Spis treści

	<i>Podziękowania</i> xxi
	O autorachxxii
	Wprowadzenie xxiii
1	Zwiększanie możliwości programu Excel za pomocą języka VBA1
	Początkowe przeszkody 1
	Rejestrator makr nie działa! 2
	Nikt w zespole programu Excel nie poświęca wiele uwagi
	rejestratorowi makr
	Visual Basic nie przypomina języka BASIC2
	Dobra wiadomość: poznawanie języka VBA nie jest trudne
	Dobra wiadomość: Excel plus VBA są warte wkładanego wysiłku
	Poznawanie narzędzi: karta Deweloper 4
	Typy plików, dla których dopuszczane są makra5
	Bezpieczeństwo makr
	Dodawanie zaufanej lokalizacji7
	Zastosowanie ustawień makr w celu włączenia obsługi makr
	poza zaufanymi lokalizacjami 8
	Stosowanie opcji Wyłącz makra języka VBA z powiadomieniem
	Rejestrowanie, zapisywanie i uruchamianie makr
	Wypełnianie okna dialogowego Rejestrowanie makra
	Uruchamianie makra
	Tworzenie przycisku makra na wstążce 12
	Tworzenie przycisku makra na pasku narzędzi Szybki dostęp
	Przypisywanie makra do kontrolki formularza, pola tekstowego
	lub kształtu
	Działanie edytora Visual Basic15
	Ustawienia narzędzia VB Editor 16
	Eksplorator projektu 16
	Okno Properties 17
	Mankamenty rejestratora makr 18
	Rejestrowanie makra 20
	Analiza kodu w oknie programowania 21

	Uruchomienie tego samego makra innego dnia generuje nieoczekiwane wyniki Możliwa rozwiazapie: wykorzystywanie odwołać wzglodnych	23
	wozniwe rozwiązanie, wykorzystywanie odwołań wzgiędnych	24
	podczas rejestrowania	24
	Podczas rejestrowania nigdy nie używaj przycisku	~~
	Autosumowanie lub Szybka analiza.	29
	Cztery wskazówki dotyczące używania rejestratora makr	. 30
	Następne kroki	. 32
2	Skoro to BASIC, dlaczego nie wygląda znajomo?	33
	"Części mowy" języka VBA	34
	Język VBA naprawdę nie jest trudny	. 38
	Pliki pomocy VBA: Klawisz F1 do wyszukiwania potrzebnych informacji	. 38
	Korzystanie z pomocy	39
	Analiza kodu zarejestrowanego makra: korzystanie z edytora VB	
	i tematów pomocy	.40
	Parametry opcjonalne	41
	Zdefiniowane stałe	41
	Właściwości mogą zwracać obiekty	44
	Stosowanie narzędzi debugowania do analizy zarejestrowanego kodu	.45
	Krokowe wykonywanie kodu	.45
	Inne opcje debugowania: punkty przerwania	47
	Poruszanie się w kodzie w przód lub w tył	48
	Uruchamianie fragmentu kodu bez trybu krokowego	.48
	Tworzenie zapytań podczas krokowego wykonywania kodu	48
	Wykorzystywanie czujek do ustawiania punktów przerwań	51
	Stosowanie czujek do obiektów	51
	Narzędzie Object Browser: ostateczne źródło	53
	Siedem wskazówek poprawiania zarejestrowanego kodu	. 54
	Wskazówka 1: Niczego nie zaznaczaj	. 54
	Wskazówka 2: Używaj Cells(2,5), ponieważ jest wygodniejsze od	
	Range("E2")	55
	Wskazówka 3: Używaj bardziej niezawodnych sposobów	
	wyszukiwania ostatniego wiersza	56
	Wskazówka 4: Stosuj zmienne, by unikać "sztywnego"	
	kodowania wierszy i formuł	57
	Wskazówka 5: Używaj formuł typu R1C1, które ułatwiają życie	. 57
	Wskazówka 6: Kopiuj i wklejaj w pojedynczej instrukcji	58
	Wskazówka 7: Używaj konstrukcji WithEnd With	
	do wykonywania wielu działań	58

	Następne kroki	62
3	Odwoływanie się do zakresów Obiekt Range	63 64
	Składnia specyfikowania zakresu	64
	Zakresy nazwane	65
	Skrótowe odwołania do zakresów	65
	Odwoływanie się do zakresow w innych arkuszach	65
	Odwoływanie się do zakresu względem innego zakresu	67
	Stosowanie właściwości Offset do odwoływania się do zakresu	، 10 60
	Lizuwanie właściwości Oriset do odwoływania się do zakresu	09 71
	Stosowanie właściwości Columns i Rows do określania zakresu	72
	łaczenie wielu zakresów	73
	Tworzenie nowego zakresu na podstawie nakładających się zakresów	73
	Sprawdzanie, czy komórka jest pusta	73
	Zaznaczanie zakresu danych.	74
	Stosowanie kolekcji Areas do zwracania nieciągłego zakresu	78
	Odwołania do tabel	78
	Następne kroki	79
4	Petle i sterowanie przepływem	81
4	Pętle i sterowanie przepływem. Pętle ForNext.	81 . 81
4	Pętle i sterowanie przepływem. Pętle ForNext. Stosowanie zmiennych w instrukcji For	81 . 81 . 84
4	Pętle i sterowanie przepływem. Pętle ForNext. Stosowanie zmiennych w instrukcji For Warianty pętli ForNext	81 . 81 . 84 85
4	Pętle i sterowanie przepływem.Pętle ForNext.Stosowanie zmiennych w instrukcji ForWarianty pętli ForNextWcześniejsze przerywanie pętli po spełnieniu warunku.	81 . 81 . 84 85 86
4	Pętle i sterowanie przepływem. Pętle ForNext. Stosowanie zmiennych w instrukcji For Warianty pętli ForNext Wcześniejsze przerywanie pętli po spełnieniu warunku. Zagnieżdżanie pętli	81 . 81 . 84 85 85 86 86
4	Pętle i sterowanie przepływem. Pętle ForNext. Stosowanie zmiennych w instrukcji For Warianty pętli ForNext Wcześniejsze przerywanie pętli po spełnieniu warunku. Zagnieżdżanie pętli Pętle Do	81 . 81 . 84 85 86 86 87
4	Pętle i sterowanie przepływem. Pętle ForNext. Stosowanie zmiennych w instrukcji For Warianty pętli ForNext. Wcześniejsze przerywanie pętli po spełnieniu warunku. Zagnieżdżanie pętli Pętle Do. Stosowanie klauzul While lub Until w pętlach Do.	81 . 81 . 84 85 . 86 . 86 . 87 . 90
4	Pętle i sterowanie przepływem. Pętle ForNext. Stosowanie zmiennych w instrukcji For Warianty pętli ForNext Wcześniejsze przerywanie pętli po spełnieniu warunku. Zagnieżdżanie pętli Pętle Do. Stosowanie klauzul While lub Until w pętlach Do. Pętla VBA: For Each	81 .81 .84 .85 86 86 87 .90 92
4	Pętle i sterowanie przepływem. Pętle ForNext. Stosowanie zmiennych w instrukcji For Warianty pętli ForNext Wcześniejsze przerywanie pętli po spełnieniu warunku. Zagnieżdżanie pętli Pętle Do. Stosowanie klauzul While lub Until w pętlach Do. Pętla VBA: For Each Zmienne obiektowe	81 . 81 . 84 85 . 86 . 86 . 86 . 87 . 90 . 92 . 92
4	Pętle i sterowanie przepływem. Pętle ForNext. Stosowanie zmiennych w instrukcji For Warianty pętli ForNext. Wcześniejsze przerywanie pętli po spełnieniu warunku. Zagnieżdżanie pętli Pętle Do. Stosowanie klauzul While lub Until w pętlach Do. Pętla VBA: For Each Zmienne obiektowe Sterowanie przepływem: stosowanie konstrukcji IfThenElse i Select Case	81 . 81 85 85 86 86 86 90 90 92 94
4	Pętle i sterowanie przepływem. Pętle ForNext. Stosowanie zmiennych w instrukcji For Warianty pętli ForNext Wcześniejsze przerywanie pętli po spełnieniu warunku Zagnieżdżanie pętli Pętle Do Stosowanie klauzul While lub Until w pętlach Do. Pętla VBA: For Each Zmienne obiektowe Sterowanie przepływem: stosowanie konstrukcji IfThenElse i Select Case Podstawowe sterowanie przepływem: IfThenElse	81 84 85 85 86 86 86 90 94 94 94
4	Pętle i sterowanie przepływem. Pętle ForNext. Stosowanie zmiennych w instrukcji For Warianty pętli ForNext Wcześniejsze przerywanie pętli po spełnieniu warunku Zagnieżdżanie pętli Pętle Do. Stosowanie klauzul While lub Until w pętlach Do. Pętla VBA: For Each Zmienne obiektowe Sterowanie przepływem: stosowanie konstrukcji IfThenElse i Select Case Podstawowe sterowanie przepływem: IfThenElse Stosowanie konstrukcji Select CaseEnd Select dla wielu warunków.	81 .81 .84 .85 .86 .86 .86 .87 .90 .92 .92 94 .94 .94
4	Pętle i sterowanie przepływem. Pętle ForNext. Stosowanie zmiennych w instrukcji For Warianty pętli ForNext Wcześniejsze przerywanie pętli po spełnieniu warunku. Zagnieżdżanie pętli Pętle Do. Stosowanie klauzul While lub Until w pętlach Do. Pętla VBA: For Each Zmienne obiektowe Sterowanie przepływem: stosowanie konstrukcji lfThenElse i Select Case Podstawowe sterowanie przepływem: lfThenElse Stosowanie konstrukcji Select CaseEnd Select dla wielu warunków. Następne kroki	81 . 81 85 85 86 86 90 90 92 94 94 96 100
4	Pętle i sterowanie przepływem. Pętle ForNext. Stosowanie zmiennych w instrukcji For Warianty pętli ForNext Wcześniejsze przerywanie pętli po spełnieniu warunku. Zagnieżdżanie pętli Pętle Do. Stosowanie klauzul While lub Until w pętlach Do. Pętla VBA: For Each Zmienne obiektowe Sterowanie przepływem: stosowanie konstrukcji IfThenElse i Select Case Podstawowe sterowanie przepływem: IfThenElse Stosowanie konstrukcji Select CaseEnd Select dla wielu warunków. Następne kroki	81 . 84 85 . 86 . 86 . 86 . 87 . 90 90 94 96 100 . 101
4	Pętle i sterowanie przepływem. Pętle ForNext. Stosowanie zmiennych w instrukcji For Warianty pętli ForNext Wcześniejsze przerywanie pętli po spełnieniu warunku. Zagnieżdżanie pętli Pętle Do. Stosowanie klauzul While lub Until w pętlach Do. Pętla VBA: For Each Zmienne obiektowe Sterowanie przepływem: stosowanie konstrukcji IfThenElse i Select Case Podstawowe sterowanie przepływem: IfThenElse Stosowanie konstrukcji Select CaseEnd Select dla wielu warunków. Następne kroki Zmiana odwołań na styl R1C1	81 .81 .84 .85 .86 .86 .86 .87 .90 .92 .92 94 .94 .94 .94 100 101 102
4	Pętle i sterowanie przepływem. Pętle ForNext. Stosowanie zmiennych w instrukcji For Warianty pętli ForNext Wcześniejsze przerywanie pętli po spełnieniu warunku. Zagnieżdżanie pętli Pętle Do. Stosowanie klauzul While lub Until w pętlach Do. Pętla VBA: For Each Zmienne obiektowe. Sterowanie przepływem: stosowanie konstrukcji IfThenElse i Select Case Podstawowe sterowanie przepływem: IfThenElse . Stosowanie konstrukcji Select CaseEnd Select dla wielu warunków. Następne kroki Formuły w stylu R1C1 Zmiana odwołań na styl R1C1 Magia formuł programu Excel	81 84 85 85 86 86 86 90 90 94 94 96 94 96 94 96 94 96 91 94 96 94 96 94 96 91 94 96 94 96 94 96 96 94 96 94 96 94 96 94 96 94 96 94 96 94 96 94 96 94 96 94 96 96 94 96 96 96 90
4	Pętle i sterowanie przepływem. Pętle ForNext. Stosowanie zmiennych w instrukcji For Warianty pętli ForNext Wcześniejsze przerywanie pętli po spełnieniu warunku. Zagnieżdżanie pętli Pętle Do. Stosowanie klauzul While lub Until w pętlach Do. Pętla VBA: For Each Zmienne obiektowe Sterowanie przepływem: stosowanie konstrukcji IfThenElse i Select Case Podstawowe sterowanie przepływem: IfThenElse . Stosowanie konstrukcji Select CaseEnd Select dla wielu warunków. Następne kroki Zmiana odwołań na styl R1C1 Zmiana odwołań na styl R1C1 Magia formuł programu Excel Wprowadź formułę raz i skopiuj ją 1000 razy!	81 . 81 . 84 85 . 86 . 86 . 87 . 90 94 94 96 100 101 102 . 103 . 103

	Istota stylu odwołań R1C1 Używanie stylu R1C1 dla odwołań względnych	106 106
	Stosowanie stylu R1C1 dla odwołań bezwzględnych	107
	Stosowanie notacji R1C1 przy odwołaniach mieszanych	107
	Odwoływanie się do całych kolumn lub wierszy	108
	Zastępowanie wielu formuł A1 pojedynczą formułą R1C1	. 109
	Zapamiętywanie numerów odpowiadających literom kolumn	. 111
	Następne kroki	111
6	Tworzenie nazw i operacje na nazwach w VBA	113
	Nazwy globalne kontra lokalne	113
	Dodawanie nazw	114
	Usuwanie nazw	116
	Dodawanie komentarzy	. 116
	Typy nazw	117
	Formuły	. 117
	Ciągi znaków	. 118
	Liczby	119
	Tabele	. 120
	Używanie tablic w nazwach	. 121
	Nazwy zastrzeżone	. 121
	Ukrywanie nazw	. 123
	Sprawdzanie istnienia nazwy.	. 123
	Nastepne kroki	126
7	Programowanie zdarzeń	. 127
-	Poziomy zdarzeń	127
	Używanie zdarzeń	128
	Parametry zdarzenia	129
	Właczanie zdarzeń	129
	Zdarzenia poziomu skoroszvtu	130
	Zdarzenia dotyczące arkusza i wykresu na poziomie skoroszytu.	. 133
	Zdarzenia dotyczące arkuszą.	. 136
	Zdarzenia dotyczace wykresów	137
	Wykresy osadzone	138
	Zdarzenia dotyczące wykresu osadzonego i arkusza wykresu	138
	Zdarzenia na poziomie aplikacij	140
	Następne kroki	146
8	Tablice	147
	Deklarowanie tablicy	1/17
		. 147

	Deklarowanie wielowymiarowej tablicy Wypełnianie tablicy Pobieranie danych z tablicy	148 149 . 150
	Wykorzystywanie tablic do przyspieszenia działania kodu Tablice dynamiczne Przekazywanie tablicy Następne kroki	151 152 .154 .154
9	Tworzenie klas i kolekcji	155
	Wstawianie modułu klasy.	. 156
	Przechwytywanie zdarzeń dotyczących aplikacji i wykresów osadzonych Zdarzenia dotyczące aplikacji	. 156 156 158
	Tworzenie objektu niestandardowego	160
	Używanie objektu niestandardowego	162
	Używanie kolekcii	163
	Tworzenie kolekcii	. 163
	Tworzenie kolekcji w module standardowym	164
	Tworzenie kolekcji w module klasy	166
	Używanie słowników	. 168
	Używanie typów zdefiniowanych przez użytkownika	
	do tworzenia właściwości niestandardowych	173
	Następne kroki	. 176
10	Formularze użytkownika – wprowadzenie	. 177
	Pola wprowadzania danych	178
	Pola komunikatów	. 178
	Tworzenie formularza użytkownika	179
	Wywoływanie i ukrywanie formularza użytkownika	181
	Programowanie formularzy użytkownika	. 181
	Zdarzenia związane z formularzami użytkownika	. 181
	Programowanie kontrolek	. 183
	Używanie podstawowych kontrolek formularza	184
	Stosowanie etykiet, pól tekstowych i przycisków poleceń	184
	Wybór pomiędzy polami list a polami kombi w formularzach	187
	Stosowanie właściwości MultiSelect pola listy	. 189
	Dodawanie przycisków opcji do formularza użytkownika	191
	Dodawanie grafiki do formularza użytkownika	. 193
	Stosowanie przycisku pokrętła na formularzu użytkownika	194
	Stosowanie kontrolki MultiPage do łączenia formularzy	196
	Weryfikowanie wpisów w polach	. 199

x	Spis treści

	Nieprawidłowe zamknięcie okna	199 200
	Następne kroki	.202
11	Analiza danych za pomocą funkcji Filtr zaawansowany Zastępowanie pętli funkcją Autofiltr. Wykorzystywanie funkcji Autofiltr. Zaznaczanie tylko widocznych komórek Filtr zaawansowany – łatwiej w VBA.	203 203 206 210 . 212
	Korzystanie z interfejsu uzytkownika do budowania filtru zaawansowanego	213
	Stosowanie filtru zaawansowanego do wydobycia listy unikatowych wartości Uzyskiwanie listy unikatowych wartości za pomocą interfejsu	214
	użytkownika	. 214
	Wyodrębnianie listy unikatowych wartości za pomocą kodu VBA	. 216
	Uzyskiwanie unikatowej kombinacji dwóch lub większej liczby pół	.220
	stosowanie filtru zaawansowanego z zakresami kryteriow	. 221 223
	łączenie wielu kryteriów za pomocą konjunkcji (AND)	223
	Inne trochę bardziej złożone zakresy kryteriów Najbardziej złożone kryteria: Zastepowanie listy wartości	224
	warunkiem utworzonym jako wynik formuły Stosowanie warunków opartych na formule w kodzie VBA	224 228
	Stosowanie funkcji Filtr zamiast Filtr zaawansowany Brak rekordów w wyniku przy użyciu opcji Filtruj listę na miejscu Wyświetlanie wszystkich rekordów po uruchomieniu filtrowania	.233 .233
	listy na miejscu Prawdziwy koń pociągowy: xlFilterCopy dla wszystkich rekordów, a	234
	nie tylko unikatowych Kopiowanie wszystkich kolumn Kopiowanie podzestawu kolumn i zmiana ich kolejności Excel w praktyce: Wyłączanie kilku list rozwijanych funkcji Autofiltr	234 235 236 .242
	Następne kroki	243
12	Wykorzystywanie VBA do tworzenia tabel przestawnych	245 246 247 247 .248 .249

	Powody, dla których nie można przenosić lub zmieniać	
	fragmentów raportu przestawnego	.252
	Określanie rozmiaru gotowej tabeli przestawnej w celu	
	przekształcenia jej na wartości	.252
	Stosowanie zaawansowanych funkcji tabel przestawnych	.255
	Używanie wielu pól wartości	255
	Grupowanie dat poszczególnych dni według miesięcy, kwartałów lub lat	.256
	Zmiana obliczeń w celu prezentowania wartości procentowych	258
	Eliminowanie pustych komórek w obszarze wartości	261
	Kontrolowanie kolejności sortowania za pomocą opcji	
	autosortowania (AutoSort)	261
	Powielanie raportu dla każdego produktu	.262
	Filtrowanie zestawu danych	.265
	Ręczne filtrowanie dwóch lub kilku elementów w polu tabeli przestawne	j 265
	Stosowanie filtrów pojęciowych	266
	Stosowanie filtrów wyszukiwania	. 271
	Konfigurowanie fragmentatorów w celu filtrowania tabeli przestawnej	.274
	Konfigurowanie osi czasu tabeli przestawnej	278
	Formatowanie przecięcia wartości w tabeli przestawnej	281
	Wykorzystywanie modelu danych w programie Excel	.282
	Dodanie obu tabel do Modelu danych	282
	Tworzenie relacji pomiędzy dwoma tabelami	.283
	Definiowanie bufora tabeli przestawnej i tworzenie tabeli przestawnej	.283
	Dodawanie pól modelu do tabeli przestawnej	.284
	Dodawanie pól numerycznych do obszaru wartości	284
	Zebranie wszystkiego razem	.285
	Inne funkcje tabel przestawnych	287
	Obliczeniowe pola danych	287
	Elementy obliczeniowe	288
	Używanie właściwości ShowDetail do filtrowania zestawu rekordów	288
	Zmiana układu na karcie Projektowanie	289
	Ustawienia układu raportu	289
	Wyłączanie sum częściowych dla wielu pól wierszy	.290
	Porównanie VBA z TypeScript	. 291
	Następne kroki	295
13	Siła programu Excel	.297
	Operacje na plikach	298
	Tworzenie listy plików w katalogu	298
	Importowanie i usuwanie pliku CSV	300

	Wczytanie pliku tekstowego do pamięci i jego analiza	301
	Łączenie i rozdzielanie skoroszytów	302
	Rozdzielanie arkuszy w oddzielnych skoroszytach	302
	Łączenie skoroszytów	303
	Kopiowanie danych do oddzielnych arkuszy bez użycia filtru	304
	Eksportowanie danych do pliku XML	305
	Umieszczanie wykresu w komentarzu	306
	Śledzenie zmian użytkownika	308
	Metody dla profesjonalistów języka VBA	309
	Tworzenie w programie Excel modułu klasy stanu	309
	Drążenie w głąb tabel przestawnych	311
	Filtrowanie tabeli przestawnej OLAP według listy elementów	312
	Tworzenie niestandardowej kolejności sortowania	314
	Tworzenie wskaźnika postępu	. 315
	Stosowanie chronionych pól haseł	317
	Zaznaczanie za pomocą SpecialCells	319
	Resetowanie formatu tabeli	319
	Używanie VBA Extensibility w celu dodawania kodu do nowych	
	skoroszytów	320
	Konwertowanie formatowanego raportu o stałej szerokości na	
	zbiór danych	322
	Następne kroki	325
14	Przykłady funkcji definiowanych przez użytkownika	327
	Tworzenie funkcji definiowanych przez użytkownika	327
	Budowanie prostej funkcji użytkownika	328
	Udostępnianie funkcji UDF	330
	Użyteczne niestandardowe funkcje programu Excel	330
	Sprawdzenie, czy skoroszyt jest otwarty	330
	Sprawdzenie, czy istnieje arkusz w otwartym skoroszycie	331
	Zliczanie liczby skoroszytów w katalogu	332
	Pobieranie ID użytkownika	333
	Pobieranie informacji o dacie i godzinie ostatniego zapisu	. 334
	Pobieranie informacji o niezmieniającej się dacie i godzinie	335
	Sprawdzanie poprawności adresu e-mail	335
	Sumowanie komórek w oparciu o kolor ich wypełnienia	337
	Zliczanie unikatowych wartości	338
	Wyszukiwanie w zakresie pierwszej komórki o niezerowej długości	339
	Zastępowanie wielu znaków	340
	Uzyskiwania liszb z miaszanago takstu	2/1

	Przekształcenie numeru tygodnia na datę	. 342
	Sortowanie i łączenie	. 343
	Sortowanie liczb i znaków alfanumerycznych	344
	Wyszukiwanie ciągu wewnątrz tekstu	346
	Zwracanie adresów duplikatów wartości maksymalnych	347
	Zwracanie adresu hiperłącza	. 348
	Zwracanie litery kolumny na podstawie adresu komórki	348
	Używanie Select Case na arkuszu	349
	Tworzenie funkcji LAMBDA	350
	Budowanie prostej funkcji LAMBDA	350
	Udostępnianie funkcji LAMBDA	. 352
	Przydatne funkcje LAMBDA	352
	Następne kroki	355
15	Tworzenie wykresów	357
15	Stosowanie metody. AddChart2 do tworzenia wykresu	358
	Style wykresu	359
	Formatowanie wykresu	362
	Odwoływanie się do określonego wykresu	362
	Specyfikowanie tytułu wykresu	363
	Stosowanie koloru wykresu	. 364
	Filtrowanie wykresu	
	Używanie metody SetElement do emulowania zmian	
	dostepnych w menu ikony Plus	. 367
	Stosowanie metody Format do zarządzania formatowaniem	372
	Zmiana wypełnienia obiektu	373
	Formatowanie ustawień linii	375
	Tworzenie wykresów kombi	376
	Tworzenie wykresów kartogramowych	380
	Tworzenie wykresów kaskadowych.	. 381
	Eksportowanie wykresu jako grafiki	382
	Uwaględnianie kompatybilności wstecznej	. 383
	Następne kroki	. 383
16	Wizualizacio danych i formatowanio warunkowo	205
10	Metody i jch właściwości do wizualizacji dapych	. 303
	Dodawanie pasków danych do zakresów	388
	Dodawanie skali kolorów do zakresu	202
	Dodawanie do zakresu zestawów ikop	301
	Specyfikowanie zestawu ikon	205
	Specyfikowanie zestawu kon	206 205
		. 550

xiv	Spis	treści
	00.0	

	Triki wizualizacji danych	397
	Tworzenie zestawu ikon dla podzbioru zakresu	397
	Używanie w zakresie dwóch kolorów dla pasków danych	399
	Inne metody formatowania warunkowego	402
	Formatowanie komórek, których wartości są powyżej lub	
	poniżej średniej	402
	Formatowanie komórek, których wartości należą do pierwszych	
	10 lub ostatnich 5	403
	Formatowanie unikatowych lub duplikowanych komórek	404
	Formatowanie komórek w oparciu o ich wartość	405
	Formatowanie komórek zawierających tekst	
	Formatowanie komórek, które zawierają daty	406
	Formatowanie komórek, które są puste lub zawierają błędy	. 406
	Używanie formuły do określenia, które komórki mają być formatowane	407
	Stosowanie właściwości NumberFormat	409
	Następne kroki	. 410
17	Tworzenie pulpitów nawigacyjnych za pomoca wykresów	
	przebiegu w czasie	. 411
	Tworzenie wykresów przebiegu w czasie	412
	Skalowanie wykresów przebiegu w czasie	. 414
	Formatowanie wykresów przebiegu w czasie	418
	Stosowanie kolorów motywu	418
	Stosowanie kolorów RGB	422
	Formatowanie elementów wykresów przebiegu w czasie	423
	Formatowanie wykresów Zysk/strata	426
	Tworzenie pulpitu nawigacyjnego	428
	Uwagi dotyczące wykresów przebiegu w czasie	428
	Tworzenie setek indywidualnych wykresów przebiegu w czasie	
	na pulpicie nawigacyjnym	429
	Następne kroki	434
18	Odczytywanie stron web przy użyciu jezyków M i VBA	435
	Uzyskanie poświadczeń dostępowych dla API	436
	Budowanie w Power Ouery kwerendy pobierającej dane dla jednej	150
	ustalonei wartości	437
	Odświeżanie poświadczeń po ich wygaśnieciu	441
	Budowanie niestandardowei funkcii w Power Ouerv	442
	Używanie nowej funkcji w naszym kodzie	444
	Duplikowanie istniejacego zapytania w celu utworzenia nowego	
	Odpytywanie listy piosenek w albumie	447

	Uogólnianie zapytań za pomocą VBA	
	Upraszczanie zapytania SearchArtist do pojedynczego wiersza kodu	448
	Upraszczanie zapytania ArtistAlbums	
	Upraszczanie zapytania AlbumTracks	
	Grupowanie zapytań, aby wysprzątać listę	.450
	Planowanie rozmieszczenie wyników zapytania na naszej tablicy	
	kontrolnej	451
	Używanie zmiennych globalnych i pętli w języku M	456
	Przechowywanie zmiennych globalnych w rekordzie Settings	
	w Power Query	.456
	Prosta obsługa błędów za pomocą try i otherwise	457
	Używanie logiki warunkowej w języku M	458
	Pętle wykorzystujące List.Generate	458
	Używanie metody Application.OnTime do okresowej analizy danych	. 461
	Używanie trybu Ready w zaplanowanych procedurach	462
	Specyfikowanie okna czasowego dla aktualizacji	.462
	Anulowanie poprzednio zaplanowanego makra	462
	Zamknięcie programu Excel anuluje wszystkie oczekujące	
	zaplanowane makra	.463
	Planowanie uruchamiania makra x minut w przyszłości	
	Planowanie przypomnienia słownego	464
	Zaplanowanie uruchamiania makra co dwie minuty	.465
	Następne kroki	466
19	Przetwarzanie plików tekstowych	467
	Importowanie z plików tekstowych	
	Importowanie plików tekstowych, które maja mniej niż 1048 576 wierszy	/.467
	Obsługa plików tekstowych zawierających wiecej niż 1048 576 wierszy.	.475
	Zapisywanie plików tekstowych	.480
	Nastepne kroki	480
20		404
20		481
	Stosowanie wczesnego wiązania do odwoływania się do obiektów	400
		482
	Stosowanie poznego wiązania do odwołan do obiektow programu Word.	484
	Stosowanie słowa kluczowego New w odwołaniach do aplikacji Word	485
	Stosowanie CreateObject do tworzenia nowej instancji obiektu	486
	Stosowanie GetObject do odwoływania się do istniejącej instancji	100
		486
	Uzywanie wartości stałych	488
	Uzywanie okna czujek do uzyskiwania rzeczywistych wartości stałych	488

	Stosowanie wyszukiwarki obiektów do uzyskania rzeczywistych	
	wartości stałych	489
	Obiekty programu Word	490
	Obiekt Document	. 491
	Obiekt Selection	.493
	Obiekt Range	494
	Zakładki	.498
	Kontrolowanie pól formularzy w programie Word	499
	Następne kroki	502
21	Wykorzystanie programu Access jako zaplecza dla dostępu	
	do danych wielu użytkowników	503
	Porównanie ADO i DAO	504
	Narzedzia obiektów ADO	.507
	Dodawanie rekordów do bazy danych	509
	Pobieranie rekordów z bazy danych	. 510
	Aktualizowanie istniejącego rekordu	513
	Usuwanie rekordów poprzez ADO	515
	Sumowanie rekordów za pośrednictwem ADO	. 515
	Inne narzędzia wykorzystujące pośrednictwo ADO	. 517
	Sprawdzanie, czy tabela istnieje	517
	Sprawdzenie, czy pole istnieje	518
	Dodawanie tabeli w locie	519
	Dodawanie pola w locie	.520
	Przykłady dla SOL Server	.520
	Nastepne kroki	522
22	Zaawancowana tachniki ctocowania formularzy użytkownika	E 2 2
"	Stosowania paska parzodzi UsorForm w projektowaniu kontrolek	525
	na formularzach	522
		525
	Kontrolki Chackboy	524
	Kontrolki TabStrin	.524
		520
	Kontrolki TegglePutton	529
	Stosowania packa przewijania jako guwaka wyberu wartości	550
	Stosowanie paska przewijania jako suwaka wyboru wartosci	221
		555
		535
	Stosowanie niperłączy w formularzach uzytkownika	.536
		.53/
	Zmiana rozmiaru formularza użytkownika w locie	539

Dodawanie kontrolek w locie	
Zmiana rozmiaru w locie	
Dodawanie innych kontrolek	540
Dodawanie obrazu na bieżąco	541
Podsumowanie	542
Dodawanie wskazówek pomocy do formularza użytkownika	544
Wyświetlanie klawiszy akceleratorów	
Dodawanie kontrolki porady tekstowej	545
Określanie kolejności kart	
Kolorowanie kontrolki aktywnej	
Tworzenie przezroczystych formularzy	549
Następne kroki	550
23 Interfeis programowania aplikacii (API)	551
Deklaracje interfejsu API.	552
Używanie deklaracji API	553
Tworzenie deklaracji API zgodnych z systemami 32- i 64-bitowymi	
Przykłady funkcji API	
Uzyskiwanie nazwy komputera	555
Sprawdzenie, czy plik programu Excel jest otwarty w sieci	
Uzyskiwanie informacji dotyczących rozdzielczości ekranu	556
Dostosowywanie okna dialogowego Windows – informacje	
Blokowanie przycisku X do zamykania formularza użytkownika.	
Tworzenie czasomierza	
Odtwarzanie dźwięków	560
Następne kroki	
24 Obsługa bładów	561
Co dzieje się kiedy pojawia się bład?	561
Mylace błedy debugowania w formularzu użytkownika	563
Podstawowa obsługa błędów za pomoca składni On Error GoTo	565
Ogólne programy obsługi błędów	567
Obsługa błedów poprzez ich ignorowanie	567
Blokowanie ostrzeżeń programu Excel	
Celowe wywoływanie błedu	570
Szkolenia klientów	
Błedy, które nie ujawniają się w trybie debugowania	
Błędy podczas projektowania a błędy występujące kilka miesiecy po	óźniej 572
Runtime Error 9: Indeks poza zakresem	573
Runtime Error 1004: Niepowodzenie metody Range dla obiektu	_Global .575
Dolegliwości kodu chronionego	576

	Więcej problemów dotyczących haseł	577
	Błędy powodowane przez zmiany wersji	578
	Następne kroki	.579
25	Dostosowywanie wstążki w celu uruchamiania makr	581
	Gdzie dodawać kod: Folder i plik customui	.582
	Tworzenie kart i grup	.584
	Dodawanie kontrolki do wstążki	.584
	Uzyskiwanie dostępu do struktury plików	591
	Działanie pliku RELS	592
	Zmiana nazwy pliku Excel i otwieranie skoroszytu	593
	Używanie obrazów na przyciskach	593
	Stosowanie na wstążce ikon pakietu Microsoft Office	.594
	Dodawanie obrazów ikon do wstążki	.595
	Rozwiązywanie problemów dotyczących komunikatów o błędach	597
	Atrybut "Attribute Name" dla elementu "customui Ribbon" nie	
	jest zdefiniowany w schemacie lub definicji DTD	597
	Niedozwolony znak w nazwie kwalifikowanej	598
	Element znacznika "customui" nie jest poprawny względem	
	zawartości elementu nadrzędnego	598
	Problemy z pewnymi zawartościami	599
	Nieprawidłowa liczba argumentów lub nieprawidłowe	
	przypisanie właściwości	600
	Nieprawidłowy format lub rozszerzenie pliku	600
	Nic się nie stało	601
	Inne sposoby uruchamiania makr	. 601
	Stosowanie skrótów klawiszowych do uruchamiania makra	. 601
	Dołączanie makra do przycisku polecenia	.602
	Dołączanie makra do kształtu	603
	Dołączanie makra do kontrolki ActiveX	.603
	Uruchamianie makra za pomocą hiperłącza	604
	Następne kroki	.605
26	Tworzonia dodatków	607
20	Cochy standardowych dodatków	.007
	Przekształcanie skoroszytu programu Evcel na dodatek	608
	Lizwanie funkcji Zaniswanie jako do przekształcenia pliku pa dodatek	600
	Stosowanie edutora VB do konwersii pliku na dodatek	610
	Instalowanie dodatku w systemie klienckim	611
	Standardowe dodatki nje sa beznjeczne	612
		617
		014

	Usuwanie dodatków	614
	Ukryty arkusz jako alternatywa dla dodatku	614
	Następne kroki	616
27	Wprowadzenie do tworzenia dodatków pakietu Office	. 617
	Tworzenie pierwszego dodatku pakietu Office – Hello World	618
	Dodawanie mechanizmów interakcji do dodatku pakietu Office	623
	Wstęp do języka HTML	626
	Stosowanie znaczników	626
	Dodawanie przycisków	627
	Wykorzystanie plików CSS	628
	Używanie XML do definiowania dodatku pakietu Office	628
	Wykorzystanie kodu JavaScript w celu dodania interakcji	
	do dodatku pakietu Office	. 629
	Struktura funkcji	630
	Nawiasy klamrowe i spacje	630
	Średniki i znaki podziału wiersza	630
	Komentarze	631
	Zmienne	631
	Ciągi	632
	Tablice	. 632
	Petle for w kodzie JavaScript	633
	Działanie instrukcji if w kodzie JavaScript	634
	Działanie instrukcji SelectCase w kodzie JavaScript	634
	Działanie instrukcji For eachnext w kodzie JavaScript	636
	Operatory matematyczne, logiczne i używane do przypisywania	637
	Funkcje matematyczne w JavaScript	638
	Zapisywanie w okienku zawartości lub okienku zadań	640
	Modyfikacje kodu JavaScript, by działał w dodatku pakietu Office	640
	Następne kroki	. 641
28	Nowości i zmiany w Excel 365	643
	Subskrypcja Office 365 czy "wieczna" licencja Excel 2021?	643
	Jeśli coś zmieniło się w interfejsie użytkownika, zmieniło się też w VBA	644
	Wstążka	. 644
	Interfejs SDI (Single Document Interface)	644
	Nowoczesne formuły tablicowe	645
	Funkcja LAMBDA	646
	Narzędzie Szybka analiza	646
	Wykresy	646
	Tabele przestawne	647

Fragmentatory	647
Ikony	.648
Modele 3D	.648
Grafiki SmartArt	648
TypeScript	649
Poznawanie nowych obiektów i metod	649
Tryb zgodności	.649
Stosowanie właściwości Version	.650
Stosowanie właściwości Excel8CompatibilityMode	. 651
Następne kroki	. 651
Nazwy funkcji i błędów	.653
Angielskie >>> polskie	.653
Polskie >>> angielskie	659
Indeks	665

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. Loretta Yates z wydawnictwa Pearson jest świetnym redaktorem treści. Dziękuję Kughens za kierowanie produkcją tej ksiaźki. Nad tym wydaniem pracowałem między innymi w Kola Mi Writing Camp. Dziękuję serdecznie całemu personelowi za dopilnowanie, abym nie zbaczał z drogi podczas pracy.

Niemal wszystko, czego nauczyłem się na temat programowania VBA, zawdzięczam cudownej społeczności forum MrExcel.com. VoG, Richard Schollar i Jon von der Heyden szczególnie zasługują na wspomnienie, gdyż ich posty wniosły wiele pomysłów do tej książki. Dziękuję Pam Gensel za pierwszą lekcję na temat makr w Excelu. Mala Singh nauczyła mnie tworzenia wykresów VBA. Suat Özgür zadbał, abym był na bieżąco z nowymi trendami w VBA i wniósł wiele pomysłów do rozdziału 18.

Moja rodzina była niezwykle pomocna i cierpliwa w tym czasie. Dziękuję wam, Mary Ellen, Robert F., Barbara i Robert K.!

– Bill

Dziękuję wszystkim moderatorom forum MrExcel, którzy dbają o porządek pomimo wielkich wysiłków ze strony spamerów. Dziękuję Joe4, RoryA u Petersss za pomoc w przetwarzaniu wszystkich maili kontaktowych forum.

Programowanie to ciągłe uczenie się i naprawdę doceniam klientów, którzy zachęcili mnie do wyjścia poza moją strefę komfortu i rozpoczęcie pracy w nieznanej wcześniej dziedzinie, przez co moje umiejętności i wiedza powiększyły się tak bardzo. Dziękuję Suat Özgür za pomoc w przezwyciężaniu niektórych naprawdę zagmatwanych zagadek programistycznych.

Final Fantasy XIV stało się moim drugim domem. Chciałabym szczególnie podziękować przyjaciołom z gry, którzy nie tylko sprawiają, że jest to tak fajne, ale również pomogli mi odnaleźć się w nieznanych światach: War, Chraz i Shabadoo. Dziękuję, że dzielicie się ze mną upodobaniem w grze.

Na koniec chcę podziękować Billowi. Jego witryna, MrExcel.com, jest tym miejscem, do którego tysiące przychodzą 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

O 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 2300 odcinków swojego codziennego podcastu wideo, *Learn Excel from MrExcel*. Jest autorem 65 książek na temat programu Mic-

rosoft 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 programistką i projektantką rozwiązań dla Microsoft Excel oraz autorką przeszło dziesięciu 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, dwóch koni oraz mnóstwa dzikich lisów, wiewiórek i królików.

Wprowadzenie

W tym rozdziale:

- Czy TypeScript jest zagrożeniem dla VBA?
- Co zawiera niniejsza książka?
- Wersje programu Excel
- Elementy specjalne i konwencje typograficzne
- Pliki kodu
- Errata i aktualizacje

Kiedy okazało się, że czas realizowania żądańprzez działy IT ciągle się wydłuża, użytkownicy programu Excel odkryli, że mogą samodzielnie tworzyć raporty potrzebne do prowadzenia przedsiębiorstwa, używając języka makr *Visual Basic for Applications* (VBA). VBA umożliwia uzyskanie niewiarygodnej efektywności 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.

Czy TypeScript jest zagrożeniem dla VBA?

Pierwsze pytania, jakie zapewne stawia sobie każdy Czytelnik, to: "Czy powinienem poświęcić czas na naukę VBA? Jak długo Microsoft będzie wspierać VBA? Czy język TypeScript wprowadzony dla wydania Excel Online zastąpi VBA?"

Odpowiedź brzmi – inwestycja w VBA będzie służyć nam dobrze co najmniej do roku 2049.

Ostatnia zmiana języka makr – z XLM na VBA – nastąpiła w roku 1993. Tym niemniej, XLM nadal jest obsługiwany w Excelu. Mamy do czynienia z sytuacją, gdy VBA jest pod każdym względem *lepszy* niż XLM, ale XLM nadal jest wspierany, blisko 30 lat później. Jeśli Microsoft kiedykolwiek zdecyduje się na przejście z VBA na TypeScript, możemy oczekiwać, że wsparcie dla VBA będzie kontynuowane w wersjach Excela dla Windows i Maca przez kolejne 30 lat.

W dzisiejszym uniwersum Excela mamy wersje działające w systemach Windows, w MacOS, na telefonach komórkowych systemu Android i iOS, a także w nowoczesnych przeglądarkach z wykorzystaniem Excel Online. W moim świecie przez 99% czasu używam Excela na komputerze Windows. Zapewne w 1% przypadków otwieram skoroszyty Excela na iPadzie. Jeśli jednak ktoś pracuje w środowisku mobilnym, w którym używa Excela w przeglądarce, funkcje UDF w języku TypeScript mogą być odpowiednie.

Jako wprowadzenie do funkcji UDF TypeScript dla Excela można przeczytać książkę Suata M. Ozgura *Excel Custom Functions Straight to the Point* (ISBN 978-1-61547-259-8).

Niemniej jednak wydajność TypeScript jest nadal fatalna. Jeśli nie potrzebujemy makr, które da się uruchamiać w Excel Online, wersja VBA naszych makr będzie działać (co najmniej) osiem razy szybciej, niż wersja TypeScript. Dla tych, którzy zamierzają uruchamiać Excela wyłącznie na platformach Mac lub Windows, VBA będzie językiem wyboru co najmniej przez następną dekadę.

Zagrożeniem dla Excel VBA są natomiast nowe narzędzia Excel Power Query, które można znaleźć w sekcji **Pobieranie i przekształcanie danych** (Get & Transform) karty **Dane** programu Excel dla Windows. Jeśli ktoś tworzy makra w celu czyszczenia importowanych danych, powinien rozważyć sprzątanie ich jednorazowo za pomocą Power Query, a następnie odświeżać zapytanie każdego dnia. Mam już mnóstwo opracowanych przebiegów Power Query, które wcześniej wymagały VBA. Jako wprowadzenie do Power Query, warto zapoznać się z książką *Master Your Data with Excel and Power BI: Leveraging Power Query to Get & Transform Your Task Flow* autorstwa Kena Pulsa i Miguela Escobara (ISBN 978-1-61547-058-7).

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 to BASIC, dlaczego nie wygląda znajomo?" ułatwia zrozumienie szalonej składni języka VBA. 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.

W rozdziale 5 "Formuły w stylu R1C1" rzecz jasna opisano działanie takich 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 i kolekcji" to omówienie tablic, klas i kolekcji, a w rozdziale 10 "Formularze użytkownika – wprowadzenie" opisano niestandardowe okna dialogowe, których można używaćdo zbierania informacji od użytkowników za pomocą programu Excel.

Potęga 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 oraz tabeli przestawnych. Narzędzia automatyzacji raportów intensywnie wykorzystują te koncepcje. W rozdziale 13 "Siła 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 zapytańdo stron sieci Web i automatyzowania działańw programie Word.

Materiały techniczne potrzebne do tworzenia aplikacji

W rozdziale 21 "Wykorzystanie Access w celu zapewnienia dostępu do danych 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 formularzy użytkownika" rozwinięta 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 "Wprowadzenie do tworzenia dodatków pakietu Office" to krótkie wprowadzenie w proces konstruowania własnych aplikacji TypeScript w programie Excel, a rozdział 28 "Nowości i zmiany w Excel 365" to podsumowanie zmian wprowadzonych w wersji Excel 365.

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 użyt-kowników. Pomimo tego, ankieta przeprowadzona wśród 8000 czytelników witryny MrExcel.com pokazała, że tylko 42% użytkowników (a jest to grupa ponadprzecięt-nych 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 w dowolnym rozdziale około 60% osób nie stosowało nigdy wcześniej tabel przestawnych, a jeszcze więcej 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 tabele przestawne, 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 zaniosła 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 mogłaby 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.

Wersje programu Excel

To siódme już wydanie książki *VBA i makra* zostało opracowane do współpracy z programem Excel 365, opublikowane do sierpnia 2021 roku. Poprzednie wydania tej książki uwzględniały kod dla wersji od Excel 97 po Excel 2019. W 80% obecny kod jest identyczny z kodem dla poprzednich wersji*.

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 rozdziału 23, który wykorzystuje interfejs Windows API, nie będzie działać w systemie macOS. Tym niemniej można stwierdzić, że ogólne koncepcje omawiane w tej książce stosują się także do Maców. Ogólną listę różnic w odniesieniu do macOS znaleźć można pod adresem *http://www.mrexcel.com/ macvba.html*. VBA Editor dla Maca nie umożliwia projektowania UserForms (rozdział

^{*} Ze względu na fakt, że w Polsce zazwyczaj mamy do czynienia z polską wersją programu Excel, nazwy funkcji, a także poleceń i elementów interfejsu podawane są w języku polskim. Na końcu książki zamieściliśmy tabele zawierające nazwy funkcji występujące w polskiej wersji programu oraz ich angielskie odpowiedniki. Nazwy funkcji użytych w formułach są automatycznie tłumaczone, gdy skoroszyt utworzony w wersji angielskiej zostanie otwarty w wersji polskiej bądź odwrotnie. Nie dotyczy to jednak komentarzy/opisów zawartych w plikach przykładowych, które pozostają w wersji oryginalnej, co widać również na zawartych w książce zrzutach ekranowych. Automatycznej konwersji podlegają również formaty dat oraz znak dziesiętny, którym w wersji polskiej jest przecinek, a w angielskiej kropka. Trzeba też pamiętać, że w formułach w wersji polskiej średnik. Uwagi te *nie dotyczą* jednak kodu w języku VBA – tu nazwy funkcji, separatory itp. pozostają bez zmian, gdyż wspomniana automatyczna konwersja dotyczy jedynie wyświetlania w interfejsie użytkownika, a nie rzeczywistej zawartości skoroszytu programu Excel (wszystkie przypisy pochodzą od redakcji wydania polskiego).

10). Istnieje także nadal nieusunięty bug, który utrudnia tworzenie makr obsługi zdarzeń (rozdział 7). Excel zgłasza błąd, gdy próbujemy wybierać z list rozwijanych widocznych u góry okna Code. Trzeba najpierw skopiować i wkleić pustą procedurę zdarzenia – wówczas te listy rozwijane będą działać.

Elementy specjalne i konwencje typograficzne

W książce używane są następujące konwencje typograficzne:

- *Kursywa* wskazuje nowe pojęcie podczas jego definiowania, specjalne wyróżnienie, obce słowa czy frazy oraz litery czy słowa używane jako słowa.
- Czcionka o stałej szerokości Wskazuje częśćkodu VBA, na przykład nazwę obiektu czy metody.
- Czcionka wytłuszczona bezszeryfowa Wskazuje nazwy elementów interfejsu użytkownika oraz informacje wprowadzane przez użytkownika.

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

Oprócz ramek Studium przypadku, zobaczymy równieżramki Uwaga, Wskazówka i Ostrzeżnie:

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

Wsкаzówka 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.



(
\sum	
- 1	

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 oraz dodatkowe uwagi od autorów. Pliki kodu można pobrać ze strony dostępnej pod adresem

http://microsoftpressstore.com/ExcelVBAMacros365/downloads.

Errata i aktualizacje

Dołożyliśmy wszelkich starań, aby zapewnić dokładność tej książki i dołączonych do niej materiałów dodatkowych. Aktualizacje – w formie listy zgłoszonych błędów i poprawek – są dostępne pod adresem:

http://microsoftpressstore.com/ExcelVBAMacros365/errata

Jeśli zauważysz błąd, którego nie ma na tej liście, zgłoś go korzystając z tej samej strony.

Dodatkowe informacje i wsparcie są dostępne pod adresem

http://www.MicrosoftPressStore.com/Support.

Należy zauważyć, że wsparcie dla oprogramowania i sprzętu oferowanego przez firmę Microsoft nie jest dostępne za pośrednictwem wymienionych adresów. Pomoc dotyczącą oprogramowania lub sprzętu firmy Microsoft można uzyskać pod adresem *http:// support.microsoft.com*.

ROZDZIAŁ 1

Zwiększanie możliwości programu Excel za pomocą języka VBA

W tym rozdziale:

- Początkowe przeszkody
- Poznawanie narzędzi: karta Deweloper
- Typy plików, dla których dopuszczane są makra
- Bezpieczeństwo makr
- Rejestrowanie, zapisywanie i uruchamianie makr
- Uruchamianie makra
- Działanie edytora Visual Basic
- Mankamenty rejestratora makr

Visual Basic for Applications (VBA) w połączeniu z Microsoft Excel jest zapewne najbardziej wydajnym spośród dostępnych narzędzi. VBA jest dostępny w komputerach 900 milionów użytkowników Microsoft Office, ale większość z nich nigdy nie spróbowała nawet wykorzystać mocy VBA w Excelu. Przy użyciu VBA można przyśpieszyć niemal każle zadanie wykonywane w Excelu. Jeśli na przykład regularnie tworzysz w Excelu serie wykresów prezentujących miesięczne wyniki, możesz sprawić, aby VBA wykonał to zadanie w ciągu kilku sekund.

Początkowe przeszkody

Istnieją dwie przeszkody utrudniające opanowanie programowania w VBA. Po pierwsze, rejestrator makr w Excelu nie jest daleki od doskonałości i nie generuje działającego kodu, który możnaby wykorzystywać jako model. Po drugie, dla wielu osób, które wcześniej poznały taki język programowania na jak BASIC, składnia języka VBA jest okropnie frustrująca.

Rejestrator makr nie działa!

Firma Microsoft zaczęła dominować na rynku arkuszy kalkulacyjnych w połowie lat 90-tych. Wprawdzie odniosła znaczący sukces, tworząc wydajny program arkusza kalkulacyjnego, na który łatwo mogliby przejść użytkownicy programu Lotus 1-2-3, jednak w odniesieniu do makr było zupełnie inaczej. Niemal dla każlego, kto sprawnie tworzył makra 1-2-3, próby rejestrowania makr w Excelu zazwyczaj kończyły się niepowodzeniem. Choć język programowania VBA ma nieporównywalnie większe możliwości, niż język makr programu Lotus 1-2-3, podstawową wadą było niewłaściwe działanie rejestratora makr przy domyślnych ustawieniach.

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 Exceli makro może działać dzisiaj, ale jutro już nie. W 1995 roku, kiedy spróbowałem zarejestrować moje pierwsze makro w Excelu, byłem mocno sfrustrowany takim działaniem programu. W tej książce pokażę trzy reguły, które pozwalają najbardziej efektywnie wykorzystywać rejestrator 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 zarejestruje odpowiednią sekwencję. W poprzedniej dekadzie, rejestrowany kod *mógł* działać w niektórych sytuacjach, ale zwykle nie działał poprawnie *we wszystkich* sytuacjach. Gdyby ktoś w firmie Microsoft był bardziej zainteresowany stworzeniem przydatnego rejestratora makr, rejestrowany kod mógłby być nieco bardziej ogólny, niż jest obecnie.

Spodziewaliśmy się, że makro można nagrać dowolną z pięciu dostępnych metod, a zarejestrowany kod będzie działał. Niestety obecnie, jeśli chcemy zastosować rejestrator makr, musimy często próbować rejestrowania makra siedmioma różnymi metodami, aż znajdziemy taki zestaw kroków, które rejestrują stabilnie działający kod.

Visual Basic nie przypomina języka BASIC

Blisko trzy dekady temu kod generowany przez rejestrator makr nie przypominał niczego, z czym się spotkałem wcześniej. Nazywano to "Visual Basic" (VB). Miałem przyjemność nauczenia się tuzina języków programowania w różnych okresach, ale ten dziwacznie wyglądający język był zupełnie nieintuicyjny i zdecydownie nie przypominał języka BASIC, któego uczyłem się jeszcze w szkole średniej.

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

3

z rejestratorem makr, który nie działał, i z językiem, którego nie rozumiałem. Trudno to nazwać korzystnym splotem wydarzeń.

Jednym z założeń które przyjąłem podczas pisania tej książki, jest to, że Czytelnik jest sprawnym użytkownikiem arkuszy kalkulacyjnych oraz że prawdopodobnie umie więcej, niż 90% osób w jego biurze. Ponadto założyłem, że nawet jeśli nie jesteś programist(k)ą, zapewne zetknąłeś się z językiem BASIC gdzieś na swojej drodze edukacyjnej. Jednakże znajomość tego języka nie jest wymaganiem – 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 podobna szansa, że nie był zadowolony z uzyskanych wyników.

Dobra wiadomość: poznawanie języka VBA nie jest trudne

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. Wszyst-kim Czytelnikom, którzy programowali już w języku BASIC, opiszę język VBA w taki sposób, by mogli łatwo przeanalizować zarejestrowany kod i zrozumieć jego działanie.

Dobra wiadomość: Excel plus VBA są warte wkładanego wysiłku

Zapewnie brak możliwości skutecznego rejestrowania makr w Excelu spowodował, że wiele osób poczuło się rozczarowanych firmą Microsoft. Trzeba jednak podkreślić, że możliwości języka VBA w programie Excel są ogromne. Wszystko (naprawdę wszystko), co można wykonać za pomocą interfejsu programu, można też niezwykle szybko osiągnąć używając języka VBA. Jeśli ktoś codziennie czy co tydzieńręcznie tworzy takie same raporty, język VBA w programie Excel znacznie uprości wykonywanie takich zadań

Autorzy tej książki zapewniają doradztwo w zakresie programu Excel od ponad 20 lat. W ramach tej pracy zautomatyzowaliśmy tworzenie raportów dla setek klientów. Te historie zazwyczaj są do siebie podobne: dział IT ma wielomiesięczną listę zaległości. 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. To zaś może okazać się bardzo żmudne. 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 pozostańcie ze mną – wyjaśnię kilka podstawowych zasad.

W tym 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 (jeszcze) zrozumiały, ale nie trzeba się tym przejmować. Celem tego rozdziału jest przedstawienie podstawowego problemu dotyczącego rejestratora makr. W rozdziale zaprezentowano również podstawowe elementy środowiska programistycznego Visual Basic.

Poznawanie narzędzi: karta Deweloper

Rozpoczniemy 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 Karta Deweloper dostępnia interfejs do uruchamiania i rejestrowania makr.

W grupie **Kod** karty **Deweloper** znajdują się ikony używane do rejestrowania i odtwarzania makr VBA:

- Visual Basic 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 zarejestruje wybranie komórki A4.

Bezpieczeństwo makr Ikona umożliwia dostęp do modułu Centrum zaufania, gdzie można zezwolić na uruchamianie makr lub tego zabronić na tym (lokalnym) komputerze.

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

Grupa **Formanty** karty Deweloper zawiera menu **Wstaw**, gdzie znajduje się szereg kontrolek, które można umieszczaćna 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 "Formularze użytkownika – wprowadzenie".

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

Typy plików, dla których dopuszczane są makra

Excel 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 arkuszy zawierających makra, ale można też zmienić domyślny typ pliku używany przez program Excel.

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 jest takie, by użytkownik nie musiał obawiać się złośliwych makr, mając plik .xlsx. 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 w celu obsługi tabel 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 ładowaćsię 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. Ponad-to 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 skoroszytu z włączoną obsługa 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 **Skoroszyt programu Excel z obsługą makr**. Kliknij **OK**.

Uwaga Chociaż zwykle nie boimy się używać makr, spotykałem 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 okrzyk króla Artura w filmie "Monty Python i Święty Grail": "You got me all worked up!"*. Również Gmail należący do Google dołączył do tego poglądu i odmawia prezentowania podglądu załączników wysyłanych w formacie .xlsm.

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

- Każdy skoroszyt starszego formatu (.xls) utworzony w ciągu minionych lat 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 dostęp do danych zawartych w arkuszu.

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

^{* &}quot;Zmusiłeś mnie do mnóstwa niepotrzebnej pracy".

Bezpieczeństwo makr

Po tym, jak wykorzystano makra VBA w programie Word 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 Excelu można globalnie zmodyfikować ustawienia zabezpieczeń lub kontrolować ustawienia makr dla wskazanych skoroszytów poprzez przechowywanie ich w zaufanej lokalizacji. Makra zostaną automatycznie włączone dla wszystkich skoroszytów 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, wykonaj poniższą procedurę:

- 1. Kliknij przycisk Bezpieczeństwo makr na karcie Deweloper.
- **2.** W okienku nawigacji z lewej strony okna **Centrum zaufania** kliknij **Zaufane lokalizacje**.
- **3.** Jeśli zaufane lokalizacje mają znajdowaćsię 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).
- 5. Kliknij przycisk Przeglądaj. Program Excel wyświetli okno dialogowe Przeglądaj.
- 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 P**odfoldery tej lokalizacji są także zaufane**.
- 8. Kliknij OK, by dodać folder do listy Zaufane lokalizacje.



.!.

RYSUNEK 1.2 Zarządzanie zaufanymi folderami w oknie Centrum zaufania.

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:. Z tego względu 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. Nazwy ustawień Niskie, Średnie, Wysokie i Bardzo wysokie, które znamy z programu Excela 2003, zostały zmienione

Aby uzyskać dostęp do ustawieńmakr, 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 makra języka VBA bez powiadomienia Ustawienie to blokuje uruchamianie wszystkich makr i jest przeznaczone dla osób, które nigdy nie zamierzają korzystać z makr. Ponieważ ta książka uczy, jak używać makr, nie będziemy używać tej opcji. Ustawienie to, ogólnie rzecz biorąc, jest odpowiednikiem opcji Bardzo wysokie z programu Excel 2003. W przypadku tego ustawienia mogą działać jedynie makra zapisane w zaufanych lokalizacjach.
- Wyłącz makra języka VBA z powiadomieniem Istotne słowa w nazwie tej opcji to "z powiadomieniem". Oznacza to, że użytkownikowi wyświetlone zostanie powiadomienie, kiedy otworzy plik zawierający makra, po czym może zdecydować, 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 ustawienie zalecane. W obszarze komunikatów wyświetlane jest

powiadomienie informujące o tym, że makra zostały wyłączone. Klikając tę opcję, użytkownik może włączyć zawartość (rysunek 1.3).

- Wyłącz makra języka VBA oprócz makr podpisanych cyfrowo Ustawienie to wymaga użycia narzędzia do tworzenia podpisów cyfrowych od jakiegoś dostawcy zaufania, takiego jak VeriSign. Jest to właściwy wybór dla osób zamierzających sprzedawać dodatki innym użytkownikom, ale jest trochę uciążiwym rozwiązaniem w przypadku pisania makr na własny użytek.
- Włącz makra języka VBA (niezalecane, może zostać uruchomiony potencjalnie 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.

OSTRZEŻENIE O ZABEZPIECZENIACH Makra zostały wyłączone.
 Włącz zawartość

Stosowanie opcji Wyłącz makra języka VBA z powiadomieniem

Zalecane jest stosowanie ustawienia **Wyłącz makra języka VBA z powiadomieniem**. Jeśli użyjemy 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 włączyć 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, Excel zapamięta, że w tym skoroszycie włączaliśmy makra. Przy kolejnym otwieraniu tego skoroszytu obsługa makr będzie automatycznie włączona.

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

Rejestrowanie, zapisywanie i uruchamianie 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).

Rejestrowanie makra		?	×
Nazwa makra:			
Makro1			
Klawisz <u>s</u> krótu: Ctrl+ Przechowuj makro w:			
Ten skoroszyt			~
<u>O</u> pis:			
	OK	An	uluj

Rysuneк 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 użyćnazwy **Makro1**, a nie **Makro 1**. Ponieważ makr może być bardzo dużo, warto stosować nazwy opisowe, na przykład nazwa FormatowanieRaportu jest znacznie bardziej przydatna niż Makro1.

W drugim polu okna dialogowego Rejestrowanie makra definiujemy klawisz skrótu. Jeśli w tym polu wpiszemy 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 przypisania klawiszy. Przykładowo, jeśli przypiszemy do makra kombinację Ctrl+C, program Excel uruchomi makro zamiast wykonać standardowe kopiowanie. 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**.

WKAZÓWKA 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 jako komentarz 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.

WKAZÓWKA 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 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 z paska narzędzi 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 skoroszycie 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). Zwróć 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). 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 po prawej stronie 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.** Grupę Makra raportów możesz przenieśćw 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.
- **10.** Kliknij **OK**, by zamknąć okno **Opcje programu Excel**. Na wybranej karcie wstąźki 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 skoroszycie makr osobistych, do makra można przypisać przycisk, który będzie się stale wyświetlał na pasku narzędzi Szybki dostęp. Jeśli makro jest zapisane w bieżącym skoroszycie, 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**.
- Jeśli makro ma być dostępne tylko wtedy, gdy jest otwarty bieżący skoroszyt, rozwińlistę 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.



Rysunek 1.5 Dodawanie przycisku makra do paska narzędzi Szybki dostęp

- 6. Kliknij OK, by zamknąć okno dialogowe Modyfikowanie przycisku.
- **7.** Kliknij **OK**, by zamknąć okno dialogowe **Opcje programu Excel**. Na pasku narzędzi Szybki dostęp pojawi się nowy przycisk.

Przypisywanie makra do kontrolki formularza, pola tekstowego lub kształtu

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

Aby dołączyć makro do kontrolki formularza na arkuszu:

- Na karcie Deweloper kliknij przycisk Wstaw, by wyświetlić listę rozwijaną tego przycisku. Na tej liście Excel udostępnia 12 kontrolek formularza i 12 kontrolek ActiveX. Na początku prezentowane są kontrolki formularza, a u dołu kontrolki ActiveX. Większość ikon na tej liście w części dotyczącej kontrolek ActiveX wygląda identycznie jak ikonki kontrolek 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. Aby wyjść z trybu edycji tekstu, należy kliknąć poza przyciskiem 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 wokół przycisku i wybierz polecenie Formatuj formant. Wyświetlone zostaje okno dialogowe Formatowanie formantu, składające się z siedmiu zakładek. Jeśli okno dialogowe 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.
- 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 dowolną 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 skoroszycie co makro. Makro można przypisać do dowolnego spośród tych obiektów

Działanie edytora Visual Basic

Narzędzie VB Editor umożliwia 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

^{*} Narzędzie VB Editor dostępne jest tylko w angielskiej wersji interfejsu użytkownika.

mamy trochę inny ekran, niż okno prezentowane na rysunku, ponieważpodczas 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 Excel nie wymaga deklarowania zmiennych. Osobiście preferuję to ustawienie, ponieważ pozwala ono oszczędzić 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, wciśniętą pomiędzy ikony 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 skoroszycie 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 "Formularze użytkownika – wprowadzenie".
- Moduły Podczas rejestrowania makra program Excel automatycznie tworzy moduł, w którym umieszcza kod. Większośćtworzonego 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 i kolekcji".

Okno Properties

Okno Properties umożliwia edycję właściwości różnych komponentów, takich jak arkusze, skoroszyty, moduły i kontrolki formularzy. Lista właściwości może byćróżna, 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

Załóżny, 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 InvDate (Data faktury), InvNbr (Numer faktury), RepNbr (Numer sprzedawcy), CustNbr (Numer klienta), ProdRevenue (Dochód produktu), ServRevenue (Dochód usługi) oraz ProdCost (Koszt produktu) (patrz rysunek 1.8).

invoice.txt - Notepad
File Edit Format View Help
InvDate, InvNbr, RepNbr, CustNbr, ProdRevenue, ServRevenue, ProdCost 6/05/2021, 123813, S82, C8754, 716100, 12000, 423986 6/05/2021, 123814,C4894, 224200, 0, 131243
6/05/2021,123815,543,C7278,277000,0,139208 6/05/2021,123816,554,C6425,746100,15000,350683 6/05/2021,123817,543,C6291,928300,0,488988
6/05/2021,123818,S43,C1000,723200,0,383069 6/05/2021,123819,S82,C6025,982600,0,544025
6/05/2021,123820,517,C8026,490100,45000,243808 6/05/2021,123821,543,C4244,615800,0,300579

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.



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. Klikniecie przycisku Zakończ, aby zaimportować plik.
- **12.** Naciśnięcie klawiszy Ctrl 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 ProdRevenue i jednocześnie pozostać w tej komórce.
- **17.** Przeciągnięcie uchwytu 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 nagłówków.
- **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**.

Domyślna nazwa makra w oknie **Rejestrowanie 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	?	\times
Nazwa makra:		
ImportFaktur		
Klawisz <u>s</u> krótu: Ctrl+ <u>P</u> rzechowuj makro w:		
Ten skoroszyt		\sim
Opis:		
Import pliku invoice.txt. Dodanie wiersz Formatowanie.	a podsumov	vania.
ОК	An	uluj

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

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 pomyłkę. Zarejestrowane makra działają szybko, ale nie warto przyglądać się, jak każlorazowo 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.