\*). Życie jest piękne.

I tu mamy problem: język VBA nie działa w ten sposób. W rzeczywistości nie działa w ten sposób żaden język obiektowy. W języku obiektowym, obiekty (rzeczowniki) są najważniejsze, i stąd nazwa tych języków: obiektowe. Jeśli zamierzaliśmy napisać kod instrukcji do gry w piłkę za pomocą VBA, podstawowa struktura mogłaby mieć taką składnię:

```
Ball.Kick
```

Mamy rzeczownik piłka (`Ball`), który jest pierwszy. W języku VBA jest to *obiekt*. Następnie mamy czasownik kopnij (`Kick`). W języku VBA jest to *metoda*.

Podstawowa struktura języka VBA to seria wierszy kodu o poniższej składni:

```
Obiekt.Metoda
```

Nie trzeba mówić, że nie jest to język angielski. Jeśli w szkole posługiwaliśmy się romantycznym językiem, pamiętamy, że te języki korzystały z konstrukcji „rzeczownik-przymiotnik”, ale nikt nie używał składni „rzeczownik-czasownik”, prosząc kogoś, by coś zrobił:

```

woda.pić
jedzenie.jeść
dziewczyna.całować

```

I to jest przyczyna, dlaczego język VBA jest niezrozumiały dla osób, które poprzednio uczestniczyły w klasach programowania proceduralnego.

Rozwińmy analogię trochę bardziej. Wyobraźmy sobie, że chodzimy po trawiastym boisku, a przed nami widzimy pięć piłek. Jest to piłka do piłki nożnej (soccer), koszykówki, baseballa, kula do gry w kręgle i piłka do tenisa. Chcemy poinstruować dzieciaka w naszym zespole piłki nożnej, by kopnął piłkę (do piłki nożnej) „kick the soccer ball”.

Jeśli powiemy kopnij piłkę (czyli `ball.kick`), nie będziemy mieli pewności, która z pięciu piłek zostanie kopnięta przez dziecko. Być może kopnie piłkę, która jest najbliżej, co może być problematyczne, jeśli będzie stał naprzeciw kuli do kręgli.

W przypadku większości rzeczowników, czyli obiektów w języku VBA, istnieje zbiór tego obiektu. Pomyślmy o programie Excel. Jeśli mamy jeden wiersz (row), możemy mieć kilka wierszy (rows). Jeśli mamy jedną komórkę (cell), możemy mieć grupę komórek (cells). Jeśli mamy jeden arkusz (worksheet), również możemy mieć wiele arkuszy (worksheets).

W języku angielskim, jedyna różnica pomiędzy obiektem a zbiorem tych obiektów polega na dodaniu litery *s* do nazwy obiektu:

Row zmienia się na Rows  
Cell zmienia się na Cells  
Ball zmienia się na Balls

Jeśli odwołujemy się do czegoś, co jest zbiorem (czy inaczej kolekcją), musimy poinformować język programowania, do którego elementu się odnosimy. Istnieje kilka metod realizowania tego zadania. Możemy odnosić się do elementu za pomocą liczb. Przykładowo, jeśli piłka nożna jest drugą w kolejności piłką, możemy „powiedzieć”:

```
Balls(2).Kick
```

Metoda działa dobrze, ale może to być niebezpieczne dla programu. Na przykład, to dobrze zadziała we wtorek, ale jeśli przyjdziemy na boisko w środę i ktoś zmieni kolejność piłek, polecenie `Balls(2).Kick` może mieć bolesne konsekwencje.

Znacznie bezpieczniejsza metoda polega na używaniu nazw obiektów w kolekcji, która korzysta z takiej składni:

```
Balls("Soccer").Kick
```

Za pomocą tej metody zawsze wiemy, że kopnięta będzie piłka nożna.

Jak dotąd jest dobrze. Wiemy, że piłka będzie kopnięta i wiemy, że kopnięta będzie piłka nożna. Dla większości czasowników, czyli metod w języku Excel VBA, istnieją parametry, określające, jak ma być wykonane to działanie. Parametry te spełniają rolę przysłówków. Przykładowo, może chcieć, by piłka nożna była kopnięta silnie i w lewą stronę. W tym przypadku, metoda będzie miała kilka parametrów, które informują, w jaki sposób program powinien wykonać metodę (w poniższym przykładzie `Direction` oznacza kierunek, `Force` siłę, a `Hard` mocno):

```
Balls("Soccer").Kick Direction:=Left, Force:=Hard
```

W kodzie VBA, połączenie dwukropka i znaku równości (`:=`) wskazuje, że patrzymy na parametry, określające, jak powinna być wykonywana czynność (czyli czasownik).

Czasami metoda będzie miała listę 10 parametrów, a niektóre z nich są opcjonalne. Przykładowo, jeśli metoda Kick posiada parametr Elevation (wysokość), otrzymamy taki wiersz kodu:

```
Balls("Soccer").Kick Direction:=Left, Force:=Hard, Elevation:=High
```

W tym miejscu mamy pewne zagmatwanie: Każda metoda posiada domyślną kolejność swoich parametrów. Jeśli nie jesteśmy skrupulatnym programistą i znamy kolejność parametrów, możemy opuścić nazwy parametrów, jak w poniższym przykładzie, który jest równoważny poprzedniej linii kodu:

```
Balls("Soccer").Kick Left, Hard, High
```

Sposób ten nie ułatwia zrozumienia kodu. Bez znaków := nie jest oczywiste, że są to parametry. Jeśli nie znamy kolejności parametrów, możemy nie zrozumieć tego zapisu. Dość prosto zrozumieć zapis w przypadku Left, Hard i High, ale jeśli są to następujące parametry:

```
ActiveSheet.Shapes.AddShape type:=1, Left:=10, Top:=20, _  
Width:=100, Height:=200
```

poniższy „skrótowy” zapis nie będzie zrozumiałą:

```
ActiveSheet.Shapes.AddShape 1, 10, 20, 100, 200
```

Poprzedni wiersz kodu jest poprawny. O ile jednak nie znamy domyślnej kolejności parametrów tej metody Add, czyli Type, Left, Top, Width i Height, kod ten nie ma sensu. Domyślna kolejność parametrów danej metody to kolejność pokazywana w temacie Pomocy opisującym tę metodę.

Żeby jeszcze bardziej skomplikować życie, możemy rozpocząć specyfikowanie parametrów w ich porządku domyślnym bez nazywania parametrów, a następnie możemy rozpocząć nazywanie parametrów, jeśli akurat użyjemy parametru, który nie spełnia domyślnej kolejności. Przykładowo, jeśli chcemy kopnąć piłkę w lewo (left) i wysoko (high), ale nie ma znaczenia dla nas siła kopnięcia (czyli chcemy użyć wartości domyślnej dla parametru siły), poniższe dwie instrukcje mają identyczne działanie:

```
Balls("Soccer").Kick Direction:=Left, Elevation:=High  
Balls("Soccer").Kick Left, Elevation:=High
```

Warto jednak pamiętać, że jeśli rozpoczniemy nazywanie parametrów, muszą one być również nazywane w pozostałej części wiersza.

Niektóre metody działają na swój własny sposób. Aby zasymulować naciśnięcie klawisza F9, używamy tego kodu:

```
Application.Calculate
```

Inne metody wykonują działanie i coś tworzą. Na przykład, za pomocą poniższego kodu możemy dodać arkusz:

```
Worksheets.Add Before:=Worksheets(1)
```

Ponieważ jednak metoda `Worksheets.Add` tworzy nowy obiekt, możemy do zmiennej przypisać wynik działania tej metody. W tym przypadku, musimy ująć parametry nawiasami:

```
Set MyWorksheet = Worksheets.Add(Before:=Worksheets(1))
```

Potrzebne jest jeszcze poznanie ostatniego fragmentu gramatyki: przymiotników. Podobnie jak przymiotnik opisuje rzeczownik, *właściwości* opisują obiekt. Ponieważ jesteśmy fanami programu Excel, przejdziemy z analogii piłkarskich do analogii z programem Excel. Istnieje obiekt do opisywania komórki aktywnej i na szczęście ma intuicyjną nazwę:

```
ActiveCell
```

Załóżmy, że chcemy zmienić kolor komórki aktywnej na czerwony. Istnieje właściwość nazwana `Interior.Color` dla komórki, która używa złożonego kodu. Możemy jednak zmienić kolor komórki na czerwony za pomocą poniższego kodu:

```
ActiveCell.Interior.Color = 255
```

Widzimy, jak może to być mylące. Mamy poprzednią konstrukcję *rzeczownik-kropka-coś*, ale tym razem jest to raczej *Obiekt.Właściwość*, a nie *Obiekt.Metoda*. Sposób rozróżniania tego jest dość subtelny: Przed znakiem równości brak jest dwukropka. Prawie zawsze wartość właściwości jest ustawiana tak samo jak coś innego lub inaczej mówiąc wartości właściwości przypisywana jest inna wartość (stąd znak równości).

Aby kolor tej komórki był taki sam, jak dla komórki A1, możemy „powiedzieć”:

```
ActiveCell.Interior.Color = Range("A1").Interior.Color
```

`Interior.Color` to właściwość. Poprzez zmianę wartości właściwości zmieniamy wygląd rzeczy. Trochę to dziwne: Zmieniając przymiotnik, tak naprawdę modyfikujemy komórkę. Ludzie powiedzieliby „Pokoloruj komórkę na czerwono”, natomiast w języku VBA zdanie takie ma postać:

```
ActiveCell.Interior.Color = 255
```

W tabeli 2.1 zamieszczono podsumowanie „części mowy” języka VBA.

**Tabela 2.1** Części języka programowania VBA

| Składnik VBA | Analogia                     | Uwagi                                                                                                                                                        |
|--------------|------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Obiekt       | Rzeczownik                   | Przykład: komórka lub arkusz.                                                                                                                                |
| Kolekcja     | Rzeczownik w liczbie mnogiej | Zazwyczaj określa, który z obiektów: <code>Sheets(1)</code> .                                                                                                |
| Metoda       | Czasownik                    | Występuje w postaci: <code>Obiekt.Metoda</code> .                                                                                                            |
| Parametr     | Przysłówek                   | Lista parametrów po metodzie. Nazwa parametru oddzielana jest od wartości za pomocą znaków <code>:</code> .                                                  |
| Właściwość   | Przymiotnik                  | Można ustawić właściwość (na przykład <code>activecell.height=10</code> ) lub zapisać wartość właściwości (na przykład <code>x = activecell.height</code> ). |

## Język VBA nie jest bardzo trudny

Informacja o tym, czy mamy do czynienia z właściwościami czy z metodami ułatwia określenie prawidłowej składni kodu. Nie trzeba przejmować się, że w tym momencie wygląda to trochę zagmatwanie. Kiedy piszemy kod VBA od początku, wystarczy wiedzieć, czy proces zmieniający kolor komórki na żółty wymaga czasownika, czy przymiotnika. Innymi słowy, czy jest to metoda, czy właściwość?

I to jest moment, w którym rejestrator makr jest szczególnie przydatny. Jeśli nie wiem, jak zakodować jakieś działanie, rejestrujemy krótkie makro, przyglądamy się zarejestrowanemu kodowi i domyślamy się jego działania.

### Pliki pomocy VBA: Stosowanie klawisza F1 do wyszukiwania potrzebnych informacji

Moduł pomocy programu Excel VBA to zdumiewająca funkcja, która zakłada, że jesteśmy połączeni z Internetem. Jeśli planujemy napisanie makr VBA, bezwzględnie *musimy* mieć dostęp do zainstalowanych plików pomocy VBA. Poniższa procedura opisuje, jak prosto jest uzyskać pomoc w programie VBA:

1. Otwórz program Excel i przejdź do edytora VB, naciskając klawisze Alt+F11. W menu Insert wybierz opcję Module.
2. Wpisz poniższe trzy wiersze kodu:

```
Sub Test()  
MsgBox "Hello World!"  
End Sub
```

3. Kliknij wewnątrz słowa MsgBox.
4. Mając kursor w słowie MsgBox, naciśnij klawisz F1. Jeśli działa połączenie z Internetem, wyświetlony zostanie temat pomocy dotyczący funkcji MsgBox.

### Korzystanie z pomocy

Jeśli zażądamy informacji pomocy na temat funkcji lub metody, uzyskany w odpowiedzi temat obejmie różne dostępne argumenty. Przechodząc na koniec tematu pomocy, zobaczymy bardzo przydatne źródło informacji: przykłady kodu poniżej nagłówka Example (rysunek 2.1).

Mamy możliwość zaznaczenia kodu, skopiowania go do Schowka (klikając Ctrl+C), a następnie wklejenia kodu w module (naciskając Ctrl+V).

Po zarejestrowaniu makra, jeśli są tam obiekty lub metody, co do działania których nie mamy pewności, możemy uzyskać pomoc wstawiając kursor w dowolnym słowie kluczowym i naciskając klawisz F1.

**Example**

This example uses the **MsgBox** function to display a critical-error message in a dialog box with Yes and No buttons. The No button is specified as the default response. The value returned by the **MsgBox** function depends on the button chosen by the user. This example assumes that **DEMO.HLP** is a Help file that contains a topic with a Help context number equal to is a Help file that contains a topic with a Help context number equal to **1000**.

**VBA** Copy

```
Dim Msg, Style, Title, Help, Ctxt, Response, MyString
Msg = "Do you want to continue ?" ' Define message.
Style = vbYesNo + vbCritical + vbDefaultButton2 ' Define buttons.
Title = "MsgBox Demonstration" ' Define title.
Help = "DEMO.HLP" ' Define Help file.
Ctxt = 1000 ' Define topic
' context.
' Display message.
Response = MsgBox(Msg, Style, Title, Help, Ctxt)
If Response = vbYes Then ' User chose Yes.
    MyString = "Yes" ' Perform some action.
Else ' User chose No.
    MyString = "No" ' Perform some action.
End If
```

Rysunek 2.1 W większości tematów pomocy zamieszczone są przykłady

## Analiza kodu zarejestrowanego makra: korzystanie z edytora VB i tematów pomocy

Przyjrzyjmy się kodowi zarejestrowanemu w rozdziale 1, by sprawdzić, czy jest on dla nas bardziej zrozumiały teraz, kiedy już mamy pewne informacje na temat obiektów, właściwości i metod. Zobaczymy również, że można poprawiać błędy powstałe w rejestratorze makr.

Na rysunku 2.2 przedstawiono pierwszy kod zarejestrowany przez program Excel w przykładzie z rozdziału 1.

(General) ImportInvoices

```
Sub ImportInvoices ()
'
' Makro ImportInvoices
' Import pliku invoice.txt, dodanie wiersza podsumowania. Formatowanie.
'
' Klawisz skrót: Ctrl+I
'
    ThisPath = ThisWorkbook.Path & Application.PathSeparator
    Workbooks.OpenText Filename:=ThisPath & "invoice.txt", _
        Origin:=xlWindows, StartRow:=1, DataType:=xlDelimited, TextQualifier:= _
        xlDoubleQuote, ConsecutiveDelimiter:=False, Tab:=False, Semicolon:=False, _
        Comma:=True, Space:=False, Other:=False, FieldInfo:=Array(Array(1, 3), _
        Array(2, 1), Array(3, 1), Array(4, 1), Array(5, 1), Array(6, 1), Array(7, 1)), _
        TrailingMinusNumbers:=True
    Selection.End(xlDown).Select
    Range("A1").Select
    ActiveCell.FormulaR1C1 = "Razem"
    Range("E1").Select
    Selection.FormulaR1C1 = "=SUM(R[-9]C:R[-1]C)"
    Selection.AutoFill Destination:=Range("E1:G11"), Type:=xlFillDefault
    Range("E1:G11").Select
    Rows("1:1").Select
    Selection.Font.Bold = True
    Rows("11:11").Select
    Selection.Font.Bold = True
    Cells.Select
    Range("A1").Activate
    Selection.Columns.AutoFit
End Sub
```

Rysunek 2.2 Kod zarejestrowany w przykładzie w rozdziale 1

Teraz, po poznaniu koncepcji `Rzeczownik.Czasownik` lub `Obiekt.Metoda`, przeanalizujemy pierwszy wiersz kodu: `Workbooks.OpenText`. W tym przypadku, słowo `Workbooks` jest obiektem, a `OpenText` metodą. Możemy kliknąć kursorem wewnątrz słowa `OpenText` i nacisnąć klawisz F1, by uzyskać wyjaśnienie na temat metody `OpenText` (rysunek 2.3).

W pliku pomocy znajdziemy potwierdzenie, że `OpenText` to metoda, czyli słowo opisujące działanie. Domyślna kolejność wszystkich argumentów używanych w połączeniu z metodą `OpenText` widoczna jest na szarym polu. Warto zwrócić uwagę, że tylko jeden argument jest wymagany: `Filename`. Wszystkie pozostałe to argumenty opcjonalne.

## Parametry opcjonalne

W pliku pomocy znajdziemy informacje o tym, co stanie się, jeśli pominiemy parametr opcjonalny. W przypadku parametru `StartRow` (wiersz początkowy), przyjmowana jest wartość domyślna 1. Jeśli pominiemy parametr `StartRow`, program Excel rozpocznie importowanie od wiersza 1. Jest to całkiem bezpieczny sposób.

Teraz zobaczymy, jakie uwagi znajdziemy w pliku pomocy na temat parametru `Origin`. Jeśli ten argument zostanie pominięty, odziedziczymy wartość, jaka była zastosowana dla parametru `Origin`, kiedy ktoś użył tej funkcji w programie Excel na tym komputerze. I to jest przepis na katastrofę. Przykładowo, nasz kod może działać przez większość czasu, ale niespodziewanie, kiedy ktoś zaimportował plik arabski, program Excel zapamiętał ustawienie i następnie założył, że to jest wartość potrzebna w naszym makro, jeśli w kodzie nie wyspecyfikujemy wprost tego parametru.

### Parameters

| Name                        | Required/Optional | Data Type              | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|-----------------------------|-------------------|------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>Filename</i>             | Required          | <b>String</b>          | Specifies the file name of the text file to be opened and parsed.                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| <i>Origin</i>               | Optional          | <b>Variant</b>         | Specifies the origin of the text file. Can be one of the following <b>XlPlatform</b> constants: <b>xlMacintosh</b> , <b>xlWindows</b> , or <b>xlMSDOS</b> . Additionally, this could be an integer representing the code page number of the desired code page. For example, "1256" would specify that the encoding of the source text file is Arabic (Windows). If this argument is omitted, the method uses the current setting of the <b>File Origin</b> option in the <b>Text Import Wizard</b> . |
| <i>StartRow</i>             | Optional          | <b>Variant</b>         | The row number at which to start parsing text. The default value is 1.                                                                                                                                                                                                                                                                                                                                                                                                                               |
| <i>DataType</i>             | Optional          | <b>Variant</b>         | Specifies the column format of the data in the file. Can be one of the following <b>XLTextParsingType</b> constants: <b>xlDelimited</b> or <b>xlFixedWidth</b> . If this argument is not specified, Microsoft Excel attempts to determine the column format when it opens the file.                                                                                                                                                                                                                  |
| <i>TextQualifier</i>        | Optional          | <b>XLTextQualifier</b> | Specifies the text qualifier.                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| <i>ConsecutiveDelimiter</i> | Optional          | <b>Variant</b>         | <b>True</b> to have consecutive delimiters considered one delimiter. The default is <b>False</b> .                                                                                                                                                                                                                                                                                                                                                                                                   |

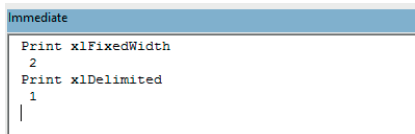
Rysunek 2.3 Fragment tematu pomocy opisującego metodę `OpenText`

## Zdefiniowane stałe

Przyjrzyjmy się plikowi pomocy dla parametru `DataType` (rysunek 2.3), który informuje, że typem danych może to być jedna z tych stałych: `xlDelimited` lub `xlFixedWidth`. Stałe te są ważnymi stałymi `xlTextParsingType`, które zostały wcześniej zdefiniowane w programie Excel VBA. W edytorze VB, naciskamy `Ctrl+G`, by wywołać oknie Immediate. W oknie Immediate, wpisujemy poniższy wiersz kodu i naciskamy `Enter`:

```
Print xlFixedWidth
```

Odpowiedź wyświetlona zostaje w oknie Immediate. Parametr `xlFixedWidth` jest równy wartości 2 (rysunek 2.4). W oknie Immediate wpisujemy `Print xlDelimited`, co jest tym samym, co wpisanie 1. Firma Microsoft prawidłowo zakłada, że łatwiej jest komuś przeczytać kod, który stosuje pojęcie przypominające angielskie wyrażenie `xlDelimited`, niż przeczytać 1.



**Rysunek 2.4** W oknie Immediate edytora VB, można sprawdzić prawdziwe wartości stałych, takich jak `xlFixedWidth`.

Jeśli byliśmy złośliwymi programistami, z pewnością mogliśmy zapamiętać wszystkie te stałe i pisać kod przy użyciu ekwiwalentów numerycznych tych stałych. Jednak bogowie programowania (i osoby, które muszą przeglądać ten kod) będą nas za to przeklinać.

W większości przypadków, w pliku pomocy albo są wyspecyfikowane prawidłowe wartości stałych, albo zamieszczone są łącza otwierające tematy pomocy z pełną listą wartości stałych (rysunek 2.5).

Wadą tego wspianego systemu pomocy jest to, że nie identyfikuje, które parametry są nowe w danej wersji. W tym konkretnym przypadku, parametr `TrailingMinusNumbers` był wprowadzony w programie Excel 2002. Jeśli damy ten program komuś, kto nadal używa wersji Excel 2000, kod nie będzie działał, ponieważ program Excel nie zrozumie parametru `TrailingMinusNumbers`. Niestety jedyna metoda poznania tych informacji przebiega na zasadzie prób i błędów.

Jeśli przeczytamy temat pomocy dotyczący metody `OpenText`, możemy przypuszczać, że jest to zasadniczo ekwiwalent otwierania pliku za pomocą Kreatora importu tekstu. W pierwszym kroku kreatora, zazwyczaj wybieramy parametr `Delimited` lub `Fixed Width` (rozdzielanie wartościami lub stała szerokość). Możemy również określić pochodzenie (origin) i wiersz początkowy. Ten pierwszy krok kreatora jest obsługiwany przez wymienione poniżej parametry metody `OpenText`:

```
Origin:=437
StartRow:=1
DataType:=xlDelimited
```

## xlColumnDataType Enumeration (Excel)

Office 2013 and later | [Other Versions](#)

## Contribute to this content

Use GitHub to [suggest and submit changes](#). See our [guidelines for contributing](#) to VBA documentation.

Specifies how a column is to be parsed.

| Name            | Value | Description           |
|-----------------|-------|-----------------------|
| xlDMYFormat     | 4     | DMY date format.      |
| xlDYMFormat     | 7     | DYM date format.      |
| xlEMDFormat     | 10    | EMD date format.      |
| xlGeneralFormat | 1     | General.              |
| xlMDYFormat     | 3     | MDY date format.      |
| xlMYDFormat     | 6     | MYD date format.      |
| xlSkipColumn    | 9     | Column is not parsed. |
| xlTextFormat    | 2     | Text.                 |
| xlYDMFormat     | 8     | YDM date format.      |
| xlYMDFormat     | 5     | YMD date format.      |

**Rysunek 2.5** Kliknięcie łącza pozwala zobaczyć wszystkie możliwe wartości stałych. Na rysunku, w nowym temacie pomocy pokazanych jest 10 możliwych stałych xlColumn-DataTypes.

Krok drugi kreatora importu tekstu pozwala wyspecyfikować, że nasze pola będą rozdzielane przecinkami. Ponieważ nie chcemy traktować dwóch przecinków jako jednego, nie powinno być zaznaczone pole wyboru. Kolejne ograniczniki traktuj jako jeden. Czasami pola mogą zawierać przecinki, na przykład „XYZ, Inc”. W takim przypadku, w polu wartości powinny być ujęte w cudzysłowy, tak jak specyfikuje to pole Kwalifikator tekstu. Ten drugi etap kreatora importu tekstu jest obsługiwany przez następujące parametry metody `OpenText`:

```
TextQualifier:=xlDoubleQuote
ConsecutiveDelimiter:=False
Tab:=False
Semicolon:=False
Comma:=True
Space:=False
Other:=False
```

Krok trzeci kreatora to etap, w którym faktycznie identyfikujemy typy plików. W tym przypadku, wszystkie pola pozostawiamy jako typ Ogólny, za wyjątkiem pierwszego pola, które oznaczone jest jako data w formacie DMR (Dzień, Miesiąc, Rok). W kodzie reprezentuje te informacje parametr `FieldInfo`.

Trzeci krok kreatora importu tekstu jest dość złożony. Cały parametr `FieldInfo` metody `OpenText` powiela wybory dokonane w tym kroku za pomocą kreatora. Jeśli na tym etapie kreatora klikniemy przycisk **Zaawansowane**, będziemy mieli możliwość wyspecyfikowania innych wartości niż domyślny separator dziesiętny i tysięcy, a także będziemy mogli włączyć opcję **Znak minus na końcu liczb ujemnych**.

---

#### — UWAGA —

---

Warto zwrócić uwagę, że rejestrator makr nie tworzy kodu dla parametru `DecimalSeparator` lub `ThousandsSeparator`, o ile nie zmienimy ich wartości domyślnych. Rejestrator makr zawsze jednak rejestruje parametr `TrailingMinusNumbers`.

---

Pamiętajmy, że podczas rejestrowania makra każde działanie wykonywane w programie Excel jest przekształcane na kod VBA. W przypadku wielu okien dialogowych, niezmiennie ustawienia są często rejestrowane wraz ze modyfikowanymi ustawieniami. Po kliknięciu **OK**, by zamknąć okno dialogowe, rejestrator makr często rejestruje w makro wszystkie bieżące ustawienia danego okna dialogowego.

Poniżej przedstawiono inny przykład. Kolejnym wierszem kodu makra jest zapis:

```
Selection.End(xlDown).Select
```

W tym wierszu kodu możemy kliknąć i uzyskać trzy tematy pomocy: `Selection`, `End` i `Select`. Zakładając, że samo słowo `Selection` i `Select` w pewnym stopniu wyjaśnia działanie, klikniemy słowo `End` i naciśniemy klawisz **F1**, by uzyskać pomoc. Wyświetlone zostanie kontekstowe okno dialogowe pomocy, informujące, że dla słowa `End` istnieją dwa obszary tematów pomocy – jedno w bibliotece programu Excel i drugie w bibliotece VBA.

Osoba słabo znająca VBA może nie wiedzieć, którą bibliotekę pomocy wybrać. Wybierzemy pierwszą i klikniemy przycisk **Help**. W tym przypadku wyświetlony zostanie temat w bibliotece VBA dotyczący instrukcji `End`, a to nie jest nam potrzebne.

Zamykamy pomoc, ponownie naciskamy **F1** i wybieramy obiekt `End` w bibliotece programu Excel. Ten temat pomocy informuje, że `End` to właściwość. Właściwość zwraca obiekt `Range` (zakres), który jest równoważny naciśnięciu w interfejsie programu Excel klawiszy `End+strzałka w górę` lub `End+strzałka w dół` (rysunek 2.6). Jeśli klikniemy niebieskie hiperłącze dla `xlDirection`, zobaczymy ważne parametry, które można przekazywać do funkcji `End`.

## Range.End Property (Excel)

Office 2013 and later | [Other Versions](#)

### Contribute to this content

Use GitHub to [suggest and submit changes](#). See our [guidelines for contributing](#) to VBA documentation.

Returns a **Range** object that represents the cell at the end of the region that contains the source range. Equivalent to pressing END+UP ARROW, END+DOWN ARROW, END+LEFT ARROW, or END+RIGHT ARROW. Read-only **Range** object.

### Syntax

*expression*.**End**(*Direction*)

*expression* A variable that represents a **Range** object.

### Parameters

| Name             | Required/Optional | Data Type          | Description                     |
|------------------|-------------------|--------------------|---------------------------------|
| <i>Direction</i> | Required          | <b>XIDirection</b> | The direction in which to move. |

**Rysunek 2.6** Prawidłowy temat pomocy dla właściwości End.

## Właściwości mogą zwracać obiekty

Przypomnijmy sobie z wcześniejszego opisu w tym rozdziale, że podstawowa składnia języka VBA to obiekt.*Metoda*. Przeanalizujemy poniższy wiersz kodu:

```
Selection.End(xlDown).Select
```

W tym wierszu kodu metodą jest *Select*. Słowo kluczowe *End* jest właściwością, ale na podstawie pliku pomocy wiemy, że zwraca ona obiekt *Range*. Ponieważ metoda *Select* może być stosowana do obiektu *Range*, metoda jest w rzeczywistości dodana do właściwości.

Opierając się na tych informacjach, możemy założyć, że w tym wierszu kodu *Selection* to obiekt. Jeśli klikniemy myszą w słowie *Selection* i naciśniemy klawisz F1, zobaczymy, że zgodnie z informacjami pomocy, *Selection* jest tak naprawdę właściwością, a nie obiektem. W rzeczywistości, prawidłowy kod powinien mieć postać *Application.Selection*. Jednakże, kiedy pracujemy w programie Excel, VBA zakłada, że odnosimy się do modelu obiektów programu Excel, tak więc możemy pominąć obiekt *Application*. Jeśli pisalibyśmy program w Word VBA, by zautomatyzować program Excel, wymagane byłoby dołączenie zmiennej obiektu przed właściwością *Selection*, by określić, do której aplikacji odwołujemy się.

W tym przypadku, *Application.Selection* może zwrócić kilka typów obiektów. Jeśli zaznaczona jest komórka, zwracany jest obiekt *Range*.

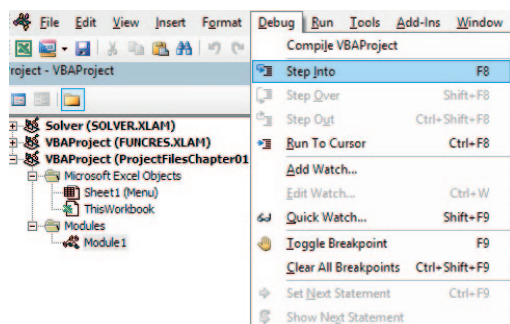
## Stosowanie narzędzi debugowania do analizy zarejestrowanego kodu

W kolejnych podrozdziałach omówimy niektóre doskonałe narzędzia debugowania dostępne w edytorze VB. Narzędzia te bardzo ułatwiają sprawdzenie tego, co dzieje się w zarejestrowanym kodzie.

### Krokowe wykonywanie kodu

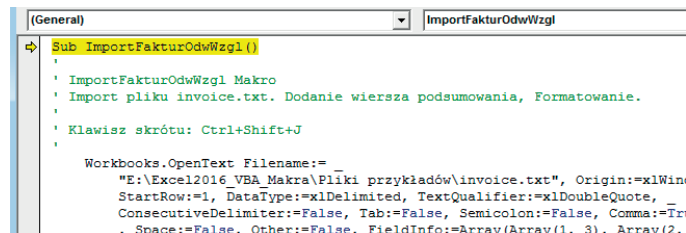
Mówiąc ogólnie, makro działa szybko: uruchamiamy makro i w przeciągu niecałej sekundy jest wykonane. Jeśli coś nie działa prawidłowo, nie mamy możliwości zauważenia, jakie działania wykonuje makro. Analizę umożliwia jednak funkcja programu Excel Step Into, która pozwala uruchamiać jeden wiersz kodu krok po kroku.

Aby użyć tej funkcji, trzeba upewnić się, czy kursor znajduje się w procedurze, którą chcemy uruchomić, na przykład w procedurze `ImportInvoice`, a następnie z menu `Debug` wybrać opcję `Step Into` (rysunek 2.7). Inny sposób uruchomienia funkcji to naciśnięcie klawisza `F8`.



Rysunek 2.7 Funkcja `Step Into` pozwala uruchamiać pojedynczo poszczególne wiersze kodu.

Edytor VB jest teraz w trybie `Break`. Uruchamiany wiersz jest wyróżniony kolorem żółtym i żółtą strzałką na marginesie przed kodem (rysunek 2.8).



Rysunek 2.8 Pierwszy wiersz makra zostanie za chwilę wykonany.

W tym przypadku, następnym wierszem, który zostanie wykonany, jest wiersz `ImportInvoice()`. Wiersz ten zazwyczaj informuje „Zamierzasz uruchomić tę procedurę”. Naciśnięcie klawisza `F8` powoduje wykonanie wiersza wyróżnionego kolorem żółtym