helion.pl
KSIĘGARNIA INTERNETOWA · WYDAWNICTWO

IDŹ DO
PRZYKŁADOWY ROZDZIAŁ
SPIS TREŚCI

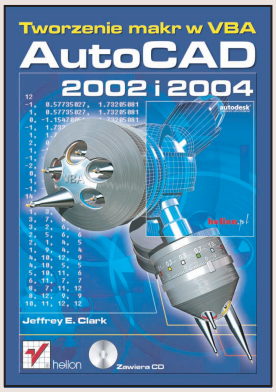
KATALOG KSIĄŻEK
KATALOG ONLINE
ZAMÓW DRUKOWANY KATALOG

TWÓJ KOSZYK
DODAJ DO KOSZYKA

CENNIK I INFORMACJE
ZAMÓW INFORMACJE O NOWOŚCIACH
ZAMÓW CENNIK

CZYTELNIA
FRAGMENTY KSIĄŻEK ONLINE

AutoCAD 2002 i 2004. Tworzenie makr w VBA



Autor: Jeffrey E. Clark
Tłumaczenie: Jacek Marzec
ISBN: 83-7197-861-8
Tytuł oryginału: VBA for AutoCAD 2002 Writing AutoCAD Macros
Format: B5, stron: 672

AutoCAD 2002 jest nie tylko najpopularniejszym na świecie narzędziem do projektowania i kreślenia: to także potężna, programowalna platforma służąca do automatyzacji złożonych zadań projektowych integrująca dane projektowe z firmowymi systemami informatycznymi i pozwalająca rozwiązać wiele problemów technicznych i biznesowych. Kluczem do tych możliwości jest język Visual Basic for Applications. Z kolei klucz do programowania za pomocą tego języka trzymasz właśnie w swoich rękach, jest nim książka „AutoCAD 2002. Tworzenie makr w VBA"

Jej autorowi, jak nikomu innemu, udało się powiązać kluczowe narzędzia AutoCAD-a z możliwościami języka VBA. Ukazując współdziałanie AutoCAD-a i VBA w warunkach rzeczywistej pracy pomagają czytelnikowi tworzyć praktyczne aplikacje.

W kolejnych etapach czytelnik:

- Przejmie kontrolę nad AutoCAD-em: opanuje podstawy automatyzacji, COM oraz programowanie zorientowane obiektowo
- Zacznie swobodnie poruszać się po interaktywnym środowisku projektowania VBA dla AutoCAD-a
- Wniknie w budowę pliku DXF, aby lepiej zrozumieć strukturę bazy danych rysunku AutoCAD-a
- Opanuje model obiektu AutoCAD-a: dokumenty, interfejs użytkownika, zarządzanie plikami, zbiory, obiekty i inne elementy
- Zautomatyzuje obiekty graficzne 2D i 3D, zewnętrzne odniesienia, elementy, bryły, wymiary oraz zdarzenia
- Zastosuje język VBA do sterowania obszarem papieru, rzutniami oraz wydrukiem
- Poszerzy możliwości AutoCAD-a o komunikację z innymi aplikacjami

Pogłębisz swoje umiejętności śledząc powstawanie kompletnej aplikacji, integrującej rysunek AutoCAD-a z bazą danych Microsoft Access za pomocą technologii DAO (Data Access Objects) Microsoftu. Książka „AutoCAD 2002. Tworzenie makr w VBA" zawiera również zwięzłe opisy zmiennych systemowych i wyliczeniowych AutoCAD-a, dziedziczenia obiektów oraz elementów graficznych. Jeśli jesteś gotów okiełznać całą moc zawartą w AutoCAD-zie 2002, jest to książka której szukałeś!

Wydawnictwo Helion
ul. Chopina 6
44-100 Gliwice
tel. (32)230-98-63
e-mail: helion@helion.pl

Spis treści

Przedmowa **13**

Część I Środowisko VBA AutoCAD-a **21**

Rozdział 1. Przejęcie kontroli nad AutoCAD-em **23**

Komponenty i technologia Automation 24
Podstawy 24
Spojrzenie ogólne 27
Szczegóły 28
Tworzenie rysunku za pomocą Visual Basica 31
Ustawianie dostępnych odniesień 32
Tworzenie procedury 32
Więcej na temat komponentów 33
Dziedziczenie w AutoCAD-zie 33
Tablice metod, właściwości i zdarzeń 35
Podsumowanie 37

Rozdział 2. Środowisko VBA **39**

Tworzenie i edycja makr 39
Okna projektowania 40
Okna usuwania błędów 42
Zarządzanie makrami i ich wykonywanie 47
Menadżer VBA 47
Okno dialogowe Open VBA Project 48
Okno dialogowe Macros 49
Wiersz poleceń AutoCAD-a 50
Podsumowanie 51

Rozdział 3. DXF: Klucz do struktury rysunku **53**

Format DXF 53
Sekcje Objects i Classes 55
Stałe kody grup 55
Zmienne kody grup 56
Znaczniki podkategorii 57
Oglądanie rysunku za pomocą VBA 58
Procedura GetSubclass odczytuje plik DWG 58
Procedura DXFsearch odczytuje plik DXF 61
AutoLISP a VBA 65
Zalety VBA 66
Program narzędziowy VBA 67
Podsumowanie 67

Rozdział 4. Elementy modelu obiektu 69
Interfejsy Automation 69
Obiekt Application 70
Zbiór Documents 71
Zbiory ModelSpace i PaperSpace 71
Zbiory tabelaryczne 72
Zbiory obiektowe 73
Obiekt Document i jego elementy 74
Obiekt Preferences 78
Menu i paski narzędzi użytkownika 79
Zbiór MenuBar 79
Zbiór MenuGroups 80
Podsumowanie 81

Część II Zastosowanie modelu obiektu AutoCAD-a 83

Rozdział 5. Dokumenty i interfejs użytkownika 85
Metody i właściwości 87
Zarządzanie plikami 88
Tworzenie pliku rysunkowego 88
Otwieranie, zapisywanie i zamykanie rysunku 89
Metoda WBlock 93
Eksportowanie i importowanie plików 93
Inne narzędzia zarządzania plikami 94
Interfejs użytkownika 95
Okna 95
Zoom 99
Widoki zdefiniowane przez użytkownika 100
Rzutnie 102
Hiperłącza 109
Zmienne wyliczeniowe AutoCAD-a 110
Tablice metod, właściwości i zdarzeń 111
Podsumowanie 111

Rozdział 6. Zbiory i obiekty 121
Obiekt Application 121
Metody 121
Właściwości 121
Obiekt Document 123
Metody 123
Właściwości 125
Zbiory 130
Zarządzanie zbiorami 131
Metody 131
Właściwości 134
Zbiory tabelaryczne 136
Zbiór Layers 136
Zbiór LayerStateManager 138
Zbiór Linetypes 140
Zbiór RegisteredApplications 141
Zbiór TextStyles 141
Lokalne układy współrzędnych 143
Tablice metod, właściwości i zdarzeń 145
Podsumowanie 145

Rozdział 7. Obiekty Utility 153
Zbiory wskazań, filtry i grupy 154
Zbiory wskazań 154
Filtry 159
Grupy 160
Obiekt Utility 161
Pobieranie danych 162
Konwersja danych 169
Dostęp do Internetu 174
Słowniki i obiekt XRecord 177
Tablice metod, właściwości i zdarzeń 179
Podsumowanie 179

Rozdział 8. Bloki i zewnętrzne odnośniki 183
Bloki 184
Korzystanie z bloków 185
Dostęp do odnośników bloków 188
Atrybuty i odniesienia do atrybutów 189
Właściwości 190
Blok zawierający dane 191
Obiekt MInsertBlock 195
Właściwości 195
Obiekt Database 197
Metoda CopyObjects 197
Właściwości obiektu Database 199
Zewnętrzne odnośniki 200
Metody 200
Redefinicja zewnętrznych odnośników 202
Arkusze 205
Tablice metod, właściwości i zdarzeń 205
Podsumowanie 205

Rozdział 9. Elementy 213
Wspólne metody i właściwości 213
Definicje 214
3DFace 214
Metody 214
Właściwość 215
3DPoly 217
Metoda 217
Właściwości 217
Arc 218
Właściwości 219
Circle 221
Właściwości 221
Ellipse 223
Właściwości 223
Hatch 225
Metody 225
Właściwości 226
LightWeightPolyline 228
Metody 228
Właściwości 229

Line 231
Właściwości 231
MLine 232
Właściwości 232
MText 235
Właściwości 236
Point 238
PolyfaceMesh 239
Właściwości 240
PolygonMesh (3DMesh) 243
Metoda 243
Właściwości 243
Polyline 248
Metody 248
Właściwości 248
Raster 250
Metoda 251
Właściwości 251
Ray 253
Właściwości 253
Region 255
Metoda 256
Właściwości 256
Shape 258
Metoda 258
Właściwości 258
Solid 260
Spline 261
Metody 262
Właściwości 263
Text 265
Właściwości 265
Trace 268
XLine 270
Właściwości 271
Metody, właściwości i zdarzenia związane z elementami graficznymi 271
Podsumowanie 272

Rozdział 10. Bryły 279

Box 280
Cone 281
Cylinder 283
EllipticalCone 284
EllipticalCylinder 284
ExtrudedSolid 285
ExtrudedSolidAlongPath 287
RevolvedSolid 289
Sphere 291
Torus 292
Wedge 293
Metody edycji brył 294
Właściwości masowe 298
Podsumowanie 301

Rozdział 11. Wymiary 303
Pojęcia 303
Menadżer stylów wymiarowania 304
Zmienne wymiarowania, style i nadpisywanie 306
Asocjatywność 306
Rodzaje wymiarów 307
Dim3PointAngular 307
DimAligned 309
DimAngular 310
DimDiametric 312
DimOrdinate 313
DimRadial 315
DimRotated 316
Linia odniesienia 317
Tolerancja 320
Właściwości wymiarowania 322
Linie i strzałki 323
Tekst 326
Dopasowanie 326
Jednostki podstawowe 327
Jednostki dodatkowe 328
Tolerancje 329
Inne właściwości wymiarowe 330
Zmienne wymiarowe niezwiązane z właściwościami 332
Tablice metod, właściwości i zdarzeń 333
Podsumowanie 338

Rozdział 12. Edycja 339
Metody związane z edycją 339
Array 340
Copy 342
Explode 343
GetBoundingBox 344
IntersectWith 345
Mirror 346
Move 348
Offset 348
Rotate 349
ScaleEntity 350
TransformBy 351
Undo 353
Właściwości informacyjne 354
Dostęp do wiersza poleceń 359
Inne metody, właściwości i procedury związane z edycją 360
Podsumowanie 361

Rozdział 13. AutoCAD i zdarzenia 363
Zdarzenia obiektu Application 363
Zdarzenia poziomu aplikacji 365
Zdarzenia obiektu Document 366
Procedury zdarzeń obiektu AcadDocument 367
Zdarzenia poziomu dokumentu 368
Zdarzenia obiektu Object 369
Ograniczenia obsługi zdarzeń 371
Podsumowanie 372

Rozdział 14. Formularze i kontrolki 373
Zdarzenia dotyczące poleceń 376
Zdarzenia dotyczące formularzy 378
Kolejność zaznaczania 379
Narzędzie Relative 380
Moduł Relative 381
Formularz frmRelative 386
Metody i właściwości kontrolek 392
Podsumowanie 392

Rozdział 15. Obszar papieru i plotowanie 397
Interfejs plotowania 397
Tabele stylów plotowania 397
Ustawienia plotera 397
Arkusze obszaru papieru 398
Ustawienia systemowe 398
Obiekty PlotConfiguration oraz Layout 399
Ustawienia wydruku 399
Arkusze 402
Wspólne metody i właściwości 402
Rzutnie obszaru papieru 410
Metody 412
Właściwości 413
Obiekt Plot 414
Metody 414
Właściwości 415
Projekt BatchPlot 415
Moduł FrmPlotFiles 416
Moduł BatchPlot 424
Metody i właściwości drukowania 428
Podsumowanie 434

Rozdział 16. Ustawienia 435
Style programowania 436
Pliki 436
Metody 436
Właściwości 437
Ekran 438
Wydajność wyświetlania 439
Rozdzielczość wyświetlania 440
Elementy arkusza 440
Elementy okna 441
Otwieranie i zapisywanie plików 443
Zewnętrzne odnośniki 443
Środki ochrony pliku 444
Otwieranie i zapisywanie plików 444
Aplikacje ObjectARX 445
Wydruk 445
Standardowe ustawienia wydruku dla nowych rysunków 445
Standardowe zachowanie stylów wydruku dla nowych rysunków 446
Ogólne ustawienia wydruku 447
System 447
Opcje połączenia baz danych 448
Opcje ogólne 448

Parametry użytkownika....449
AutoCAD DesignCenter....450
Hiperłącze....450
Ustawienia szerokości linii....451
Właściwości sortowania obiektów....451
Priorytet przy wprowadzaniu współrzędnych....452
Standardowe zachowanie Windows....452
Pomoce rysunkowe....453
AutoSnap....453
AutoTrack....454
Wybór....455
Uchwyty....455
Tryby wyboru oraz wskaźnik zbioru wskazań....456
Profile....456
Metody....457
Metody i właściwości związane z ustawieniami....458
Podsumowanie....458

Rozdział 17. Menu....463

Menu użytkownika i paski narzędziowe....464
Spojrzenie na strukturę menu....465
Metody....467
Właściwości....470
Menu rozwijane....472
Paski narzędzi....474
Zapisywanie modyfikacji....477
Tablice metod i właściwości....477
Podsumowanie....477

Część III Komunikacja z innymi aplikacjami i Internetem....481

Rozdział 18. Rozszerzalność....483

Dane dodatkowe....483
Metody....484
Wykorzystanie polilinii i danych dodatkowych....488
Interfejs VBA....499
Metody....499
Właściwości....500
Zdalne uruchamianie VBA....501
Status braku dokumentów....502
Aplikacje ARX....502
Metody....502
Podsumowanie....503

Rozdział 19. Projekt przestrzeni biurowej....505

Baza danych Microsoft Access....505
Tabele, klucze oraz ograniczenia integralności....506
Tworzenie okna dialogowego użytkownika....507
Tworzenie zapytań w tle okna dialogowego....509
Interfejs AutoCAD — Access....513
Procedura cbo_Group_Change....516
Procedura lstDepartment_AfterUpdate....518
Procedura lstProgDetail_Click....520
Przydział powierzchni....520
Podsumowanie....524

Rozdział 20. Model obiektu DWF 525
Drawing Web Format 525
Struktura pliku 526
Kody operacji 527
Współrzędne logiczne 528
Aplikacja odczytująca Excela 528
Moduł XLocate 529
Formularz XLdwf 530
Opis obiektu Whip! 538
Metody i właściwości ogólne 538
Inne zbiory i obiekty 546
Wbudowane zdarzenia aplikacji Whip! 547
Metody i właściwości Whip! 548
Podsumowanie 551

Dodatki 553

Dodatek A Zmienne systemowe AutoCAD-a 2002 555

Dodatek B Zmienne systemowe AutoCAD-a 2004 583

Dodatek C Wielkości wyliczeniowe (enum) 613

Dodatek D Dziedziczenie w obiektach 621

Dodatek E Program narzędziowy formatowania pliku DXF 625

Skorowidz 643

Rozdział 12.

Edycja

W rozdziale tym przyjrzymy się kilku wbudowanym w VBA funkcjom AutoCAD-a związanym z edycją. Odejdziemy tu od naszej wcześniejszej praktyki pisania samowystarczalnych przykładów, czyli tworzenia pewnego elementu i następnie zmieniania go w pewien sposób w tej samej procedurze. Aby zastosować omawiane metody edycji, tak jak robi się to zazwyczaj, utworzymy obiekty za pomocą procedur bazujących na listingach z wcześniejszych rozdziałów, a następnie zastosujemy metody zbioru `SelectionSet`, by wskazać elementy i zmodyfikować je.

Użyjemy sześciu tego typu procedur: `Add3DPoly`, `AddEllipse`, `AddHatch`, `AddLine`, `AddText` oraz `Boole`. (Nazwy tych procedur zaznaczyliśmy w listingach tego rozdziału kursywą). Procedury `AddEllipse`, `AddLine`, oraz `AddText` zostały uproszczone dla celów tego rozdziału, a ich kod umieściliśmy na końcu tego rozdziału. Dodajmy jeszcze, że kody tych listingów, jak również wszystkich innych zawartych w tej książce, można ściągnąć ze strony internetowej wydawnictwa Prentice Hall pod adresem *www.phptr.com*.

W związku z tym, że często stosujemy zbiory wskazań, a w listingach zazwyczaj używamy nazwy `newSelSet`, zbiór wskazań o określonej nazwie jest wymazywany na końcu każdego listingu. Jeśli z jakiegoś powodu dana procedura nie zadziała poprawnie, należy wówczas ręcznie wymazać zbiór `SelectionSet`, któremu nadano określoną nazwę; w przeciwnym razie jeśli spróbujemy użyć ponownie tej samej nazwy, AutoCAD wygeneruje komunikat o błędzie. W tym celu można zastosować podprogram `SelSetDelete` z rozdziału 7. (patrz listing 7.6). (Jak wspomnieliśmy w rozdziale 7., można też ewentualnie opróżnić zbiór wskazań i ponownie użyć go za pomocą metody `Clear`).

Metody związane z edycją

Najczęściej używane polecenia AutoCAD-a związane z edycją, takie jak `Array`, `Copy`, `Explode`, `Mirror`, `Move`, `Offset`, `Rotate`, `Scale` oraz `Undo`, mają swoje odpowiedniki w metodach VBA. Polecenia `Extend` i `Trim` nie posiadają bezpośrednich odpowiedników, ponieważ funkcje te wykonuje się przez zmianę wektorów definiujących modyfikowane elementy. Istnieją również trzy metody edycyjne, nieposiadające odpowiedników w interfejsie użytkownika. Są to: `GetBoudingBox`, `IntersectWith` oraz `Transform`. Jak będziemy mogli się przekonać, ostatnia z nich posiada wyjątkowo duże możliwości.

Array

Metoda Array ma dwie opcje: polar (tablica biegunowa) i rectangular (tablica prostokątna), podobnie jak odpowiadające jej polecenie AutoCAD-a. Opcja biegunowa służy do kopiowania wybranych obiektów określoną ilość razy, rozmieszczając je jednocześnie na okręgu wokół podanego punktu środkowego. Natomiast tablica prostokątna tworzy układ skopiowanych elementów ułożonych w rzędy i kolumny o określonej odległości od obiektu oryginalnego, który definiuje jeden z narożników tablicy.

Metoda ArrayPolar podczas kopiowania automatycznie obraca każdy z obiektów, orientując go tak, że jego oś symetrii przechodzi przez punkt środkowy tablicy. Pod tym względem metoda ta różni się od odpowiadającej jej komendy AutoCAD-a, która umożliwia wybór między obrotem obiektu a kopiowaniem bez obrotu.

ArrayPolar (metoda ogólna (39)). ReturnValue jako tablica Variant zmiennych Object =object.ArrayPolar (NumberOfObjects jako Integer [> 1, tylko wprowadzanie], AngleToFill jako Double [tylko wprowadzanie], CenterPoint jako Variant [tylko wprowadzanie]). Kąt należy podać w radianach; wartości ujemne wskazują na obrót w kierunku przeciwnym do ruchu wskazówek zegara, natomiast zero powoduje błąd.

Listing 12.1 rozpoczyna się od wywołania procedury AddEllipse służącej do utworzenia obiektu, który będziemy potem edytować. Możemy tutaj zastosować dowolny element AutoCAD-a lub grupę wskazanych elementów. Natomiast zbiór wskazań SelectionSet tworzony jest za pomocą metody narzędziowej SelectOnScreen.

Listing 12.1. *Tworzenie biegunowej tablicy obiektów*

```
Sub ArrayPolar()
    AddEllipse
    Dim newSSet As AcadSelectionSet
    Set newSSet = ThisDrawing.SelectionSets.Add("newSelSet")
    newSSet.SelectOnScreen

    Dim NumObj As Integer
    Dim Fangle As Double
    Dim CtrPnt(0 To 2) As Double
    NumObj = 6
    Fangle = 3.14
    CtrPnt(0) = 3: CtrPnt(1) = 3: CtrPnt(2) = 0#

    Dim Object As AcadEntity
    Dim NewObj As Variant
    For Each Object In newSSet
        NewObj = Object.ArrayPolar(NumObj, Fangle, CtrPnt)
    Next Object

    ZoomAll
    ThisDrawing.SelectionSets.Item("newSelSet").Delete
End Sub
```

Następnie deklarujemy i nadajemy wartości kilku obiektom, które mają zostać skopiowane do tablicy, jak również określamy kąt dopełnienia (w radianach) oraz współrzędne punktu środkowego. Deklarujemy również nowy obiekt AddEntity, który jest kojarzony

dynamicznie, ponieważ musi on być w stanie przechować dowolny rodzaj obiektu dodanego do zbioru wskazań. Po zadeklarowaniu zmiennej typu `Variant`, służącej do przechowywania nowo utworzonych obiektów, wykorzystujemy pętlę `For Each...Next`, aby do każdego obiektu zastosować metodę `ArrayPolar`. Na koniec za pomocą metody `ZoomAll` wykonujemy zmianę wielkości wyświetlanego obszaru, aby pokazać całą tablicę.

ArrayRectangular (metoda ogólna (39)). `ReturnValue` jako tablica `Variant` = `object.ArrayRectangular` (`NumberOfRows` jako `Integer` [> 1, tylko wprowadzanie], `NumberOfColumns` jako `Integer` [> 1, tylko wprowadzanie], `NumberOfLevels` jako `Integer` [tylko wprowadzanie], `DistBetweenRows` jako `Double` [tylko wprowadzanie], `DistBetweenColumns` jako `Double` [tylko wprowadzanie], `DistBetweenLevels` jako `Double` [tylko wprowadzanie]). Oryginalny obiekt, z którego tworzona jest tablica, zajmuje w niej lewy dolny narożnik; natomiast ujemne parametry dotyczące kolumn i rzędów oznaczają, że tablica budowana jest w dół i na lewo.

Metoda `ArrayRectangular` działa we wszystkich trzech osiach. Użytkownik podaje liczbę rzędów, kolumn i poziomów (dla osi *Z*) łącznie z odstępami omiędzy nimi. Natomiast znak użyty przy wprowadzaniu każdego z odstępów oznacza kierunek w osiach *X*, *Y* i *Z*.

Ustawienia w listingu 12.2 są bardzo podobne do tych z listingu 12.1. Natomiast do utworzenia obiektu testowego możemy użyć zarówno procedury `AddEllipse`, jak też wskazać obiekty z dowolnego rysunku. Najpierw deklarujemy wymagane zmienne i nadajemy wartości ilościom elementów oraz odległościom między rzędami, kolumnami i poziomami. Potem przeprowadzamy iterację zbioru wskazań za pomocą metody`ArrayRectangular`, aby utworzyć nowy obiekt.

Listing 12.2. *Tworzenie prostokątnej tablicy obiektów*

```
Sub ArrayRectangular()
    AddEllipse
    Dim newSSet As AcadSelectionSet
    Set newSSet = ThisDrawing.SelectionSets.Add("newSelSet")
    newSSet.SelectOnScreen

    Dim RowNum As Long
    Dim ColNum As Long
    Dim LevNum As Long

    Dim RowDis As Double
    Dim ColDis As Double
    Dim LevDis As Double
    RowNum = 5: RowDis = 3
    ColNum = 5: ColDis = 3
    LevNum = 3: LevDis = 2

    Dim Object As AcadEntity
    Dim NewObj As Variant
    For Each Object In newSSet
        NewObj = Object.ArrayRectangular(RowNum, ColNum, LevNum, RowDis, ColDis, LevDis)
    Next Object

    ZoomAll
    ThisDrawing.SelectionSets.Item("newSelSet").Delete
End Sub
```

Copy

Aby za pomocą metody Copy przenieść nowo utworzony obiekt w inne miejsce, należy zastosować ją wraz z metodą Move, nazwa Copy wprowadza tu w błąd. Oprócz utworzenia zbioru wskazań obiektów, które chcemy skopiować, w ustawieniach należy dodać także dwa punkty służące jako wektor przemieszczenia. Funkcję tę spełniają dwa pierwsze paragrafy listingu 12.3.

Copy (metoda ogólna (39)). ReturnValue jako Object = object.Copy. (Obiekt skopiowany zostaje w tym samym miejscu).

Procedura ta umożliwia wybranie dowolnej liczby obiektów rysunkowych. W operacji kopiowania wykorzystujemy pętlę For...Next, służącą do iteracji zbioru wskazań, aby skopiować, przesunąć i zmienić kolor nowych obiektów. Aby zrealizować te czynności, musimy najpierw zadeklarować zmienną pętli (i) oraz nadać rozmiary tablicy Variant (newObj), bazując na właściwości Count należącej do obiektu SelectionSet. W ten sposób korzystamy z tej właściwości dwukrotnie, ale nie musimy deklarować dwóch zmiennych: jednej do zliczania elementów oraz obiektu AcadEntity dla składników zbioru wskazań. Z kolei w listingu dotyczącym metody Explode (patrz listing 12.4) do wykonania podobnej czynności używamy alternatywnej konstrukcji For Each...Next. Zastosowanie tej składni jest szybsze pod wieloma względami.

Listing 12.3. *Prosta operacja kopiowania*

```
Sub Copy()
    Dim newSSet As AcadSelectionSet
    Set newSSet = ThisDrawing.SelectionSets.Add("newSelSet")
    newSSet.SelectOnScreen

    Dim P1(0 To 2) As Double
    Dim P2(0 To 2) As Double
    P1(0) = 0: P1(1) = 0: P1(2) = 0
    P2(0) = 3: P2(1) = 4: P2(2) = 0
    MsgBox "Zastosuj wektor przesunięcia..."

    Dim i As Integer
    ReDim newObj(0 To newSSet.Count - 1) As Variant
    For i = 0 To newSSet.Count - 1
        Set newObj(i) = newSSet.Item(i).Copy
        NewObj(i).Move P1, P2
        NewObj(i).Color = acGreen
    Next i
    ThisDrawing.SelectionSets.Item("newSelSet").Delete
End Sub
```

CopyObjects (metoda obiektu Document, Database). ReturnValue jako tablica Variant zmiennych typu Object = object.CopyObjects. (Objects jako tablica Variant zmiennych typu Object [tylko wprowadzanie], Owner jako pojedynczy obiekt Variant [opcjonalnie, tylko wprowadzanie], IDPairs jako tablica Variant zmiennych typu IDPair [opcjonalnie, tylko wprowadzanie]).

Metoda CopyObjects powiela kilka obiektów należących do tego samego dokumentu lub do różnych dokumentów, używając w tym przypadku opcjonalnego parametru Owner. Jeśli nie podano tego parametru, wówczas nowe obiekty tworzone są w tym samym dokumencie co obiekt źródłowy. Tablicę Objects tworzymy przy użyciu podstawowych obiektów, które mają zostać skopiowane, natomiast wszelkie drugorzędne obiekty, podległe lub związane z podstawowymi, również zostaną skopiowane.

Metodę CopyObjects przedstawiliśmy już w rozdziale 8. przy okazji omawiania obiektu Database, a jej zastosowanie można było zobaczyć w listingu 8.10. Natomiast informacje dotyczące procesu kopiowania i przenoszenia zwracane są w tablicy obiektów IDPair, jeśli były one zawarte w odwołaniu do metody CopyObjects. Obiekt IDPair posiada kilka właściwości.

IsCloned. object.IsCloned jako Boolean (True = obiekt źródłowy został sklonowany, False = obiekt źródłowy nie został sklonowany, tylko do odczytu).

IsOwnerXlated. object.IsOwnerXlated jako Boolean (True = obiekt będący właścicielem został przeniesiony z jednej bazy danych do innej, False = obiekt będący właścicielem istnieje już w nowym rysunku i nie został skopiowany, tylko do odczytu).

IsPrimary. object.IsPrimary jako Boolean (True = obiekt źródłowy jest składnikiem podstawowego zbioru obiektów, False = pochodny obiekt źródłowy jest własnością podstawowego zbioru obiektów, tylko do odczytu).

Key. object.Key jako Long (tylko do odczytu) zwraca identyfikator obiektu źródłowego.

Value. object.Value jako Long (tylko do odczytu) zwraca identyfikator nowego, skopiowanego obiektu.

Explode

Metoda Explode rozbija złożony obiekt na jego elementarne składniki. W przeciwieństwie do odpowiadającego jej polecenia AutoCAD-a, metoda ta nie rozbija wymiarów, multilinii oraz większości obiektów 3D. Jeśli zastosujemy ją do obiektów, których nie dotyczy, system wygeneruje komunikat o błędzie object doesn't support this property or method („obiekt nie obsługuje tej własności lub metody"). Obiekty obsługujące metodę Explode zostały wymienione poniżej. Należy także pamiętać o wyraźnym wymazaniu rozbitego obiektu, który pozostaje po utworzeniu nowego zbioru jego elementów składowych.

Explode. ReturnValue jako tablica Variant zmiennych typu Object =object.Explode Nazwa obiektu(ów), który ma zostać rozbity. Dostęp do pojedynczych elementów zawartych w blokach możliwy jest za pomocą metody Item, nie jest zatem konieczne rozbijanie bloku w celu jego edycji. Metoda Explode obsługiwana jest przez obiekty: 3DPoly, BlockRef, LightWeightPolyline, MInsertBlock, PolygonMesh, Polyline oraz Region.

Listing 12.4 rozpoczyna się od utworzenia obiektu testowego 3DPolyline oraz prośby o wybranie zbioru wskazań. Jeśli chcemy, możemy wybrać również kilka obiektów. Natomiast drugi paragraf tej procedury przeprowadza faktyczne rozbicie. Deklarujemy również kilka zmiennych, łącznie z sCount, która dostosowana jest do liczby elementów w zbiorze wskazań (właściwość Count zwraca liczbę o jeden większą od sCount, ponieważ numeracja rozpoczyna się od zera).

Pętla `For Each...Next`, zawarta w tej procedurze, przeprowadza iterację zbioru wskazań. Potrzebna jest nam także dodatkowa zmienna (`i`), aby śledzić pozycję tablicy `ExpObj`. Należy jednak wspomnieć, że w tym przypadku bardziej efektywna byłaby tradycyjna pętla `For...Next`. Kolejną czynnością jest usunięcie wszystkich obiektów ze zbioru wskazań po tym, jak zostały rozbite.

Trzeci paragraf tego listingu przeprowadza po prostu iterację elementów zbioru wskazań oraz zestawów rozbitych obiektów, zmieniając ich kolor na zielony, a następnie przywracając domyślny kolor warstwy po to, abyśmy mogli zobaczyć je na ekranie. (Aby przetestować tę metodę, nie wskażemy zapewne dużej ilości obiektów).

Listing 12.4. *Rozbicie kilku obiektów*

```
Sub Explode()
    Add3DPoly
    Dim newSSet As AcadSelectionSet
    Set newSSet = ThisDrawing.SelectionSets.Add("newSelSet")
    newSSet.SelectOnScreen

    Dim Object As AcadEntity
    Dim i As Integer
    Dim j As Integer
    Dim sCount As Integer
    sCount = newSSet.Count - 1
    ReDim ExpObj(0 To sCount) As Variant
    For Each Object In newSSet
        ExpObj(i) = Object.Explode
        Object.Delete
        i = i + 1
    Next Object

    For i = 0 To sCount
    MsgBox "Element SelSet=" & i & " LBound=" & LBound(ExpObj(i)) & " UBound=" & _
        UBound(ExpObj(i))
        For j = LBound(ExpObj(i)) To UBound(ExpObj(i))
            ExpObj(i)(j).Color = acGreen
            ExpObj(i)(j).Update
            MsgBox "Rozbity obiekt " & j & ": " & _
                    ExpObj(i)(j).ObjectName
            ExpObj(i)(j).Color = acByLayer
            ExpObj(i)(j).Update
       Next j
    Next i
    ThisDrawing.SelectionSets.Item("newSelSet").Delete
End Sub
```

GetBoundingBox

Metoda ta nie ma odpowiednika wśród poleceń AutoCAD-a. Jednak wszystkie elementy posiadają pewną ramkę ograniczającą, którą wykorzystywaliśmy, na przykład opisując geometrię brył w rozdziale 10. W listingu przedstawionym w tym rozdziale wykorzystujemy procedurę `Boole`, która tworzy złożony obiekt przez połączenie dwóch brył. Procedura z listingu 12.5 przeznaczona jest do przetwarzania jednocześnie tylko jednego obiektu. Jeśli więc wskażemy większą liczbę obiektów, przetworzony zostanie tylko pierwszy z nich.

GetBoundingBox (metoda ogólna (39)). `ReturnValue` jako `Object` =`object.GetBoundingBox`. `MinPoint` jako `Variant` (tylko odczyt), `MaxPoint` jako `Variant` (tylko odczyt).

W listingu 12.5 deklarujemy dwie zmienne `Variant` obsługujące dolną i górną granicę ramki ograniczającej. Parametry te zwracane są przez metodę `GetBoundingBox` jako trójelementowa tablica zmiennych `Double` oraz wyświetlane na ekranie za pomocą okna dialogowego. Natomiast w końcowym paragrafie stosujemy metodę `AddPoint`, aby w tych miejscach umieścić widoczne punkty. W ten właśnie sposób wykonywaliśmy rysunki w rozdziale 10.

Listing 12.5. *Ustalenie punktów obszaru ograniczającego*

```
Sub GetBoundingBox()
    Boole
    Dim newSSet As AcadSelectionSet
    Set newSSet = ThisDrawing.SelectionSets.Add("newSelSet")
    newSSet.SelectOnScreen

    Dim eMin As Variant
    Dim eMax As Variant
    NewSSet.Item(0). GetBoundingBox eMin, eMax
    MsgBox "Punkty definiujące obszar graniczny: " & eMin(0) & "," & eMin(1) & "," & _
        eMin(2) & " (min) " & eMax(0) & "," & eMax(1) & "," & eMax(2) & " (maks) "

    Dim aPoint As AcadPoint
    Set aPoint = ThisDrawing.ModelSpace.AddPoint(eMax)
    Set aPoint = ThisDrawing.ModelSpace.AddPoint(eMin)
    ThisDrawing.SetVariable "PDMODE", 34
    ThisDrawing.SelectionSets.Item("newSelSet").Delete
End Sub
```

IntersectWith

Jest to druga metoda związana z edycją, która nie posiada odpowiednika wśród poleceń AutoCAD-a. Zwraca ona punkty, w których przecinają się dane obiekty. Zadaniem procedury z listingu 12.6 jest podanie punktów przecięcia tylko dwóch pierwszych wskazanych obiektów.

IntersectWith. `ReturnValue` jako tablica `Variant` zmiennych typu `Double` =`object.IntersectWith` (`IntersectObject` [dowolny obiekt rysunkowy, tylko wprowadzanie], `ExtendOption` jako `Integer` [tylko wprowadzanie, enum `acExtendOption`]). Metoda ta służy do podania punktów, w których przecinają się wskazane obiekty rysunkowe (wszystkie z wyjątkiem `PViewport` i `PolygonMesh`). Natomiast parametr `ExtendOption` określa, który z przecinających się elementów ma zostać przedłużony do miejsca ich teoretycznego przecięcia. Obowiązujące wartości wielkości wyliczeniowej `acExtendOption` są następujące:

(0) `acExtendNone`,

(1) `acExtendThisEntity`,

(3) `acExtendOtherEntity`,

(4) `acExtendBoth`.

Procedura z listingu 12.6 tworzy linię przecinającą w dwóch punktach elipsę, a następnie wyświetla kursor wyboru. Na początku deklarujemy tablicę `Variant`, przeznaczoną do przechowywania punktów przecięcia (`IntPts`), która zapełniana jest za pomocą pierwszego zbioru wskazań. Pętla w drugim paragrafie procedury wyświetla na ekranie punkty przecięcia, które przypisywane są do kolejnych trójpunktowych grup tablicy `IntPts`.

Listing 12.6. *Znajdowanie punktów przecięcia*

```
Sub IntersectWith()
    AddEllipse
    AddLine
    ZoomAll
    Dim newSSet As AcadSelectionSet
    Set newSSet = ThisDrawing.SelectionSets.Add("newSelSet")
    newSSet.SelectOnScreen

    Dim IntPts As Variant
    IntPts = newSSet.Item(0).IntersectWith(newSSet.Item(1), acExtendNone)
    Dim i As Integer
    Dim j As Integer
    Dim Msg As String
    For i = LBound(IntPts) To UBound(IntPts) Step 3
        MsgBox = "Punktem przecięcia[" & j & "] jest: " & IntPts(i) & "," & IntPts(i + _
            1) & "," & IntPts(i + 2)
        MsgBox Msg, , "Przykład IntersectWith"
        j = j + 1
    Next i
    ThisDrawing.SelectionSets.Item("newSelSet").Delete
End Sub
```

Mirror

Dwie metody VBA związane z lustrzanym odbiciem odzwierciedlają odpowiadające im polecenia interfejsu użytkownika w wersji 2D i 3D. Każda z nich może być zastosowana do dowolnego elementu AutoCAD-a.

Mirror (metoda ogólna (39)). `ReturnValue` jako odbity element typu `Object` = `object.Mirror` (`Point1` jako `Variant` [tylko odczyt], `Point2` jako `Variant` [tylko odczyt]. Metoda ta służy do lustrzanego odbijania płaskich obiektów względem wskazanej osi. Zmienna systemowa `MIRRTEXT` steruje odbijaniem tekstu (0 = tekst pozostaje czytelny, `False` = tekst odbity dosłownie).

W listingu 12.7 wykorzystaliśmy obiekty `Ellipse` i `Line` do zaprezentowania lustrzanego odbicia względem linii zdefiniowanej za pomocą dwóch punktów. Po wykonaniu tej czynności kolor odbitego obiektu zostaje zmieniony na czerwony. Procedura z omawianego listingu pozwala na wybranie wielu elementów. Należy także dodać, że zarówno w przypadku wersji 2D, jak i 3D, jeśli chcemy wymazać oryginalny obiekt, musimy osobno zastosować metodę `Delete`.

Listing 12.7. *Lustrzane odbicie w płaszczyźnie X-Y*

```
Sub Mirror()
    AddEllipse
```

```
    AddLine
    ZoomAll
    Dim newSSet As AcadSelectionSet
    Set newSSet = ThisDrawing.SelectionSets.Add("newSelSet")
    newSSet.SelectOnScreen

    Dim P1(0 To 2) As Double
    Dim P2(0 To 2) As Double
    P1(0) = 0: P1(1) = 4.25: P1(2) = 0
    P2(0) = 4: P2(1) = 4.25: P2(2) = 0
    MsgBox "Odbij symetrycznie zbiór wskazań..."

    Dim Object As AcadEntity
    Dim NewObj As Variant
    For Each Object In newSSet
        Set NewObj = Object.Mirror(P1, P2)
        NewObj.Color = acRed
    Next Object

    ZoomAll
    ThisDrawing.SelectionSets.Item("newSelSet").Delete
End Sub
```

Mirror3D (metoda ogólna (39)). `ReturnValue` jako odbity element typu `Object` =`object.Mirror3D` (`Point1` jako `Variant` [tylko odczyt], `Point2` jako `Variant` [tylko odczyt], `Point3` jako `Variant` [tylko odczyt]). Metoda służy do lustrzanego odbijania brył względem wskazanej płaszczyzny.

Procedura przedstawiona w listingu 12.8 pozwala na lustrzane odbicie jedynie pierwszego wskazanego elementu. W naszym przypadku elementem tym jest obiekt `Boole`, odbijany symetrycznie względem płaszczyzny zdefiniowanej za pomocą trzech punktów. Po wykonaniu operacji odbicia dodatkowo zmieniany jest kolor nowego elementu.

Listing 12.8. *Lustrzane odbicie w przestrzeni trójwymiarowej*

```
Sub Mirror3D()
    Boole
    Dim newSSet As AcadSelectionSet
    Set newSSet = ThisDrawing.SelectionSets.Add("newSelSet")
    newSSet.SelectOnScreen

    Dim P1(0 To 2) As Double
    Dim P2(0 To 2) As Double
    Dim P3(0 To 2) As Double
    P1(0) = 20: P1(1) = 0: P1(2) = 0
    P2(0) = 20: P2(1) = 20: P2(2) = 0
    P3(0) = 20: P3(1) = 20: P3(2) = 20

    Dim MirObj As Acad3DSolid
    Set MirObj = newSSet.Item(0).Mirror3D(P1, P2, P3)
    MirObj.Color = 45

    ZoomAll
    ThisDrawing.SelectionSets.Item("newSelSet").Delete
End Sub
```

Move

Metodę `Move` zastosowaliśmy już wcześniej w połączeniu z metodą `Copy` (listing 12.3). Natomiast listing 12.9 przedstawia operację przesunięcia zbioru wskazań o określoną odległość X.

Move (metoda ogólna (39)). `Object, Move Point1` jako `Variant` (tylko wprowadzanie), `Point2` jako `Variant` (tylko wprowadzanie).

W rozdziale 14. utworzymy narzędzie do przekształcania wybranych obiektów względem punktu początkowego lub bazowego, na przykładzie którego omówimy kilka odmian kopiowania i przesuwania obiektów o różne odległości oraz pod określonym kątem.

Listing 12.9. *Przesunięcie kilku obiektów*

```
Sub Move()
    Dim newSSet As AcadSelectionSet
    Set newSSet = ThisDrawing.SelectionSets.Add("newSelSet")
    newSSet.SelectOnScreen

    Dim P1(0 To 2) As Double
    Dim P2(0 To 2) As Double
    Dim Dist As Integer
    Dim i As Integer
    Dist = ThisDrawing.Utility.GetInteger("Podaj odległość X: ")
    P1(0) = 0:    P1(1) = 0: P1(2) = 0
    P2(0) = Dist: P2(1) = 0: P2(2) = 0

    For i = 0 To newSSet.Count - 1
        NewSSet.Item(i).Move P1, P2
    Next i
    ThisDrawing.SelectionSets.Item("newSelSet").Delete
End Sub
```

Offset

Metoda `Offset` tworzy równoległe krzywe i linie oraz współśrodkowe okręgi i elipsy. Jej zastosowanie jest ograniczone do wymienionych przed chwilą obiektów. Jeśli zastosujemy ją do obiektów, których ona nie dotyczy, system wygeneruje komunikat o błędzie „`object doesn't support this property or method`” („obiekt nie obsługuje tej właściwości lub metody”).

Offset. `ReturnValue` jako tablica `Variant` zmiennych typu `Object = object.Offset` (`Distance` jako `Double` [tylko wprowadzanie]). Metoda ta tworzy nowy obiekt odsunięty o określoną odległość (dodatnią lub ujemną, ale nie zerową) od istniejącego obiektu. Metodę tę można stosować wyłącznie z obiektami, takimi jak: `Arc`, `Circle`, `Elipse`, `Line`, `LightWeightPolyline`, `Polyline`, `Spline` oraz `XLine`.

W listingu 12.10 tworzymy, a następnie wskazujemy pojedynczą elipsę, która zostaje powielona przez odsunięcie od oryginalnego obiektu o odległość jednej jednostki wymiarowej.

Listing 12.10. *Odsunięcie obiektu*

```
Sub Offset()
    AddEllipse
    Dim newSSet As AcadSelectionSet
    Set newSSet = ThisDrawing.SelectionSets.Add("newSelSet")
    newSSet.SelectOnScreen
    MsgBox "Odsuń zbiór wskazań..."

    Dim NewObj As Variant
    NewObj = newSSet.Item(0).Offset(1#)
    NewObj(0).Color = acRed

    ZoomAll
    ThisDrawing.SelectionSets.Item("newSelSet").Delete
End Sub
```

Rotate

Metody obrotu w 2D i 3D można zastosować do dowolnego obiektu rysunkowego łącznie z odnośnikami atrybutów.

Rotate (metoda ogólna (39)). `object.Rotate BasePoint` jako `Variant` [tylko wprowadzanie], `RotationAngle` jako `Double` (kąt w radianach, tylko wprowadzanie). Metoda ta służy do obrotu obiektu wokół określonego punktu w dwóch wymiarach.

Procedura z listingu 12.11 pozwala na jednoczesne wskazanie kilku obiektów. Definiuje ona także punkt bazowy (`CtrPnt`) oraz kąt obrotu wynoszący w tym przypadku 45 stopni (wyrażony w radianach).

Listing 12.11. *Obrót w płaszczyźnie X-Y*

```
Sub Rotate()
    Dim newSSet As AcadSelectionSet
    Set newSSet = ThisDrawing.SelectionSets.Add("newSelSet")
    newSSet.SelectOnScreen

    Dim CtrPnt(0 To 2) As Double
    Dim Rangle As Variant
    CtrPnt(0) = 2: CtrPnt(1) = 1: CtrPnt(2) = 0
    Rangle = 0.7853981

    Dim i As Integer
    For i = 0 To newSSet.Count - 1
        newSSet.Item(i).Rotate CtrPnt, Rangle
    Next i
    ThisDrawing.SelectionSets.Item("newSelSet").Delete
End Sub
```

Rotate3D (metoda ogólna (39)). `object.Rotate3D Point1` jako `Variant` [tylko wprowadzanie], `Point2` jako `Variant` [tylko wprowadzanie], `RotationAngle` jako `Double` (kąt w radianach, tylko wprowadzanie). Metoda ta służy do obrotu bryły wokół określonej linii.

Procedura z listingu 12.12 przystosowana jest do obsługi tylko pierwszego wskazanego obiektu, ale obydwie metody związane z obrotem można stosować do wielu elementów, jeśli w podprogramie umieścimy odpowiednią pętlę przetwarzającą wszystkie składniki zbioru wskazań. W omawianym listingu umożliwiliśmy wprowadzanie kąta obrotu w stopniach, które są następnie konwertowane na radiany.

Listing 12.12. *Obrót w przestrzeni trójwymiarowej*

```
Sub Rotate3D()
    Boole
    Dim newSSet As AcadSelectionSet
    Set newSSet = ThisDrawing.SelectionSets.Add("newSelSet")
    newSSet.SelectOnScreen

    Dim P1(0 To 2) As Double
    Dim P2(0 To 2) As Double
    Dim RotAng As Double
    P1(0) = 0: P1(1) = 2: P1(2) = 0
    P1(0) = 0: P1(1) = -2: P1(2) = 0
    RotAng = 90
    RotAng = RotAng * 3.141592 / 180

    newSSet.Item(0).Rotate3D P1, P2, RotAng
    ZoomAll
    ThisDrawing.SelectionSets.Item("newSelSet").Delete
End Sub
```

ScaleEntity

Podobnie jak odpowiadające jej polecenie `Scale`, metoda `ScaleEntity` powiększa lub pomniejsza wszelkie elementy AutoCAD-a o podany współczynnik. Skalowanie odbywa się równomiernie we wszystkich trzech osiach.

ScaleEntity. `object.ScaleEntity BasePoint` jako `Variant` (tylko wprowadzanie), `ScaleFactor` jako `Double` (tylko wprowadzanie). Metoda ta przeprowadza skalowanie równomiernie we wszystkich trzech osiach.

W procedurze z listingu 12.13 zastosowaliśmy metodę `Select`, aby automatycznie wskazać wszystkie widoczne obiekty na rysunku testowym. Wszystkie zaznaczone elementy zostają pomniejszone o dwie trzecie przy użyciu punktu początkowego rysunku jako punktu bazowego skalowania.

Listing 12.13. *Skalowanie w przestrzeni trójwymiarowej*

```
Sub ScaleEntity()
    Dim newSSet As AcadSelectionSet
    Set newSSet = ThisDrawing.SelectionSets.Add("newSelSet")
    newSSet.Select acSelectionSetAll

    Dim BasePt(0 To 2) As Double
    Dim Factor As Double
    BasePt(0) = 0: BasePt(1) = 0: BasePt(2) = 0
    Factor = 0.3333
```

```
    Dim i As Integer
    For i = 0 To newSSet.Count - 1
        newSSet.Item(i).ScaleEntity BasePt, Factor
    Next i
    ThisDrawing.SelectionSets.Item("newSelSet").Delete
End Sub
```

TransformBy

`TransformBy` jest ostatnią z trzech metod edycyjnych, które nie posiadają odpowiedników wśród poleceń AutoCAD-a. Umożliwia ona dokonywanie zmian geometrycznych, które mają fundamentalne znaczenie dla każdej grafiki komputerowej. Nie będziemy zagłębiać się w teorię matematyczną transformacji macierzy 3D, ale należy poznać chociażby podstawową notację, aby korzystać z metody `TransformBy`, która posiada bardzo rozbudowane możliwości.

TransformBy (metoda ogólna (39)). `object.TransformBy TransformationMatrix` jako `Variant` (tablica 4×4 zmiennych `Double`). Metoda ta umożliwia jednocześnie przesunięcie, obrót oraz skalowanie obiektu za pomocą tablicy przekształceń o wymiarach 4×4. Jeśli metoda ta zostanie niewłaściwie sformułowana, AutoCAD zwraca komunikat o błędzie, a w niektórych przypadkach może to spowodować „ogólny błąd modelowania", który uniemożliwia dalsze działanie programu.

Jak mogliśmy zaobserwować przy okazji omawiania metody `Move`, przesunięcie realizowane jest po prostu przez określenie starego i nowego położenia obiektu za pomocą punktów w układzie współrzędnych AutoCAD-a. Następnie odejmujemy od siebie współrzędne tych punktów, uzyskując wektor przesunięcia, który ma zastosowanie do dowolnego punktu obiektu przesuwanego równolegle z jednego punktu do drugiego.

Natomiast operacje obrotu i skalowania nie są już tak proste, ponieważ każdy punkt obiektu musi zostać przemieszczony w odmienny sposób w stosunku do punktu odniesienia. W przypadku skalowania każdy punkt obiektu odsuwany jest wzdłuż osi zaczepionej w punkcie odniesienia. Reprezentacja obrotu może mieć podobną formę, zrealizowana za pomocą sinusa i cosinusa wymaganego kąta. Obydwie te funkcje można połączyć, mnożąc każdy z punktów definicji obiektu przez tablicę o wymiarach 2×2, definiującą funkcję transformacji.

Istnieje możliwość złożenia (konsolidacji) operacji przesunięcia, skalowania i obrotu w dwóch wymiarach w tablicy o wymiarach 3×3. Reprezentację tę można rozszerzyć również na trzy wymiary, jeśli zastosujemy tablicę 4×4. Ogólna tablica metody `TransformBy` AutoCAD-a przedstawiona została w listingu 12.14a. Parametr `R` oznacza tam kryteria obrotu, zdefiniowane za pomocą sinusa i cosinusa. Natomiast `S` i `T` odnoszą się odpowiednio do skalowania i przesunięcia. Wyrażone są one jako współczynniki skalowania w pierwszym przypadku oraz jako odległość — w drugim. Wszystkie one są liczbami rzeczywistymi (typ `Double`).

W listingach 12.14b, c i d przedstawiono odpowiednio obrót wokół osi *X*, *Y* i *Z* przy użyciu funkcji sinus i cosinus. Obrót wokół każdej osi należy skonfigurować za pomocą osobnej tablicy. Podobnie w przypadku skalowania, gdzie wykorzystywane są pozycje matryc R00, R11 i R22 (patrz listing 12.14a), którego nie można połączyć z obrotem. AutoCAD

wymaga również, aby współczynniki skalowania miały jednakową wartość we wszystkich trzech kierunkach. Istnieje natomiast możliwość połączenia przesunięcia z obrotem lub skalowaniem, ponieważ wykorzystuje ono inne pozycje tablicy.

Listing 12.14. *Tablice transformacji (4×4)*

```
     | R00/S  R01    R02    T0|
     | R10    R11/S  R12/S  T1|
     | R10    R21    R22/S  T2|
     |  0      0      0     1|        (a)

Z:   |Cos(R) -Sin(R)  0      0|
     |Sin(R)  Cos(R)  0      0|
     |  0      0      1      0|
     |  0      0      0      1|        (b)

X:   |  1      0      0      0|
     |  0     Cos(R) -Sin(R) 0|
     |  0     Sin(R)  Cos(R) 0|
     |  0      0      0      1|        (c)

Y:   | Cos(R)  0     Sin(R)  0|
     |  0      1      0      0|
     |-Sin(R)  0     Cos(R)  0|
     |  0      0      0      1|        (d)
```

Procedura pokazana w listingu 12.15 przedstawia dwie funkcje przekształceń tablicowych. Zaczynamy od utworzenia w nowym rysunku obiektu `Boole`. Następnie deklarujemy dwuwymiarową tablicę (`M`) jako zmienną matrycy 4×4 oraz zbiór wskazań i obiekt ogólny. Obiekt `Boole` (z rozdziału 10.) wskazywany jest automatycznie za pomocą opcji `acSelectionSetAll` należącej do metody `Select`. Ten pojedynczy element przypisywany jest zmiennej obiektu ogólnego (która może przechowywać dowolny rodzaj elementu i jest wiązana dynamicznie), tak że konieczne jest tylko jednokrotne oszacowanie `newSSet.Item(0)`.

Następnie tworzymy zmienne potrzebne do samego przekształcenia, czyli kąt obrotu (`R`) oraz punkt odniesienia (`T`). Kąt należy podać w stopniach, które są potem zamieniane na radiany, wymagane przez metodę `TransformBy`.

W listingu tym wykonujemy dwa przekształcenia, z których każde poprzedzone jest wyświetleniem okna dialogowego pokazującego używane parametry. Najpierw obracamy obiekt o kąt 60 stopni (w kierunku zgodnym z ruchem wskazówek zegara) bez składników translacji. Następnie zmieniamy skalę obiektu na mniejszą o połowę, jednocześnie przemieszczając go o –5 jednostek we wszystkich trzech osiach. Natomiast na koniec wymazujemy zbiór wskazań.

Można także ściągnąć z Internetu drugą aplikację związaną z metodą `TransformBy` dotyczącą tego rozdziału, która obraca obiekt o niewielkie kąty, jednocześnie stopniowo zmniejszając wyświetlany widok za pomocą polecenia `Zoom`. Tworzy to interesujący efekt animacji, który daje bliższe pojęcie o zasadzie działania polecenia `Orbit` AutoCAD-a, a uzyskujemy to przez zawarcie kilku funkcji transformacji w pętli `For...Next`.

Listing 12.15. *Przekształcenie macierzowe*

```
Sub TransformBy()
    Boole
    Dim M(0 To 3, 0 To 3) As Double
    Dim newSSet As AcadSelectionSet
    Dim Object As Object
    Set newSSet = ThisDrawing.SelectionSets.Add("newSelSet")
    newSSet.Select acSelectionSetAll
    Set Object = newSSet.Item(0)

    Dim R As Double
    Dim T(0 To 2) As Double
    T(0) = 0: T(1) = 0
    R = -60
    R = R * 3.141592 / 180
    MsgBox "Obróć " & R & " radianów wokół osi Z w " & T(0) & "," & T(1) & "," & T(2)
    M(0, 0) = Cos(R): M(0, 1) = -Sin(R)
                      M(0, 2) = 0: M(0, 3) = T(0)
    M(1, 0) = Sin(R): M(1, 1) = Cos(R)
                      M(1, 2) = 0: M(1, 3) = T(1)
    M(2, 0) = 0:      M(2, 1) = 0
                      M(2, 2) = 1: M(2, 3) = T(2)
    M(3, 0) = 0:      M(3, 1) = 0
                      M(3, 2) = 0: M(3, 3) = 1
    Object.TransformBy (M)
    Object.Update

    MsgBox "Skaluj .5 w -5,-5,-5..."
    M(0, 0) = 0.5: M(0, 1) = 0
                   M(0, 2) = 0:   M(0, 3) = -5
    M(1, 0) = 0:   M(1, 1) = 0.5
                   M(1, 2) = 0:   M(1, 3) = -5
    M(2, 0) = 0:   M(2, 1) = 0
                   M(2, 2) = 0.5: M(2, 3) = -5
    M(3, 0) = 0:   M(3, 1) = 0
                   M(3, 2) = 0:   M(3, 3) = 1
    Object.TransformBy (M)

    ThisDrawing.SelectionSets.Item("newSelSet").Delete
End Sub
```

Undo

Polecenie `Undo` AutoCAD-a posiada kilka opcji służących do sterowania rezultatem jej działania, które polega na cofnięciu wykonanego poprzednio polecenia. Opcje te wykorzystywane są do pogrupowania czynności w sekwencje, które można cofnąć jako całość. Natomiast metoda `Undo` VBA obsługuje dwie z tych opcji, umożliwiając określenie początku i końca sekwencji wykonanych czynności.

EndUndoMark (metoda obiektu `Document`). `object.EndUndoMark` ustawia środowisko AutoCAD-a w ten sposób, że czynności wykonane między `StartUndoMark` a tą metodą traktowane są jako oddzielna grupa (patrz listing `StartUndoMark`).

StartUndoMark (metoda obiektu Document). `object.StartUndoMark` ustawia środowisko AutoCAD-a w ten sposób, że czynności wykonane między tą metodą a `EndUndoMark` traktowane są jako oddzielna grupa. W ten sposób zdefiniowane zostają oddzielne znaczniki `Undo` dla każdej tego typu czynności tak, że później można je cofnąć raczej indywidualnie, a nie jako grupę.

Procedura w listingu 12.16 tworzy rząd okręgów odsuniętych od siebie o 3 jednostki miary, z których każdy posiada zdefiniowany znacznik cofnięcia. Możemy teraz wprowadzić polecenie `Undo` lub `U` i w ten sposób cofniemy po jednym okręgu za każdym razem. Jeśli natomiast nie wywołamy metody `StartUndoMark`, wówczas polecenie `Undo` AutoCAD-a spowoduje cofnięcie od razu całego rzędu okręgów.

Listing 12.16. *Znaczniki cofania poleceń*

```
Sub UndoMarks()
    Dim Object As AcadCircle
    Dim center(0 To 2) As Double
    Dim Radius As Double
    center(0) = 0: center(1) = 0: center(2) = 0
    Radius = 1

    Dim i As Integer
    For i = 0 To 7
        ThisDrawing.StartUndoMark
        Set Object = ThisDrawing.ModelSpace.AddCircle(center, Radius)
        center(0) = center(0) + 3
        ThisDrawing.EndUndoMark
    Next
    ZoomAll
End Sub
```

Właściwości informacyjne

Właściwości AutoCAD-a związane z edycją posiadają dostęp typu „odczyt i zapis", dzięki czemu są bardzo pomocne w aktywnym wykonywaniu funkcji edycyjnych oraz przy zapytaniach. Na przykład właściwości `Coordinate` i `Coordinates` można wykorzystać do programowego wykonania tych samych czynności, które realizują polecenia `Extend` i `Trim`. Aby fizycznie zmienić cechy charakterystyczne elementów rysunkowych, można zastosować również właściwości `InsertionPoint`, `Rotation` oraz `Thickness`.

Właściwość `Area` umożliwia określenie powierzchni ograniczonej za pomocą kilku elementów AutoCAD-a. Właściwość tę wykorzystujemy obszernie w projekcie przestrzeni biurowej w rozdziale 19.

Area. Podaje obszar ograniczony obiektami. Typ: `Double` (odczyt i zapis). Właściwość ta dotyczy wyłącznie obiektów, takich jak: `Arc`, `Circle`, `Ellipse`, `LightWeightPolyline`, `Polyline`, `Region` oraz `Spline`.

Procedura w listingu 12.17 tworzy uproszczoną polilinię (obiekt `LightWeightPolyline`), a następnie wyświetla za pomocą standardowego okna dialogowego powierzchnię ograniczoną przez ten obiekt. Natomiast jeśli podejmiemy próbę zastosowania tej właściwości

(jak również właściwości omówionych poniżej) w stosunku do elementów, które jej nie obsługują, AutoCAD wygeneruje komunikat o błędzie.

Listing 12.17. *Obliczanie powierzchni*

```
Sub Area()
    Dim Object As AcadLWPolyline
    Dim P(0 To 7) As Double
    P(0) = 0:  P(1) = 0:  P(2) = 0:  P(3) = 10
    P(4) = 10: P(5) = 10: P(6) = 10: P(7) = 0
    Set Object = ThisDrawing.ModelSpace.AddLightWeightPolyline(P)
    Object.Closed = True
    ZoomAll
    MsgBox "Obszar polilinii: " & Object.Area
End Sub
```

Coordinate. Podaje współrzędną pojedynczego wierzchołka danego obiektu. Typ: `Variant` (odczyt i zapis). Właściwość ta dotyczy wyłącznie obiektów, takich jak: `3DPoly`, `Leader`, `LightWeightPolyline`, `Point`, `PolyfaceMesh`, `Polyline`, `Solid` oraz `Trace`.

Właściwość `Coordinate` umożliwia określenie lub zmianę pojedynczego wierzchołka należącego do obiektów wymienionych powyżej. W listingu 12.18 wykorzystaliśmy uproszczoną polilinię z listingu 12.17.

Listing 12.18. *Zmiana jednej współrzędnej*

```
Sub Coordinate()
    Dim newSSet As AcadSelectionSet
    Set newSSet = ThisDrawing.SelectionSets.Add("newSelSet")
    newSSet.Select acSelectionSetAll

    Dim Coords As Variant
    Coords = newSSet.Item(0).Coordinate(2)
    MsgBox "3-ci w: " & Coords(0) & "," & Coords(1)

    Coords(0) = 4: Coords(1) = 4
    NewSSet.Item(0).Coordinate(2) = Coords
    NewSSet.Item(0).Update
    MsgBox "3-ci jest w: " & Coords(0) & "," & Coords(1)

    ThisDrawing.SelectionSets.Item("newSelSet").Delete
End Sub
```

Deklarujemy zmienną `Variant` (`Coords`) i w tym przypadku nadajemy jej wartości trzeciego wierzchołka polilinii (współrzędna X-Y określona za pomocą punktów `P(4)` i `P(5)` z listingu 12.17). Wiemy, że jest to punkt `Coordinate(2)`, pamiętając, że numeracja rozpoczyna się od zera. Natomiast elementy tablicy `Coords` (`0` i `1`) zwracają współrzędne tego punktu, wyświetlane następnie w oknie komunikatów. Następnie nadajemy nowe wartości współrzędnym X i Y, wykorzystując do zmiany położenia wierzchołka właściwość `Coordinate` w trybie zapisu. Czynność tę zamyka metoda `Update`, po której usuwamy zbiór wskazań.

Coordinates. Podaje współrzędne wszystkich wierzchołków danego obiektu. Typ: tablica `Variant` zmiennych `Double` (odczyt i zapis). Właściwość ta dotyczy wyłącznie obiektów,

takich jak: 3DFace, 3DPoly, Leader, LightWeightPolyline, MLine, Point, PolyfaceMesh, PolygonMesh, Polyline, Solid oraz Trace.

Właściwość będąca zwielokrotnieniem właściwości Coordinate jest do niej bardzo podobna (ale potencjalnie bardziej zagmatwana). Umożliwia ona dostęp do wszystkich wierzchołków wybranego obiektu. Po raz kolejny wykonamy tu operację na polilinii z listingu 12.17.

Właściwość Coordinates wykorzystana w listingu 12.19 przypisuje wszystkie wierzchołki do tablicy typu Variant o nazwie Coords. Współrzędne X zawarte są w nieparzystych elementach tej tablicy, natomiast współrzędne Y — w parzystych. Jeśli chcemy więc zmienić drugi i czwarty wierzchołek, stosujemy składniki tablicy przedstawione w omawianej procedurze. Na tym etapie pierwotna polilinia, która w listingu 12.17 miała początkowo kształt kwadratu (porównaj), ma kształt grotu strzały skierowanej w kierunku lewego dolnego narożnika ekranu.

W międzyczasie należy wspomnieć także o odległości, którą, jeśli byłaby ona właściwością VBA, należałoby w tym momencie wymienić, trzymając się porządku alfabetycznego. Odległości można określać za pomocą metody GetDistance, która jest jedną z metod obiektu narzędziowego, omówionych w rozdziale 7. (listing 7.12).

Listing 12.19. *Zmiana kilku współrzędnych*

```
Sub Coordinates()
    Dim newSSet As AcadSelectionSet
    Set newSSet = ThisDrawing.SelectionSets.Add("newSelSet")
    newSSet.Select acSelectionSetAll

    Dim Coords As Variant
    Coords = newSSet.Item(0).Coordinates
    MsgBox "Obiekt posiada " & (UBound(Coords)+ 1) / 2 & "wierzchołków" & vbCr & "2-gi w: _
    " & Coords(2) & ", " & Coords(3) & "4-ty w: " & Coords(6) & ", " & Coords(7)
    Coords(2) = 5:   Coords(3) = 9
    Coords(6) = 9:   Coords(7) = 5

    newSSet.Item(0).Coordinates = Coords
    ThisDrawing.Regen True
    MsgBox "2-gi jest w: " & Coords(2) & ", " & Coords(3) & "4-ty jest w: " & _
        Coords(6) & ", " & Coords(7)
    ThisDrawing.SelectionSets.Item("newSelSet").Delete
End Sub
```

InsertionPoint (właściwość ogólna (9)). Podaje punkt, w którym dany obiekt wstawiany jest do rysunku. Typ: tablica Variant zmiennych Double (odczyt i zapis). Jest to współrzędna 3D dotycząca obiektów, takich jak: Attribute, AttributeRef, BlockRef, ExternalReference, MInsertBlock, MText, Shape, Text oraz Tolerance.

Listing 12.20 zmienia punkt wstawienia obiektu Text. Obiekt ten wskazywany jest za pomocą metody SelectAtPoint; następnie jego punkt wstawienia przypisujemy do zmiennej R typu Variant i wyświetlamy jego współrzędne na ekranie. Zmienna ta jest potem redefiniowana, a jej nowe wartości zwrócone zostają do wskazanego elementu, który na koniec zostaje uaktualniony.

Listing 12.20. *Zmiana punktu wstawienia*

```
Sub InsertionPoint()
    AddText
    ThisDrawing.Regen True
    Dim newSSet As AcadSelectionSet
    Set newSSet = ThisDrawing.SelectionSets.Add("newSelSet")
    Dim P(0 To 2) As Double
    P(0) = 3: P(1) = 3: P(2) = 0
    newSSet.SelectAtPoint P

    Dim R As Variant
    R = newSSet.Item(0).InsertionPoint
    MsgBox "Punkt wstawienia: " & R(0) & ", " & R(1) & ", " & R(2)

    R(0) = 5: R(1) = 7: R(2) = 0
    newSSet.Item(0).InsertionPoint = R
    newSSet.Item(0).Update
    ThisDrawing.Regen True
    MsgBox "Nowy punkt: " & R(0) & ", " & R(1) & ", " & R(2)
    ThisDrawing.SelectionSets.Item("newSelSet").Delete
End Sub
```

Normal (właściwość ogólna (28)). Podaje trójwymiarowy wektor normalny osi *Z* obiektu graficznego (trójelementowa tablica zmiennych `Double`). Typ: `Variant` (odczyt i zapis). Właściwość ta podaje wektor, który określa kierunek normalny i który może być dodany do wybranego punktu, aby otrzymać inny punkt. Należy pamiętać, że nie jest to współrzędna podająca umiejscowienie w przestrzeni.

Właściwość `Normal` określa oś *Z* obiektu we współrzędnych układu WCS. Właściwości tej można użyć jako parametru `OCSNormal`, gdy przeprowadzamy konwersję współrzędnych za pomocą metody `TranslateCoordinates` (patrz listing 7.27).

W listingu 12.21 zmieniamy właściwość `Normal` elementu `Hatch`, utworzonego za pomocą procedury `AddHatch` z rozdziału 9. Potem wskazujemy ten obiekt, a następnie deklarujemy zmienną `N` typu `Variant` oraz nadajemy jej wartość za pomocą właściwości `Normal` obiektu `newSSet.Item(1)`. Jest to zbiór wskazań, w którym znajdują się dwa elementy; pierwszym z nich jest zewnętrzna pętla (`Item(0)`), która określa granice kreskowania. My natomiast będziemy edytować element `Item(1)`, którym jest samo kreskowanie. W kolejnym etapie po wyświetleniu wartości wektora `Normal` za pomocą okna dialogowego nadajemy mu nową wartość, która przypisywana jest do elementu. Na koniec uaktualniamy ten obiekt i wyświetlamy na ekranie nowe współrzędne wektora.

Listing 12.21. *Zmiana wektora normalnego*

```
Sub Normal()
    AddHatch
    ThisDrawing.Regen True
    Dim newSSet As AcadSelectionSet
    Set newSSet = ThisDrawing.SelectionSets.Add("newSelSet")
    newSSet.Select acSelectionSetAll

    Dim N As Variant
    N = newSSet.Item(1).Normal
    MsgBox "Wektor normalny: " & N(0) & ", " & N(1) & ", " & N(2)
```

```
    N(0) = 1: N(1) = 1: N(2) = -1
    newSSet.Item(1).Normal = N
    newSSet.Item(1).Update
    ZoomAll
    MsgBox "Nowy wektor normalny: " & N(0) & ", " & N(1) & ", " & N(2)
    ThisDrawing.SelectionSets.Item("newSelSet").Delete
End Sub
```

Rotation (właściwość ogólna (16)). Podaje kąt obrotu obiektu w stosunku do osi *X* globalnego układu współrzędnych (WCS). Typ: `Double` (odczyt i zapis). Dodatnie kąty mierzone są w kierunku zgodnym z ruchem wskazówek zegara, patrząc w kierunku ujemnych wartości osi *Z*.

Listing 12.22. *Zmiana kąta obrotu*

```
Sub Rotation()
    AddText
    ThisDrawing.Regen True
    Dim newSSet As AcadSelectionSet
    Set newSSet = ThisDrawing.SelectionSets.Add("newSelSet")
    Dim P(0 To 2) As Double
    P(0) = 3: P(1) = 3: P(2) = 0
    newSSet.SelectAtPoint P
    MsgBox "Kąt obrotu: " & newSSet.Item(0).Rotation

    Dim RotAng As Double
    RotAng = 30
    RotAng = RotAng * 3.141592 / 180
    newSSet.Item(0).Rotation = RotAng
    newSSet.Item(0).Update
    MsgBox "Kąt obrotu wynosi teraz: " & newSSet.Item(0).Rotation

    ThisDrawing.SelectionSets.Item("newSelSet").Delete
End Sub
```

Właściwość `Rotation` zwraca lub przypisuje kąt obrotu większości obiektów graficznych AutoCAD-a. Jednostkami są w tym przypadku radiany.

Procedura przedstawiona w listingu 12.22 wykorzystuje metodę `SelectAtPoint` do wskazania linii tekstu. Należy pamiętać, że w tym przypadku przed wybraniem obiektu trzeba koniecznie zregenerować rysunek. W przeciwnym razie bowiem metoda nie będzie działać poprawnie.

Okno komunikatów wyświetla aktualny kąt obrotu, po czym nadajemy mu nową wartość, podaną w stopniach, która jest niezwłocznie zamieniana na radiany. Następnie za pomocą właściwości `Rotation` w trybie zapisu nową wartość kąta przypisujemy do wskazanego elementu i ponownie wyświetlamy ją na ekranie.

Ostatnią właściwością z tej grupy jest `Thickness`, która służy do określenia lub wprowadzenia wymiaru Z elementu dwuwymiarowego.

Thickness. Właściwość ta określa odległość, na jaką wyciągnięty zostaje obiekt 2D ponad lub poniżej poziomu jego wzniesienia. Typ: `Double` (odczyt i zapis). Grubość ta wykorzystywana jest do nadania trzeciego wymiaru obiektowi 2D. Właściwość obsłu-

giwana jest tylko przez obiekty: Arc, Attribute, AttributeRef, Circle, LightWeight-Polyline, Line, Point, Polyline, Shape, Solid, Text oraz Trace.

W listingu 12.23 wskazujemy najpierw wszystkie elementy znajdujące się na rysunku, po czym za pomocą pętli For...Next zwiększamy o 5 jednostek grubość tych obiektów, które obsługują właściwość Thickness. W procedurze tej umieściliśmy całkiem rozbudowany mechanizm wychwytywania błędów, służący do rozpoznania obiektów nieposiadających właściwości Thickness przez zmianę ich koloru na czerwony. Natomiast okna dialogowe wyświetlają status każdego elementu przed jego edycją i po niej.

Listing 12.23. *Zmiana grubości obiektu*

```
Sub Thickness()
    Dim newSSet As AcadSelectionSet
    Set newSSet = ThisDrawing.SelectionSets.Add("newSelSet")
    newSSet.Select acSelectionSetAll

    Dim i As Integer
    Dim Thickn As Variant
    On Error GoTo ErrorHandler
    For i = 0 To newSSet.Count - 1
        Thickn = newSSet.Item(i).Thickness
        MsgBox "Element " & i & " Grubość: " & newSSet.Item(i).Thickness
        newSSet.Item(i).Thickness = Thickn + 5
        newSSet.Item(i).Update
        MsgBox "Element " & i & " Nowa Grubość: " & newSSet.Item(i).Thickness
    Next i
    ThisDrawing.SendCommand „_VPOINT R 315 30 "
    ThisDrawing.SelectionSets.Item("newSelSet").Delete
Exit Sub

ErrorHandler:
    newSSet.Item(i).Color = acRed
    newSSet.Item(i).Update
    MsgBox "Element " & i & " Nie posiada grubości."
    i = i + 1
    Resume Next
End Sub
```

Dostęp do wiersza poleceń

Metoda SendCommand pozwala w języku VBA dla AutoCAD-a na tworzenie skryptów, umożliwiając emulację wiersza poleceń aplikacji z poziomu VBA. Polecenia wykonywane tą drogą działają zwykle wolniej niż odpowiadające im funkcje VBA, chociaż metoda ta może być w pewnych okolicznościach przydatna. Z tego właśnie powodu firma Autodesk zaleca, aby w przypadku istnienia odpowiednika wśród innych metod VBA nie stosować SendCommand.

SendCommand (metoda obiektu Document). object.SendCommand (Command jako String [tylko wprowadzanie]).

Metoda SendCommand emuluje wykonywanie serii czynności z poziomu wiersza poleceń AutoCAD-a. Jej łańcuch musi zawierać wszystkie argumenty plecenia dokładnie w takiej kolejności, w jakiej są one wymagane łącznie z odstępami lub znakiem ASCII 13, tam gdzie wymagany jest tzw. powrót karetki. (Wyrażenie vbCR jest stałą VBA odpowiadającą znakowi powrotu karetki).

Listing 12.24. *Wysyłanie poleceń*

```
Sub SendCommand()
    ThisDrawing.SendCommand "TEXT 1,1,0 1 45 To jest krótki paragraf," + vbCr
    ThisDrawing.SendCommand "   obrócony i wyświetlony" + vbCr
    ThisDrawing.SendCommand "   pod kątem 45 stopni..." + vbCr + vbCr
    ThisDrawing.SendCommand "ZOOM W -1,-1 20,20 "
    ThisDrawing.SendCommand „_VPOINT R 315 45 "
End Sub
```

Składnia metody SendCommand jest bardzo ściśle określona i wymaga, aby wszystkie argumenty wywoływanego polecenia podawane były dokładnie w takiej kolejności, w jakiej spodziewane jest to w AutoCAD-zie. W listingu 12.24 pokazano zastosowanie tej metody do utworzenia kilku wierszy tekstu, w którym konieczne jest użycie znaku Chr(13) w celu rozróżnienia między końcem wiersza a odstępami w tekście. Natomiast za pomocą polecenia Undo można przywrócić początkowy kąt wyświetlania tekstu.

Inne metody, właściwości i procedury związane z edycją

W modelu obiektu AutoCAD-a istnieje jeszcze wiele innych komponentów związanych z edycją obiektów oraz obsługą procesu edycji. Poniżej przedstawiliśmy zestawienie najważniejszych z nich, wraz z podaniem rozdziału, w którym zostały omówione.

Linie konstrukcyjne. Xline oraz Ray omówiono w rozdziale 9.

Pobieranie danych. W rozdziale 7. przedstawiono kilka metod związanych z pobieraniem danych od użytkownika, takich jak GetPoint, GetKeyword oraz GetString.

Konwersja danych. Metody służące do przekształcania danych, takie jak AngleToString oraz RealToString, omówione zostały w rozdziale 7.

Siatka węzłów, skok oraz tryb ortogonalny. Operacje, takie jak włączanie i wyłączanie siatki węzłów i skoku, ustawianie odstępów siatki oraz właściwość OthoOn, są częścią dyskusji na temat rzutni w rozdziale 5.

Tryb uchwytów rysunkowych. Rozdział 6. obejmuje m.in. omówienie włączania i wyłączania trybu uchwytów rysunkowych (Osnap).

Dokładność. Metody związane z ustawieniami dokładności liczbowej, które związane są przede wszystkim z wymiarami, takie jak PrimaryUnitsPrecision, TextPrecision, TolerancePrecision, AltTolerancePrecision oraz AltUnitsPrecision, przedstawiono w rozdziale 11.

Lokalne układy współrzędnych. Metoda `AddUCS` oraz właściwość `ActiveUCS` omówione zostały w rozdziale 6.

W rozdziale niniejszymużywaliśmy często sześciu procedur służących do tworzenia elementów rysunkowych, które poddawane były następnie edycji. Metody `Add3Dpoly` i `Add3Hatch` pochodzą z rozdziału 9. Z kolei `Boole`, która tworzy interesujący układ walca i kuli, znajduje się w rozdziale 10. Natomiast procedury `AddEllipse`, `AddLine`, oraz `AddText` (patrz rozdział 9.) zostały uproszczone dla celów tego rozdziału, a ich zmodyfikowany kod pokazano poniżej. Warto wspomnieć, że kody wszystkich listingów umieszczonych w tej książce można ściągnąć ze strony internetowej wydawnictwa Prentice Hall pod adresem *www.phptr.com*.

```
Sub AddEllipse()
    Dim Object As AcadEllipse
    Dim center(0 To 2) As Double
    Dim AxisMj(0 To 2) As Double
    Dim RadRat As Double

    center(0) = 0: center(1) = 0: center(2) = 0
    AxisMj(0) = 1: AxisMj(1) = 2: AxisMj(2) = 0
    RadRat = 0.5
    Set Object = ThisDrawing.ModelSpace.AddEllipse (center, AxisMj, RadRat)
End Sub

Sub AddLine()
    Dim Object As AcadLine
    Dim P1(0 To 2) As Double
    Dim P2(0 To 2) As Double
    Dim RadRat As Double

    P1(0) = -2.5: P1(1) = -1.5: P1(2) = 0
    P2(0) =  2.5: P2(1) =  1.5: P2(2) = 0
    Set Object = ThisDrawing.ModelSpace.AddLine (P1, P2)
End Sub

Sub AddText()
    Dim Object As AcadText
    Dim Text As String
    Dim Height As Double
    Dim P(0 To 2) As Double
    Text = "Halo, Centrala?"
    Height = 1
    P(0) = 3: P(1) = 3: P(2) = 0
    Set Object = ThisDrawing.ModelSpace.AddText (Text, P, Height)
End Sub
```

Podsumowanie

Rozdział ten przybliżył wiele metod AutoCAD-a służących do zmiany jego obiektów graficznych. Przyjrzeliśmy się nie tylko podstawowym metodom edycji, takim jak `Copy`, `Move`, `Mirror`, czy też `ScaleEntity`, ale także bardziej zaawansowanym, czyli tablicom biegunowym i prostokątnym oraz przekształceniom macierzowym. Następnie skupiliśmy

naszą uwagę na właściwościach związanych z informacjami o elementach, takimi jak `Area` lub `Thickness`.

W rozdziale 13. omówimy pewne rodzaje zjawisk występujących w normalnym toku działania aplikacji, dla których możemy utworzyć specjalne procedury, zwane podprogramami obsługi zdarzeń, które uruchamiane będą na skutek wystąpienia tychże zdarzeń. Dyskusja ta wprowadzi nas do rozdziału 14., dotyczącego formularzy i kontrolek, które stanowią rdzeń zarówno interfejsu użytkownika w VBA, jak i interaktywnego środowiska programowania IDE.