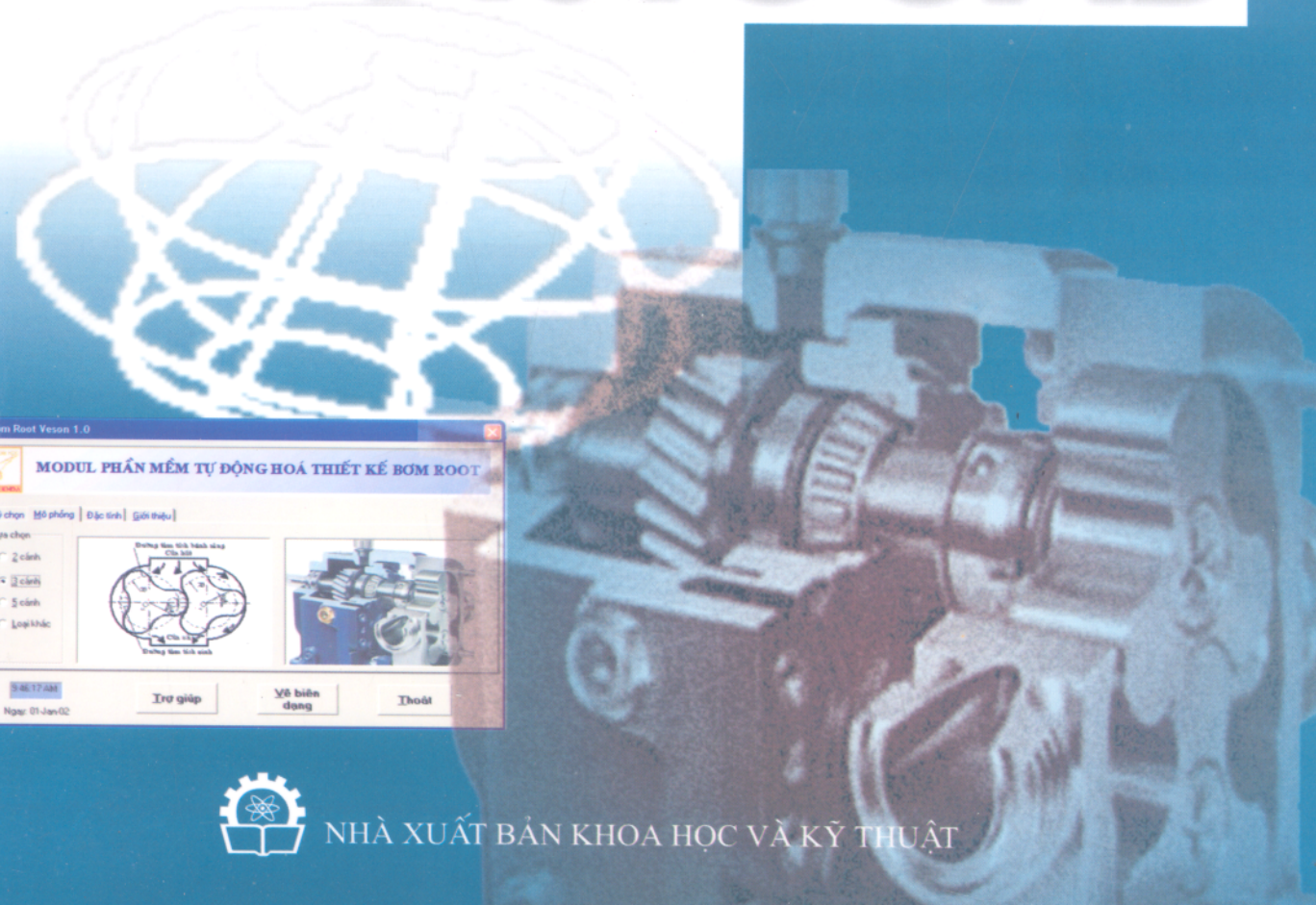
TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI
NGUYỄN HỒNG THÁI (Chủ biên)
VƯƠNG VĂN THANH
ĐẶNG BẢO LÂM

TỦ SÁCH TIN HỌC KỸ THUẬT

# Cơ sở lập trình tự động hóa tính toán, thiết kế với VB & VBA trong môi trường AUTOCAD




NHÀ XUẤT BẢN KHOA HỌC VÀ KỸ THUẬT

NGUYỄN HỒNG THÁI (Chủ biên)
VƯƠNG VĂN THANH, ĐẶNG BẢO LÂM

# CƠ SỞ LẬP TRÌNH TỰ ĐỘNG HOÁ TÍNH TOÁN, THIẾT KẾ VỚI VB & VBA TRONG MÔI TRƯỜNG AUTOCAD

NHÀ XUẤT BẢN KHOA HỌC VÀ KỸ THUẬT
HÀ NỘI

# CƠ SỞ LẬP TRÌNH TỰ ĐỘNG HOÁ TÍNH TOÁN, THIẾT KẾ VỚI VB & VBA TRONG MÔI TRƯỜNG AUTOCAD

**Tác giả:**

**NGUYỄN HỒNG THÁI (chủ biên)**

**VƯƠNG VĂN THANH, ĐẶNG BẢO LÂM**

*Chịu trách nhiệm xuất bản*: PGS. TS. TÔ ĐĂNG HẢI

*Biên tập và sửa bài*: ThS. NGUYỄN HUY TIẾN

HOÀNG GIANG

*Trình bày bìa*: HƯƠNG LAN

**NHÀ XUẤT BẢN KHOA HỌC VÀ KỸ THUẬT**

**70 Trần Hưng Đạo, Hà Nội**

---

In 500 cuốn, khổ 19 × 27 cm, tại Xí nghiệp In Thương Mại (Bộ Công Thương)

Quyết định xuất bản số: 476–2007/CXB/26–18/KHKT-20/10/2007

In xong và nộp lưu chiểu Quý I năm 2008.

# LỜI NÓI ĐẦU

Phần mềm CAD – (*Computer Aided Design*) là phần mềm thiết kế được sử dụng cho hầu hết các ngành (cơ khí chế tạo máy, kiến trúc, xây dựng, cầu đường, điện v.v...) mang lại hiệu quả cao. Tuy nhiên để nâng cao khả năng thiết kế với việc tạo ra các macro lệnh mới, modul chương trình tính toán, vẽ bản vẽ tự động sử dụng thường xuyên tích hợp trong môi trường CAD hay các chương trình ứng dụng riêng biệt bằng các ngôn ngữ Lisp, Visual Lisp, Visual C, Visual Basic và VBA là rất cần thiết.

VBA là ngôn ngữ bậc cao cũng như ngôn ngữ Lisp được tích hợp trong môi trường CAD. Nhưng dễ sử dụng và quản lý cơ sở dữ liệu, thiết kế giao diện tốt hơn Lisp, phù hợp với việc tạo các phần mềm ứng dụng nhằm tự động hoá thiết kế.

Trong tài liệu này chúng tôi trình bày các kiến thức cơ bản của VBA trong môi trường CAD, với nhiều ví dụ để độc giả thực hành lập trình theo sách.

Sách được sử dụng để làm tài liệu giảng dạy lập trình tự động hoá thiết kế cho sinh viên ngành cơ điện tử B của Trường Đại học Bách khoa Hà Nội và các khoá đào tạo phát triển phần mềm ứng dụng tại Trường Đại học Bách khoa Hà Nội. Ngoài ra sách còn là tài liệu đào tạo các khoá lập trình tại trung tâm Nghiên cứu kỹ thuật cơ khí chính xác Trường Đại học Bách khoa Hà Nội.

Các tác giả tỏ lòng biết ơn Nhà xuất bản Khoa học và Kỹ thuật Hà Nội đã động viên và tạo điều kiện thuận lợi để tài liệu này sớm được hoàn thành và ra mắt bạn đọc. Chúng tôi trân trọng cảm ơn các đồng nghiệp trong Bộ môn Cơ sở thiết kế máy & Robot, Khoa Cơ khí, Trường Đại học Bách khoa Hà Nội đã tạo điều kiện cho các tác giả hoàn thành cuốn sách.

Sách được xuất bản lần đầu nên chắc chắn còn nhiều khiếm khuyết cần được bổ sung. Các tác giả trân trọng cảm ơn và mong nhận được nhiều ý kiến đóng góp của độc giả bổ sung, chỉnh sửa cho các lần tái bản sau để cuốn sách ngày càng hoàn thiện.

Mọi ý kiến đóng góp xin gửi về địa chỉ:

Bộ môn Cơ sở Thiết kế máy & Robot – Khoa Cơ khí

Trường Đại học Bách khoa Hà Nội

Email: *hongthai-dtm@mail.hut.edu.vn*

Điện thoại: 04.8684932

**Các tác giả**

# DANH MỤC CÁC TỪ VIẾT TẮT

| STT | Ký hiệu và các từ viết tắt | Tiếng Anh và ý nghĩa |
|---|---|---|
| 1 | VB | Visual Basic |
| 2 | VBA | Visual Basic for Application |
| 3 | ACAD | Auto Cad |
| 4 | MS | Model Space |
| 5 | PS | Paper Space |
| 6 | R | Ma trận |
| 7 | P | Vector |
| 8 | VL | Visual Lip |
| 9 | ACAD Activex | Auto Cad ActiveX Automation Interface |
| 10 | B | Bước |
| 11 | ↵ | Enter |
| 12 | => | Chọn |
| 13 | IDE | Intergrated Development Environment |
| 14 | DAO | Data Access Object |
| 15 | VC | Visual C |

# CÁC TỪ TIẾNG ANH DÙNG TRONG TÀI LIỆU

| STT | Tiếng Anh | Ý nghĩa |
|---|---|---|
| 1 | Radius | Bán kính |
| 2 | Center StartPoint | Điểm tâm |
| 3 | Start Angle | Góc ban đầu |
| 4 | End Point | Điểm cuối |
| 5 | End Angle | Góc cuối |
| 6 | Space | Không gian vẽ hoặc không gian giấy |
| 7 | String | Chuỗi ký tự |
| 8 | Number | Số |

# MỤC LỤC

## Chương 1

# VB & VBA NGÔN NGỮ LẬP TRÌNH TỰ ĐỘNG HOÁ THIẾT KẾ TRONG MÔI TRƯỜNG ACAD

**CAD** là phần mềm thiết kế được sử dụng phổ biến cho hầu hết các ngành như: Cơ khí, xây dựng điện, cầu đường v.v... Tuy nhiên trong quá trình thiết kế tùy thuộc vào từng ngành, công việc thiết kế cụ thể mà người dùng ứng dụng ở mức độ khác nhau. Để nâng cao hiệu quả trong quá trình tính toán thiết kế việc lập trình tạo ra các phần mềm, chương trình vẽ tự động và tra cứu các chi tiết tiêu chuẩn hay các phần mềm tính toán thiết kế nhúng trong môi trường **CAD** là rất cần thiết giúp cho quá trình tính toán, thiết kế được tự động hóa hoàn toàn, nhằm rút ngắn thời gian thiết kế và kết quả chính xác hơn, ví dụ phần mềm thiết kế bánh răng, bơm Roots, thang máy v.v... Như vậy, việc lập trình tự động hóa tính toán, thiết kế là rất cần thiết.

Lập trình nhúng trong môi trường **CAD** có rất nhiều ngôn ngữ như **VL**, **VC**, **VB**, **VBA**... Như vậy có thể lựa chọn một ngôn ngữ lập trình trong những ngôn ngữ trên để xây dựng các modul phần mềm tự động tính toán, thiết kế. Trong tài liệu này tác giả trình bày cách lập trình bằng **VB & VBA** trong môi trường **CAD**. Vậy sức mạnh của **VB & VBA** nhúng trong **CAD** ở đâu?

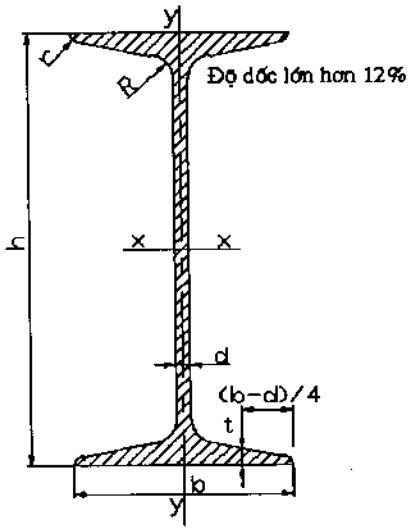
- *Sự thân thiện trong sử dụng.*
- *Hệ thống cơ sở dữ liệu được thực thi dễ dàng.*
- *Quản lý tốt các đối tượng của ACAD.*
- *v.v...*

Để chứng minh điều đó ta xét ví dụ sau.

**Ví dụ:**

Lập trình vẽ tự động thiết diện cắt ngang thép chữ I như sau:

## Thép chữ I ГОТС 8239-56



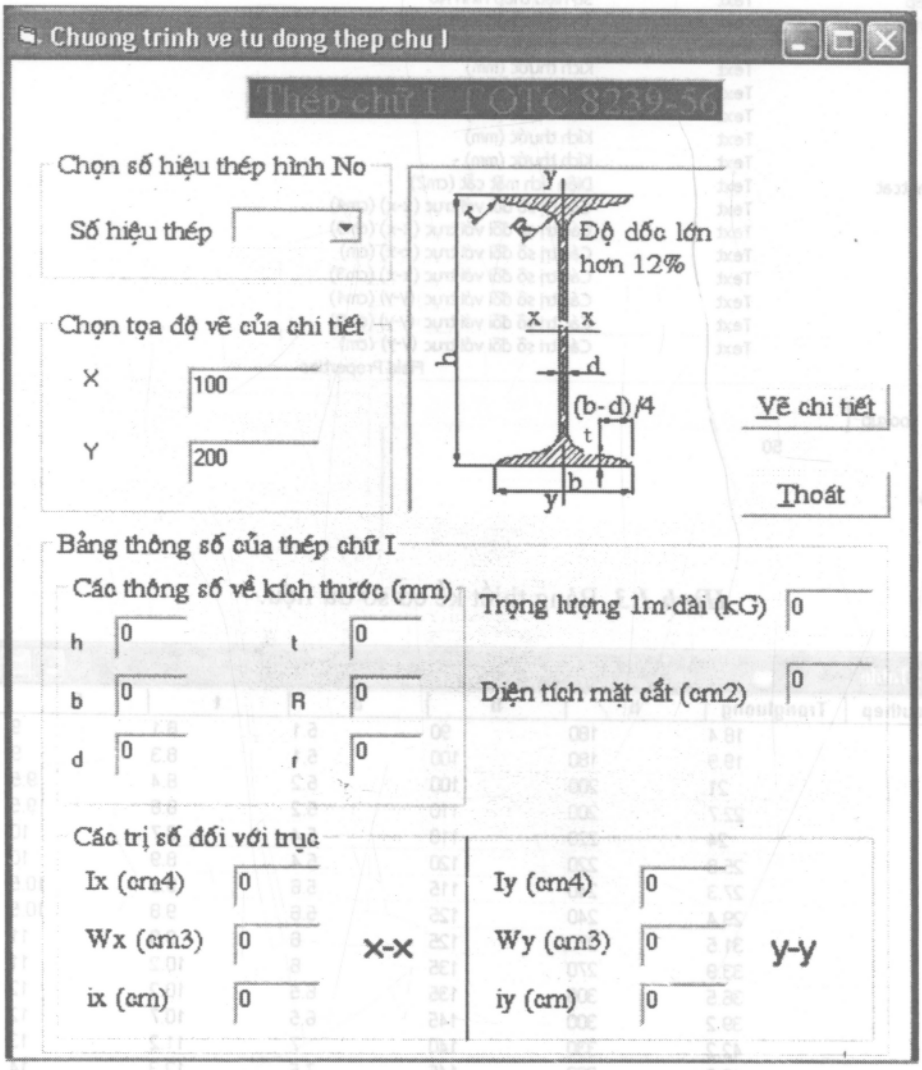

***Hình 1.1.*** **Diện tích mặt cắt ngang của thép chữ I.**

Với các số liệu cho trong bảng 1.1.

***Bảng 1.1.*** Các thông số hình học của thép chữ I

| Số hiệu thép hình N° | Trọng lượng 1 m dài, kG | Kích thước (mm) | | | | | | Diện tích mặt cắt, $cm^2$ | Các trị số đối với trục | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | x - x | | | | y - y | | |
| | | h | b | d | t | R | r | | Ix, $cm^4$ | Wx, $cm^3$ | Ix, cm | Sx, $cm^3$ | Iy, $cm^4$ | Wy, $cm^3$ | Iy, cm |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
| 10 | 9.46 | 100 | 55 | 4.5 | 7.2 | 7 | 2.5 | 12.0 | 198 | 39.7 | 4.06 | 23.0 | 17.9 | 6.49 | 1.22 |
| 12 | 11.5 | 120 | 64 | 4.8 | 7.3 | 7.5 | 3.0 | 14.7 | 350 | 58.4 | 4.38 | 33.7 | 27.9 | 8.72 | 1.38 |
| 14 | 13.7 | 140 | 73 | 4.9 | 7.5 | 8.0 | 3.0 | 17.4 | 572 | 81.7 | 5.73 | 46.8 | 41.9 | 11.5 | 1.55 |
| 16 | 15.9 | 160 | 81 | 5.0 | 7.8 | 8.5 | 3.5 | 20.2 | 873 | 109 | 6.57 | 62.3 | 58.6 | 14.5 | 1.70 |
| 18 | 18.4 | 180 | 90 | 5.1 | 8.1 | 9.0 | 3.5 | 23.4 | 1290 | 143 | 7.42 | 81.4 | 82.6 | 18.4 | 1.88 |
| 18a | 19.9 | 180 | 100 | 5.1 | 8.3 | 9.0 | 3.5 | 25.4 | 1430 | 159 | 7.61 | 89.8 | 114 | 22.8 | 2.12 |
| 20 | 21.0 | 200 | 100 | 5.2 | 8.4 | 9.5 | 4.0 | 26.8 | 1840 | 184 | 8.28 | 104 | 115 | 23.1 | 2.07 |
| 20a | 22.7 | 200 | 110 | 5.2 | 8.6 | 9.5 | 4.0 | 28.9 | 2030 | 203 | 8.37 | 114 | 155 | 28.2 | 2.32 |
| 22 | 24.0 | 220 | 110 | 5.4 | 8.7 | 10.0 | 4.0 | 30.6 | 2550 | 232 | 9.13 | 131 | 157 | 28.6 | 2.27 |
| 22a | 25.3 | 220 | 120 | 5.4 | 8.9 | 10.0 | 4.0 | 32.8 | 2790 | 254 | 9.22 | 143 | 206 | 34.3 | 2.50 |
| 24 | 27.3 | 240 | 115 | 5.6 | 9.5 | 10.5 | 4.0 | 34.8 | 3460 | 289 | 9.97 | 163 | 198 | 34.5 | 2.37 |
| 24a | 29.4 | 240 | 125 | 5.6 | 9.8 | 10.5 | 4.0 | 37.5 | 3800 | 317 | 10.1 | 179 | 260 | 41.6 | 2.63 |

Hình 1.2 dưới đây là giao diện phần mềm vẽ và tra cứu loại thép chữ I theo tiêu chuẩn ГОТС 8239 – 86. Như vậy ta thấy việc phải tra cứu các số liệu trong bảng dữ liệu rồi thực hiện vẽ tiết diện thép chữ I là mất thời gian và công sức. Nhưng với chương trình này việc tra cứu và vẽ tiết diện thép chữ I trở lên nhanh chóng, dễ dàng.




***Hình 1.2.*** Giao diện chương trình.

Dưới đây là các bước lập trình tạo chương trình trên.

***B1: Thiết kế cơ sở dữ liệu***

Cơ sở dữ liệu có thể được xây dựng bằng nhiều phương thức khác nhau ở đây các tác giả xây dựng cơ sở dữ liệu bằng ***Access* 2000** như hình 1.3 và hình 1.4.

Microsoft Access - [Bang1 : Table]

File Edit View Insert Tools Window Help

| Field Name | Data Type | Description |
|---|---|---|
| Sohieuthep | Text | Số hiệu thép hình No |
| Trongluong | Text | Trọng lượng 1 m dài (kG) |
| h | Text | Kích thước (mm) |
| b | Text | Kích thước (mm) |
| d | Text | Kích thước (mm) |
| t | Text | Kích thước (mm) |
| R | Text | Kích thước (mm) |
| r1 | Text | Kích thước (mm) |
| Dientichmatcat | Text | Diện tích mặt cắt (cm2) |
| Ix | Text | Các trị số đối với trục (x-x) (cm4) |
| Wx | Text | Các trị số đối với trục (x-x) (cm3) |
| ix1 | Text | Các trị số đối với trục (x-x) (cm) |
| Sx | Text | Các trị số đối với trục (x-x) (cm3) |
| Iy | Text | Các trị số đối với trục (y-y) (cm4) |
| Wy | Text | Các trị số đối với trục (y-y) (cm3) |
| iy1 | Text | Các trị số đối với trục (y-y) (cm) |

Field Properties

General | Lookup

Field Size 50

Format

Input Mask

Caption

*Hình 1.3.* Bảng thiết kế cơ sở dữ liệu.

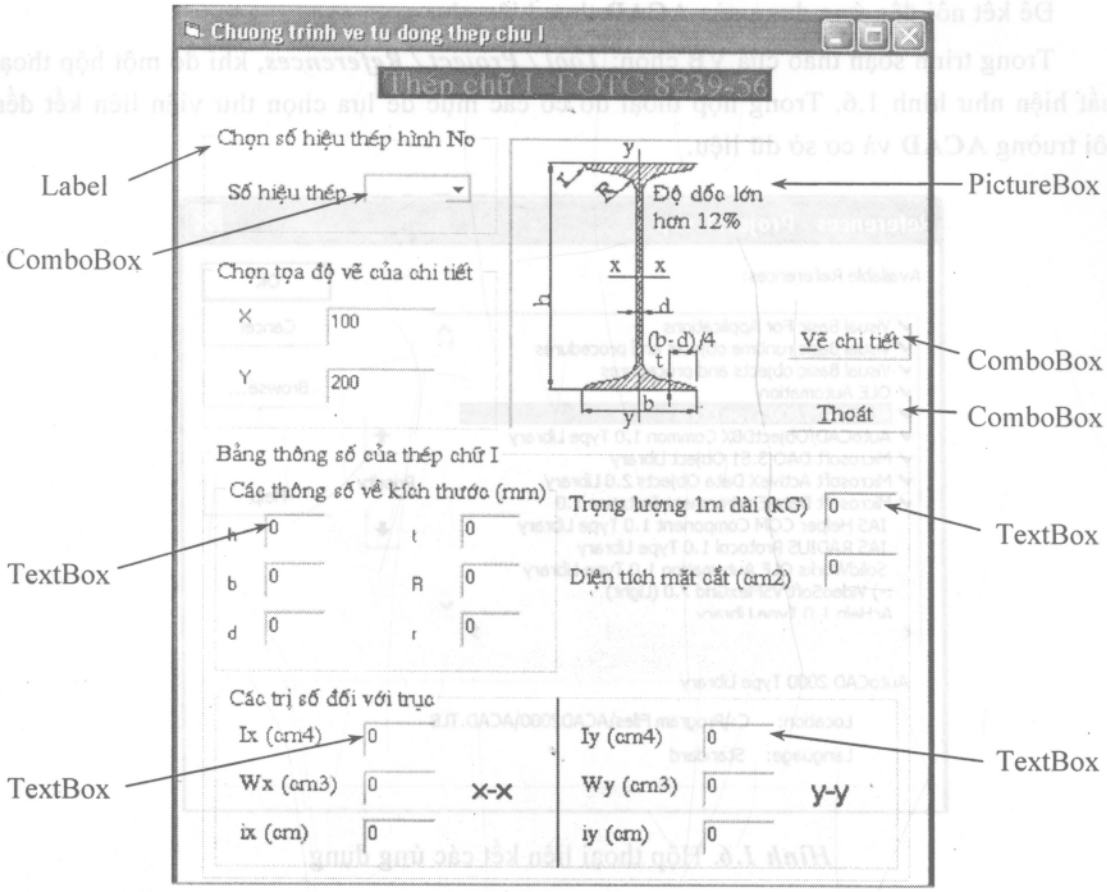
Bang1 : Table

| Sohieuthep | Trongluong | h | b | d | t | R |
|---|---|---|---|---|---|---|
| 18 | 18.4 | 180 | 90 | 5.1 | 8.1 | 9 |
| 18a | 19.9 | 180 | 100 | 5.1 | 8.3 | 9 |
| 20 | 21 | 200 | 100 | 5.2 | 8.4 | 9.5 |
| 20a | 22.7 | 200 | 110 | 5.2 | 8.6 | 9.5 |
| 22 | 24 | 220 | 110 | 5.4 | 8.7 | 10 |
| 22a | 25.8 | 220 | 120 | 5.4 | 8.9 | 10 |
| 24 | 27.3 | 240 | 115 | 5.6 | 9.5 | 10.5 |
| 24a | 29.4 | 240 | 125 | 5.6 | 9.8 | 10.5 |
| 27 | 31.5 | 270 | 125 | 6 | 9.8 | 11 |
| 27a | 33.9 | 270 | 135 | 6 | 10.2 | 11 |
| 30 | 36.5 | 300 | 135 | 6.5 | 10.2 | 12 |
| 30a | 39.2 | 300 | 145 | 6.5 | 10.7 | 12 |
| 33 | 42.2 | 330 | 140 | 7 | 11.2 | 13 |
| 36 | 48.6 | 360 | 145 | 7.5 | 12.3 | 14 |
| 40 | 56.1 | 400 | 155 | 8 | 13 | 15 |
| 45 | 65.2 | 450 | 160 | 8.6 | 14.2 | 16 |
| 50 | 76.8 | 500 | 170 | 9.5 | 15.2 | 17 |
| 55 | 89.8 | 550 | 180 | 10.3 | 16.5 | 18 |
| 60 | 104 | 600 | 190 | 11.1 | 17.8 | 20 |
| 65 | 120 | 650 | 200 | 12 | 19.2 | 22 |
| 70 | 138 | 700 | 210 | 13 | 20.8 | 24 |
| 70a | 158 | 700 | 210 | 15 | 24 | 24 |

Record: 1 of 27

*Hình 1.4.* Bảng cơ sở dữ liệu.

*B2: Thiết kế giao diện chương trình*




***Hình 1.5.*** Thiết kế cửa sổ giao diện chính của chương trình.

Dưới đây là bảng liệt kê chi tiết:

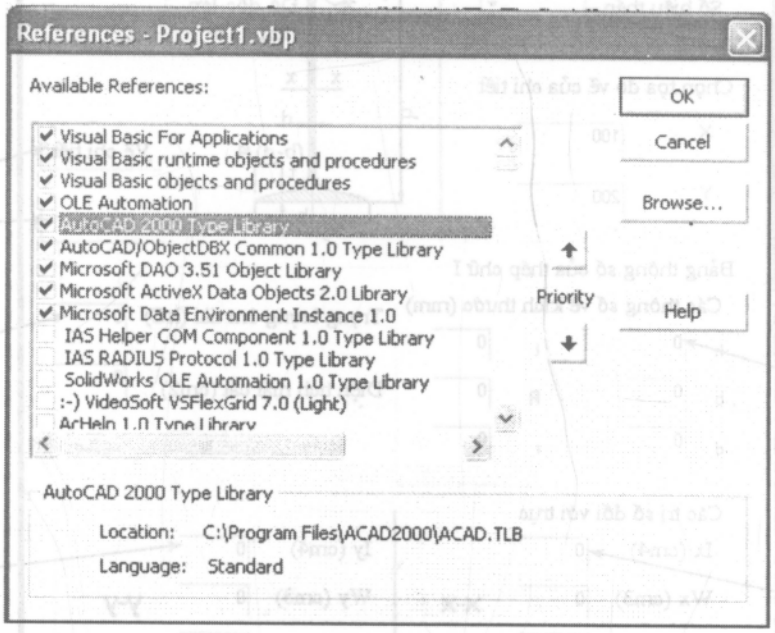
***Bảng 1.2.*** Liệt kê chi tiết các phần chính của hộp thoại

| STT | Các phần chính | Hộp thoại | Ghi chú |
|---|---|---|---|
| 1 | Chọn số hiệu thép | ComboBox | Số hiệu thép |
| 2 | Chọn toạ độ vẽ chi tiết | TextBox | Toạ độ X, Y |
| 3 | Các thông số về kích thước | TextBox | Các thông số: h, b, d, t, R, r |
| 4 | Các trị số đối với trục | TextBox | Các thông số: $I_X$, $W_X$, ix, $S_X$, $I_Y$, $W_Y$, iy. |
| 5 | Vẽ chi tiết | CommandButton | Vẽ chi tiết |
| 6 | Thoát | CommandButton | Thoát |
| 7 | Hình dáng thép chữ I | PictureBox | |
| 8 | Các chữ trên giao diện | Label | Giao tiếp với người sử dụng. |

*B 3: Kết nối ứng dụng với ACAD*

Để kết nối đến ứng dụng của **ACAD** thực hiện như sau:

Trong trình soạn thảo của **VB** chọn: ***Tool / Project / References***, khi đó một hộp thoại xuất hiện như hình 1.6. Trong hộp thoại đó có các mục để lựa chọn thư viện liên kết đến môi trường **ACAD** và cơ sở dữ liệu.




***Hình 1.6.*** Hộp thoại liên kết các ứng dụng.

***Chú ý:*** *Trong hộp thoại trên khi chọn thư viện liên kết với **ACAD** thì cần chú ý đến phiên bản **ACAD** được cài đặt trên máy tính.*

Để liên kết được với môi trường **ACAD** thực hiện như sau:

Từ thanh công cụ **ToolBar** chọn: ***Project / Add_Modul /...*** Thêm vào chương trình đoạn mã như sau:

```
; Thủ tục Public Sub Init_AutoCAD()
Public acadapp As AcadApplication
Public acaddoc As AcadDocument
Public Sub Init_AutoCAD()
    On Error Resume Next
    Set acadapp = GetObject(, "AutoCAD.Application")
    If Err Then
        Err.Clear
```

```
        Set acadapp = CreateObject("AutoCAD.Application")
        If Err Then
            MsgBox Err.Description
            Exit Sub
        End If
    End If
    Set acaddoc = acadapp.ActiveDocument
End Sub
```

***Chú ý:*** - Nếu bạn sử dụng phiên bản của ACAD 2000 thì dùng câu lệnh:

*Set acadapp = GetObject(, "AutoCAD.Application")*

*Set acadapp = CreateObject("AutoCAD.Application")*

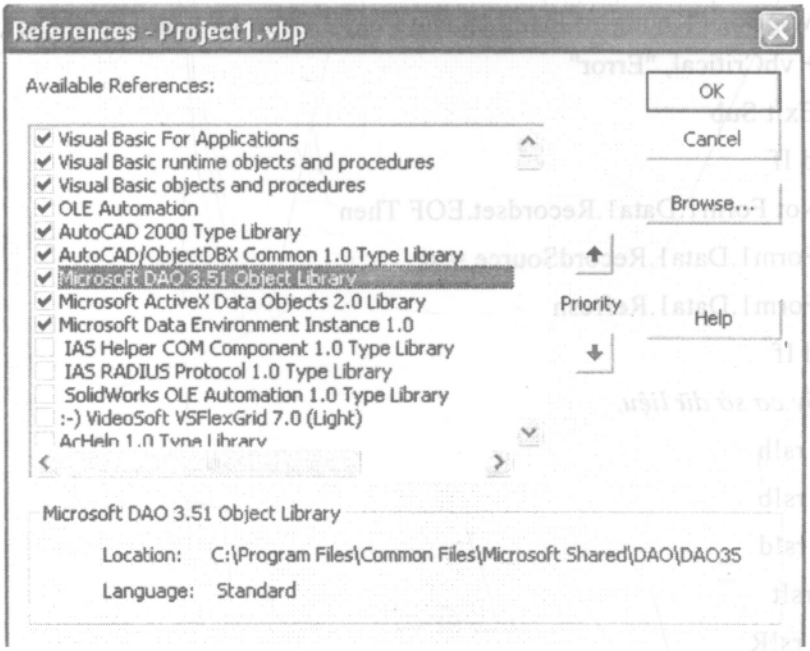
- Nếu bạn sử dụng phiên bản của ACAD 2004 thì dùng câu lệnh:

*Set acadapp = GetObject(, "AutoCAD.Application.16")*

*Set acadapp = CreateObject("AutoCAD.Application")*

***B 4: Kết nối cơ sở dữ liệu***

Có nhiều phương pháp truy nhập cơ sở liệu khác nhau, ở đây sử dụng phương pháp tra cứu cơ sở dữ liệu kiểu **DAO**. Cũng như kết nối với **ACAD** để liên kết với cơ sở dữ liệu cần thực hiện theo trình tự sau: Trong trình soạn thảo của **VB& VBA** chọn: ***Tool/ References*** và chọn Microsoft DAO 3.51 Object Library như hình 1.7.




***Hình 1.7.*** Chọn liên kết cơ sở dữ liệu.

Sau khi đã chọn xong liên kết với cơ sở dữ liệu cần thêm đoạn mã chương trình như sau:

; Tên thủ tục *Public Sub Linh_Database().*

```
Public Sub Linh_Database()
  Dim sohieuthep As String
  Dim h As Double, b As Double, d As Double
  Dim t As Double, R As Double, r1 As Double
  Dim trongluong As Double, dientich As Double
  Dim Ix As Double, Wx As Double, ix1 As Double
  Dim Sx As Double, Iy As Double, Wy As Double
  Dim iy1 As Double, st As String
  sohieuthep = Form1.Cbo_sohieuthep.text
  ' Mở bảng cơ sở dữ liệu.
   Set db = OpenDatabase(Form1.Data1.DatabaseName)
  ' Nhận giá trị các thông số của thép chữ  I  từ bảng cơ sở dữ liệu.
   st  =  "select  h,b,d,t,R,r1,Trongluong,Dientichmatcat,Ix,Wx,ix1,Sx,Iy,Wy,iy1
  from  Bang1 where sohieuthep ='" & sohieuthep & "' "
   Set rs = db.OpenRecordset(st)
   If rs.EOF Then
     MsgBox "Khong co loai thep chu I nay ! Chon lai loai thep chu I.", vbOKOnly
     + vbCritical, "Error"
     Exit Sub
   End If
   If Not Form1.Data1.Recordset.EOF Then
     Form1.Data1.RecordSource = st
     Form1.Data1.Refresh
   End If
   ' Lấy cơ sở dữ liệu.
   h = rs!h
   b = rs!b
   d = rs!d
   t = rs!t
   R = rs!R
   r1 = rs!r1
```

```
        trongluong = rs!trongluong
        dientich = rs!Dientichmatcat
        Ix = rs!Ix
        Wx = rs!Wx
        ix1 = rs!ix1
        Sx = rs!Sx
        Iy = rs!Iy
        Wy = rs!Wy
        iy1 = rs!iy1
        ' Hiện thị các giá trị lên giao diện.
          Form1.Text_h.text = h
          Form1.Text_b.text = b
          Form1.Text_d.text = d
          Form1.Text_t.text = t
          Form1.Text_R.text = R
          Form1.Text_r1.text = r1
          Form1.Text_tluong.text = trongluong
          Form1.Text_dientich.text = dientich
         Form1.Text_Ix.text = Ix
         Form1.Text_Wx.text = Wx
         Form1.Text_ix1.text = ix1
         Form1.Text_Iy.text = Iy
         Form1.Text_Wy.text = Wy
         Form1.Text_iy1.text = iy1
        ' Đóng bảng cơ sở dữ liệu.
        rs.Close
       db.Close
  End Sub
```

***B5: Tạo các modul vẽ chi tiết***

**a/ Modul tạo lớp mới**

```
  Public Sub Layer_change(layerName As String)
     Set LayerObj = acaddoc.Layers(layerName)
```

```
    acaddoc.ActiveLayer = LayerObj
  End Sub
```

**b/ Modul mở một bản vẽ**

```
  Public Sub OpenTemp(TempName As String)
    Dim DwgName As String
    DwgName = App.Path & "\" & TempName
    If Dir(DwgName) <> "" Then
      acadapp.Documents.Open DwgName
    Else
      MsgBox "File " & DwgName & " does not exist."
      Exit Sub
    End If
    Set acaddoc = acadapp.ActiveDocument
  End Sub
```

**c/ Modul tạo đường tâm**

```
  Public Sub line_center(x1 As Double, y1 As Double, x2 As Double, y2 As Double)
    ' Gọi lớp đường tâm
    Call Layer_change("center")
    Dim points(0 To 3) As Double
    Dim plineObj As AcadLWPolyline
    points(0) = x1: points(1) = y1
    points(2) = x2: points(3) = y2
    Set plineObj = acaddoc.ModelSpace.AddLightWeightPolyline(points)
  End Sub
```

**d/ Modul ghi kích thước góc**

```
  Public Sub ghi_kich_thuoc_goc(CenterX As Double, CenterY As Double, FPointx
As Double, Fpointy As Double, Lpointx As Double, Lpointy As Double, TextX As
Double, TextY As Double)
    ' Gọi lớp đường ghi kích thước
    Call Layer_change("Dimention")
    Dim angVert(0 To 2) As Double
    Dim FirstPoint(0 To 2) As Double
```

```
    Dim SecondPoint(0 To 2) As Double
    Dim TextPoint(0 To 2) As Double
    Dim dimObj As AcadDimAngular
    angVert(0) = CenterX: angVert(1) = CenterY: angVert(2) = 0
    FirstPoint(0) = FPointx: FirstPoint(1) = Fpointy: FirstPoint(2) = 0
    SecondPoint(0) = Lpointx: SecondPoint(1) = Lpointy: SecondPoint(2) = 0
    TextPoint(0) = TextX: TextPoint(1) = TextY: TextPoint(2) = 0
    Set dimObj = acaddoc.ModelSpace.AddDimAngular (angVert, FirstPoint,
    SecondPoint, TextPoint)
End Sub
```

**e/ Modul tạo đường thẳng**

```
Public Sub drawline(x1 As Double, y1 As Double, x2 As Double, y2 As Double)
    Call Layer_change("0")
    Dim points(0 To 3) As Double
    points(0) = x1: points(1) = y1
    points(2) = x2: points(3) = y2
    Set plineObj = acaddoc.ModelSpace.AddLightWeightPolyline(points)
End Sub
```

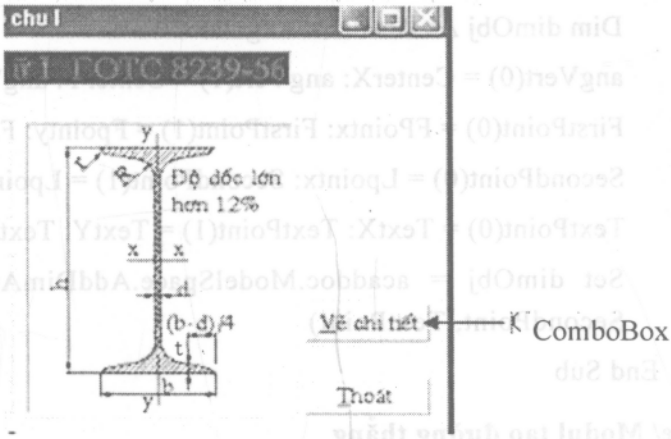
**f/ Modul tạo chữ**

```
Public Sub M_text(DiemchenX As Double, DiemchenY As Double, Do_rong As
Double, text As String)
    Call Layer_change("Dimention")
    Dim corner(0 To 2) As Double
    Dim width As Double
    Dim str As String
    corner(0) = DiemchenX: corner(1) = DiemchenY: corner(2) = 0#
    width = Do_rong
    str = text
    Set MTextObj = acaddoc.ModelSpace.AddMText(corner, width, str)
End Sub
```

*B6: Xử lý các sự kiện*

**a/ Sự kiện vẽ chi tiết**

Sau khi thực hiện các bước trên, thêm đoạn mã chương trình sau vào **ComboBox** *(Vẽ chi tiết)*.

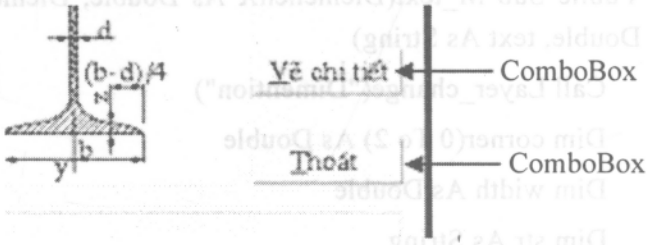



***Hình 1.8.*** Thêm mã cho nút lệnh.

```
Private Sub vechitiet()
  Call Init
  Call OpenTemp("banve.dwt")
    If MsgBox("Ve thep chu I?", vbOKCancel + vbQuestion, "Draw") = vbOK Then
    Call Ve_thep_chu_I
  End If
End Sub
```

**b/ Sự kiện thoát khỏi chương trình**

Thêm đoạn mã chương trình dưới đây vào **ComboBox** *(Thoát)*.




***Hình 1.9.*** Thêm mã cho nút lệnh

```
Private Sub Thoat()
  If MsgBox("Thoat khoi chuong trinh ?", vbOKCancel + vbQuestion, "Exit") =
  vbOK Then
  End
  End If
End Sub
```

**c/ Sự kiện chọn số hiệu thép**

```
Private Sub Cbo_sohieuthep_KeyPress(KeyAscii As Integer)
    KeyAscii = 0
End Sub
```

**d/ Sự kiện load Form**

```
Private Sub Form_Load()
    Data1.Visible = False
    Data1.DatabaseName = App.Path & "\Cosodulieuthepchul.mdb"
    Form1.Cbo_sohieuthep.Enabled = True
    Dim i As String
    i = 10
    Do
      Cbo_sohieuthep.AddItem i
     i = i + 2
   Loop While i <= 18
    Cbo_sohieuthep.AddItem "18a"
    Cbo_sohieuthep.AddItem "20"
    Cbo_sohieuthep.AddItem "20a"
    Cbo_sohieuthep.AddItem "22"
    Cbo_sohieuthep.AddItem "22a"
    Cbo_sohieuthep.AddItem "24"
    Cbo_sohieuthep.AddItem "24a"
    Cbo_sohieuthep.AddItem "27"
    Cbo_sohieuthep.AddItem "27a"
    Cbo_sohieuthep.AddItem "30"
    Cbo_sohieuthep.AddItem "30a"
    Cbo_sohieuthep.AddItem "33"
    Cbo_sohieuthep.AddItem "33a"
    Cbo_sohieuthep.AddItem "36"
    Dim i1 As String
    i1 = 45
```

```
  Do
    Cbo_sohieuthep.AddItem i1
    i1 = i1 + 5
  Loop While i1 <= 70
  Cbo_sohieuthep.AddItem "70a"
  Cbo_sohieuthep.AddItem "70b"
End Sub
```

## Chương 2

# TỔNG QUAN VỀ VBA TRONG MÔI TRƯỜNG ACAD

Chương này trình bày các kiến thức cơ sở về môi trường VBA trong ACAD giúp độc giả nhanh chóng tiếp cận môi trường lập trình VBA trong môi trường ACAD.

### 2.1. KHÁI NIỆM VBA

VBA - Viết tắt của ***"Visual Basic for Application"*** được dùng trong nhiều năm nay và VBA là mở rộng của ngôn ngữ lập trình **Visual Basic (VB)**. Mặc dù cả hai cùng một kiểu mã và làm việc trên giao diện Windows. Nhưng VBA là một phần (*tập hợp con*) của VB và cho phép xây dựng các modul phần mềm ứng dụng trong ACAD.

+ VBA là một trong những ngôn ngữ lập trình để phát triển môi trường thiết kế ACAD cung cấp nhiều ứng dụng và khả năng giống như VB.
+ Sự khác nhau duy nhất giữa VB và VBA là VBA chạy trực tiếp trên môi trường ACAD hay nói cách khác VBA rất tiện ích trong môi trường lập trình ACAD còn VB chạy độc lập trong trình làm việc riêng.

* Sự thuận lợi khi dùng VBA trong ACAD

- *VBA là một ngôn ngữ lập trình dễ học, dễ sử dụng.*
- *VBA có tiến trình chạy cùng với ACAD do đó việc biên dịch và chạy chương trình nhanh.*
- *Thiết kế giao diện nhanh, hiệu quả, điều này cho phép người sử dụng phát triển, tiếp cận các ứng dụng ban đầu nhanh và có được ngay thông tin phản hồi trong lúc thiết kế.*
- *Kết quả của chương trình có thể là bản vẽ độc lập hoặc liên kết với các bản vẽ khác. Sự lựa chọn này cho phép phát triển các ứng dụng tạo ra tính mềm dẻo cũng như sự phân phối các ứng dụng.*

Đầu tiên bản thân ACAD là một phần mềm được thiết lập rất nhiều đối tượng gói trọn trong thực thể ACAD *(dữ liệu và dòng lệnh)*, bởi vì ACAD được thiết kế như một hệ thống mở với các ứng dụng và nhiều cấp độ khác nhau. Sự hiểu biết sâu sắc về ACAD là vô cùng quan trọng trong việc sử dụng VBA có hiệu quả. Nếu sử dụng VL để điều khiển ACAD thì phải có sự hiểu biết thật sự sâu sắc về khả năng làm việc của ACAD. Tuy nhiên người lập trình sẽ thấy đơn giản hơn rất nhiều khi dùng VB&VBA vì khi đó có thể xây dựng các hàm vẽ riêng mà không nhất thiết phải tuân theo cú pháp như lệnh vẽ của ACAD, khác hẳn với VL.

Các trình điều khiển của ACAD cho phép truy cập trực tiếp đến các đối tượng của ACAD. Khi VBA & VB ghép nối với các trình điều khiển ACAD người lập trình và người sử dụng sẽ thấy được các tiện ích sau:

- *Tốc độ trong quá trình chạy chương trình với VBA&VB các trình điều khiển nhanh hơn so với VL.*
- *Ngôn ngữ và môi trường phát triển dễ sử dụng, dễ cài đặt với ACAD.*
- *Thao tác giữa các trình điều khiển của Windows và VBA được thiết kế sử dụng với nhiều ứng dụng trong các phiên bản của Windows khác nhau.*
- *Tiếp cận ví dụ đầu tiên một cách nhanh chóng giao thức ghép nối của VBA cung cấp một môi trường hoàn hảo cho những người ứng dụng lần đầu. Đặc biệt những ứng dụng này có thể phát triển ở những ngôn ngữ khác.*
- *Trên thế giới có hàng triệu người lập trình bằng VB & VBA. Vì chúng là một môi trường mở của ACAD nó giúp cho việc phát triển các ứng dụng theo yêu cầu của khách hàng và tốt hơn các ứng dụng của người sử dụng.*

* Hạn chế của VBA-ACAD

ACAD-VBA đòi hỏi cấu hình hệ thống đối với Windows NT phải từ 4.0 trở lên mới cài được ACAD-VBA, Windows 95, hoặc Windows 98 phải có thủ tục đặc biệt từ nhà cung cấp dịch vụ Microsoft.

## 2.2. CÁC LỆNH ACAD VỚI VBA

Trong môi trường ACAD các lệnh sau đây được sử dụng để thực hiện các yêu cầu và các chức năng với VBA: VBALOAD, VBAUNLOAD, VBARUN, VBAMAN, VBAIDE ...

### 2.2.1. VBALOAD

**VBALOAD** là câu lệnh quan trọng khi ta lựa chọn làm việc với VBA, lệnh này sẽ nạp một file vào trong môi trường của ACAD, cũng giống như VL và ObjecARX được nạp vào trong môi trường ACAD.

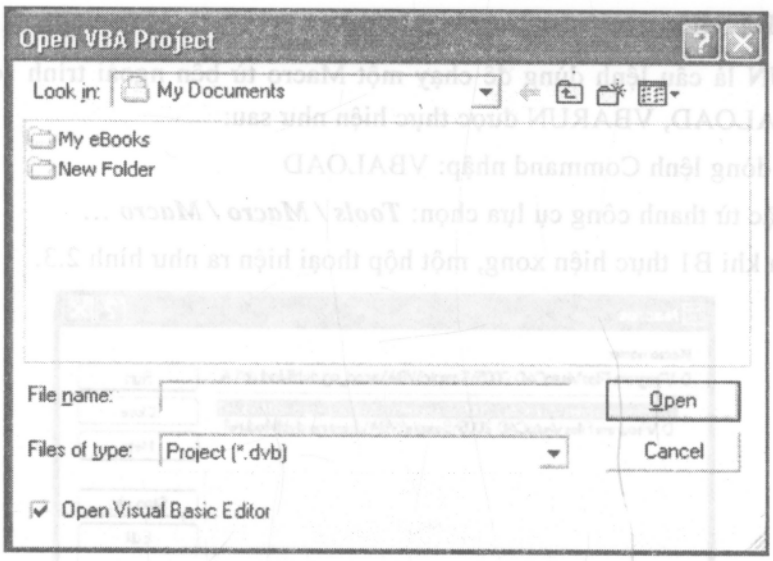
***Chúng ta xem VBALOAD được sử dụng như thế nào?***

VBALOAD có thể được sử dụng theo hai cách, có thể là từ hộp thoại hoặc là từ dòng lệnh.

***B 1:*** Từ dòng lệnh command nhập: VBALOAD

Hoặc từ thanh công cụ lựa chọn: ***Tools/ Macro/ Load Project***... Nếu không có lựa chọn này thì đừng có quá lo lắng, hộp thoại Macro sẽ hiển thị lên màn hình.

***B2:*** Sau khi đã thực hiện xong B1, ta thấy một hộp thoại xuất hiện trên màn hình như hình 2.1.

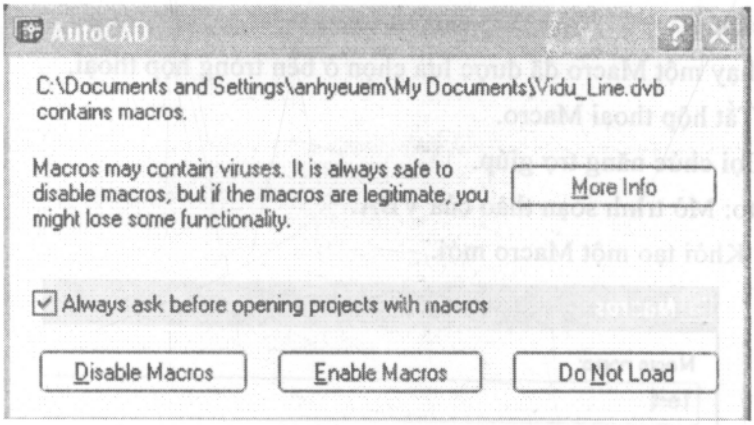



*Hình 2.1.* Hộp thoại mở một Project.

Từ hộp công cụ thực hiện các thao tác, thao tác chính là việc mở các dự án của VBA hoặc các file dự án của VBA có biểu tượng 


***B3:*** Đường dẫn đến file dự án đã được hiển thị trong hộp thoại sau đó lựa chọn dự án cần làm việc.

***B4:*** Sau khi bạn mở dự án để làm việc thì một hộp thoại thông báo được hiển thị như hình 2.2.




*Hình 2.2.* Hộp thoại khởi tạo Marcos.

Nếu kích chuột vào nút Enable Macro khi chắc chắn rằng Macro đó đã được khởi tạo. Nếu không chắc chắn Macro đó đã được khởi tạo hay chưa thì có thể kích vào cả hai nút (*Disable Macros hoặc Do Not Load*).

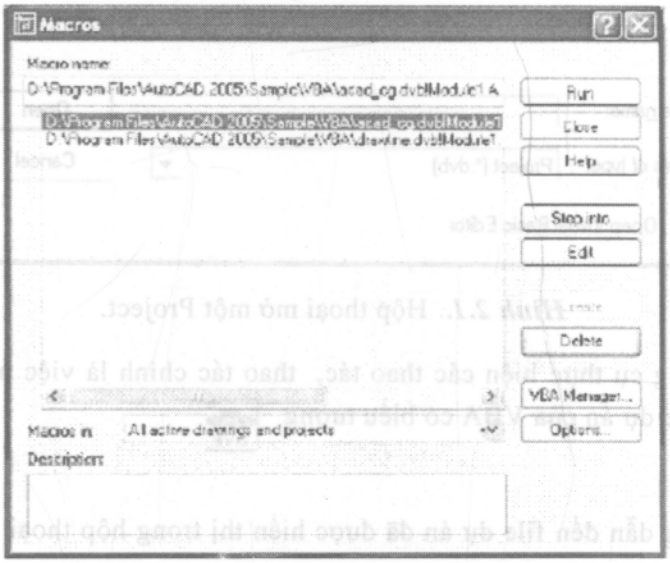
### 2.2.2. VBARUN

**VBARUN** là câu lệnh dùng để chạy một Macro từ bên ngoài trình soạn thảo. Cũng giống như VBALOAD, VBARUN được thực hiện như sau:

***B1:*** Từ dòng lệnh Command nhập: VBALOAD

Hoặc từ thanh công cụ lựa chọn: ***Tools / Macro / Macro*** …

***B2:*** Sau khi B1 thực hiện xong, một hộp thoại hiện ra như hình 2.3.




***Hình 2.3.*** Mở một Marco.

Ta thấy có rất nhiều nút lệnh trên hộp thoại này, sau đây ta sẽ đi tìm hiểu tất cả các chức năng của chúng.

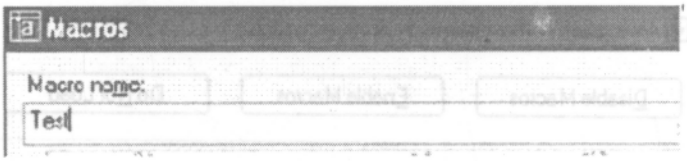
Run: Chạy một Macro đã được lựa chọn ở bên trong hộp thoại.

Close: Tắt hộp thoại Macro.

Help: Gọi chức năng trợ giúp.

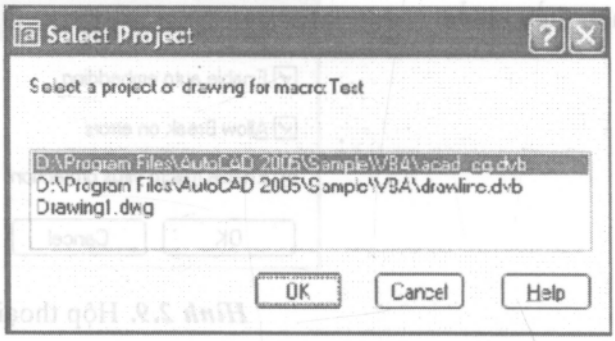
Step into: Mở trình soạn thảo của VBA.

Create: Khởi tạo một Macro mới.



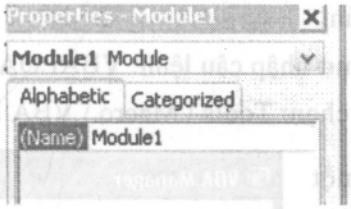

***Hình 2.4.*** Macros.

Sau khi nhập tên xong thì phải lựa chọn một dự án để thêm điều khiển Macro vào.



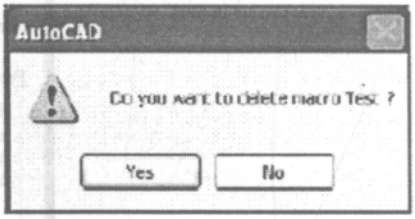

*Hình 2.5.* Hộp thoại Select Project

Modul mới được tạo ra có tên mặc định là Model1.



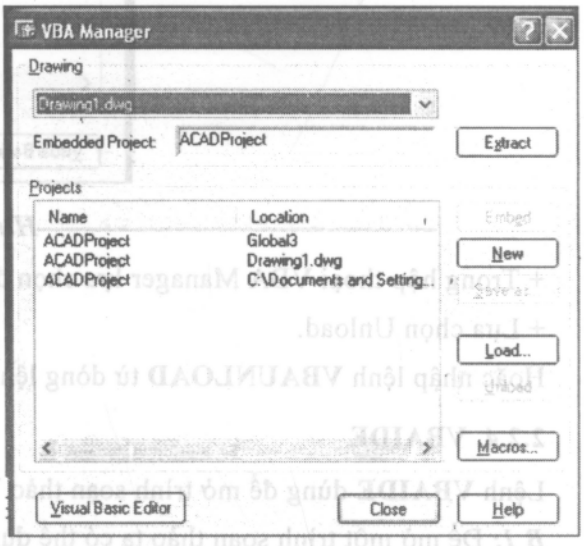

*Hình 2.6.* Properties.

Chọn Delete: Xoá Macro.



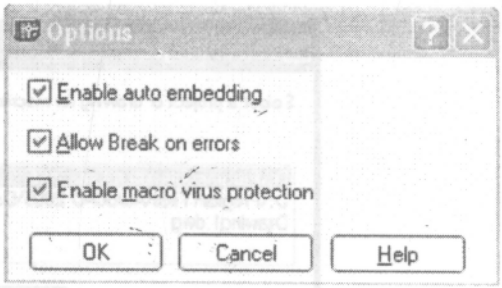

*Hình 2.7.* Xóa Macro

VBA Manager: Mở hộp thoại Manager.




*Hình 2.8.* Hộp thoại VBA Manager.

Options: Các lựa chọn điều khiển




*Hình 2.9.* Hộp thoại Options.

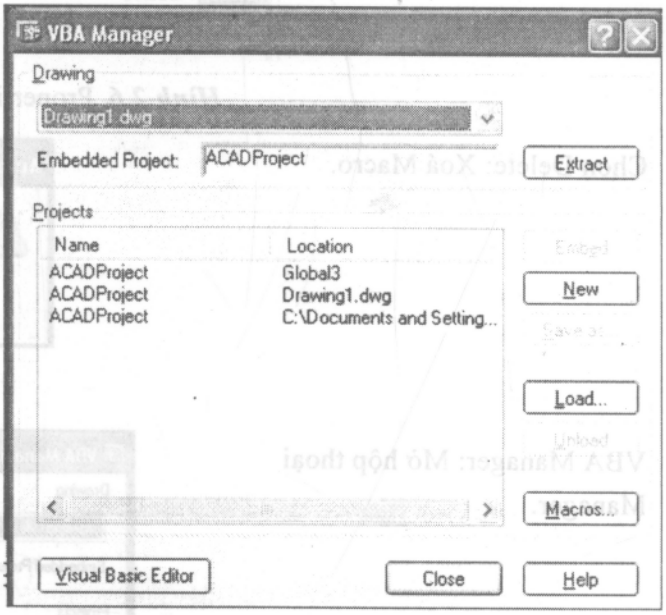
### 2.2.3. VBAMAN

**VBAMAN** dùng để hiển thị hộp thoại VBA Manager, cho phép thực hiện các việc như: Tạo, nạp, tắt, nhúng các dự án.

***B1:*** Từ dòng lệnh **Command** nhập câu lệnh: VBALOAD ↵

Hoặc từ thanh công cụ lựa chọn: Tools \ Macro \ VBA Manager ...

***B2:*** Sau khi B1 hoàn thành, một hộp thoại VBA Manager xuất hiện trên màn hình đồ hoạ hình 2.10.




*Hình 2.10.* Hộp thoại VBA Manager.

+ Trong hộp thoại VBA Manager lựa chọn đối tượng muốn thoát.

+ Lựa chọn Unload.

Hoặc nhập lệnh **VBAUNLOAD** từ dòng lệnh **Command**.

### 2.2.4. VBAIDE

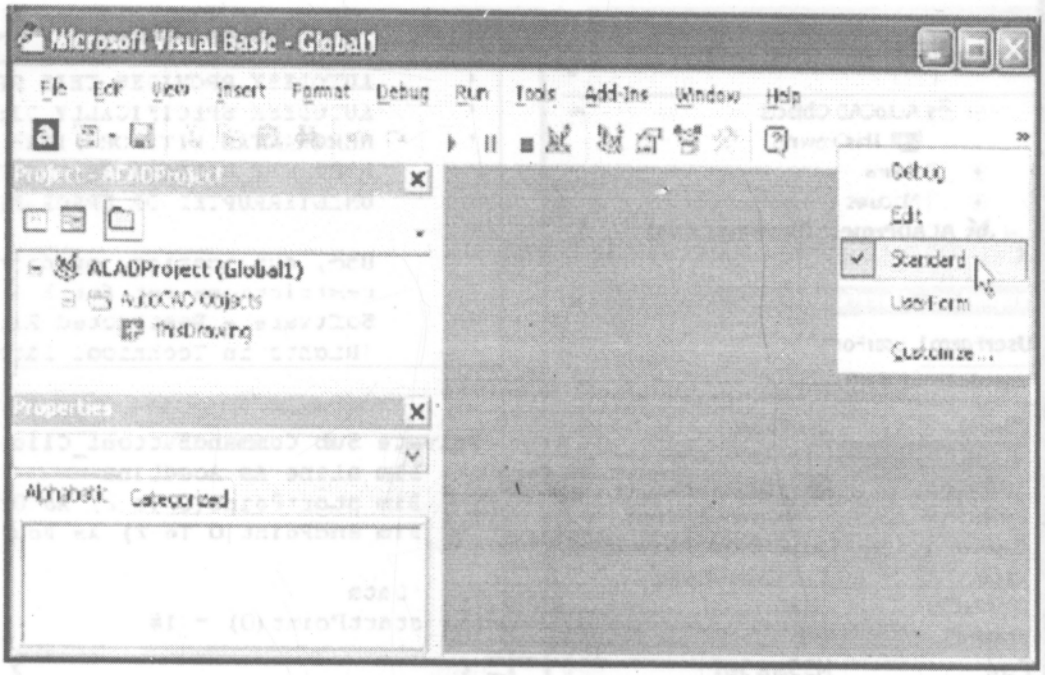
Lệnh **VBAIDE** dùng để mở trình soạn thảo VBA.

***B 1:*** Để mở một trình soạn thảo ta có thể dùng dòng lệnh hoặc từ thanh công cụ.

+ Từ thanh công cụ Command line gõ lệnh: VBA IDE.

+ Hoặc từ thanh công cụ: ***Tools Menu \ Macro \ Visual Basic Editor***.

Sau khi thực hiện lệnh trên ta thấy cửa sổ soạn thảo như hình 2.11.

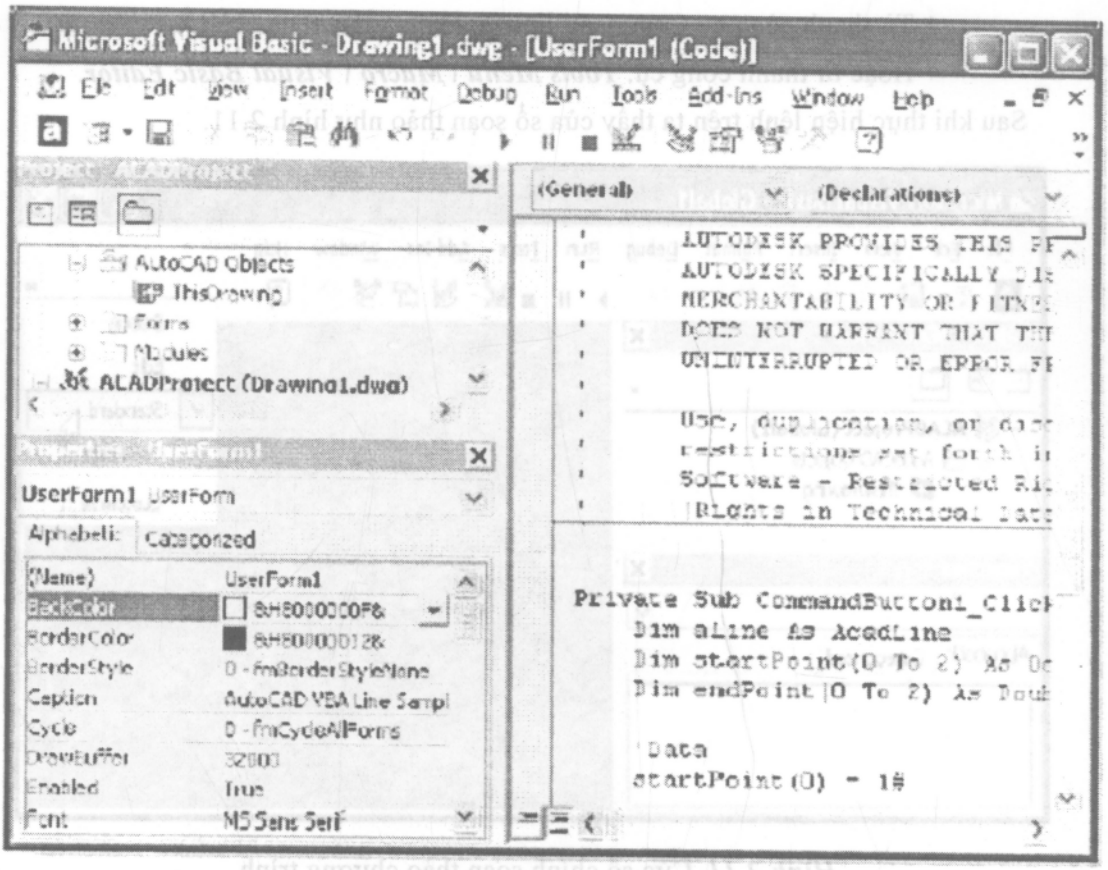



***Hình 2.11.*** Cửa sổ chính soạn thảo chương trình.

> ***Chú ý:*** *Để gọi tự động VBA IDE khi ACAD được gọi bạn cần đưa dòng lệnh:* ***acadvba.arx*** *vào trong file: acad.arx*

## 2.3. TRÌNH SOẠN THẢO VISUAL BASIC

Phần lớn thời gian lập trình là dành cho phần soạn thảo chương trình. Do đó trình soạn thảo là nơi mà người lập trình tốn phần lớn thời gian để làm việc (viết mã lệnh chương trình, thêm và hiệu chỉnh các giao diện, các nút lệnh điều khiển v.v... hình 2.12).




*Hình 2.12.* Cửa sổ trình soạn thảo.

### 2.3.1. Môi trường phát triển tích hợp - IDE

IDE là tên viết tắt của môi trường phát triển tích hợp *(Intergrated Development Environment)* để tạo ra các chương trình lập trình của VBA. IDE của VBA là nơi chứa các công cụ và cửa sổ để tạo ra chương trình. Mỗi phần của IDE có các tính năng ảnh hưởng đến các hoạt động lập trình khác nhau. Thanh menu cho phép quản lý trên toàn bộ ứng dụng.

Hiệu chỉnh đề án với VBA IDE

Một **Project** được nạp trong môi trường **ACAD**, thì có thể chỉnh mã, giao diện và thư viện mà dự án sử dụng, có thể dịch, chạy dự án từ VBA IDE. Môi trường phát triển tích hợp này dùng để viết mã lệnh, kiểm tra, chạy thử đề án. Một chương trình VBA có thể gồm nhiều thủ tục, biểu thức, hàm, các biến dữ liệu v.v...

### 2.3.2. Cửa sổ Explorer

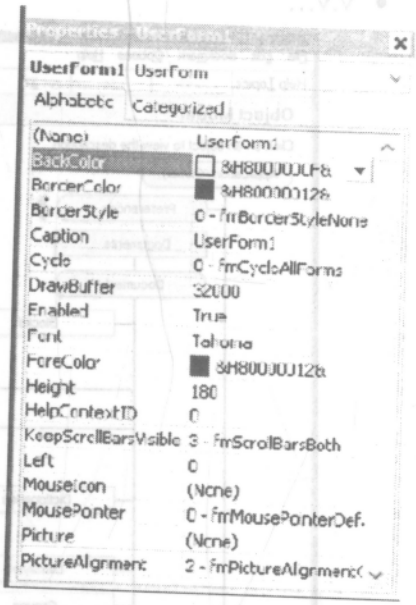
Cửa sổ **Explorer** được hiển thị bên góc phải của màn hình, giúp lựa chọn các tập tin trong đề án và truy cập chúng dưới dạng biểu mẫu hoặc chương trình thiết kế, hình 2.13.


*Hình 2.13.* Cửa sổ Explorer.

### 2.3.3. Cửa sổ thuộc tính (*Properties*)

Cửa sổ này cho phép lập trình viên xem xét và sửa đổi các thuộc tính của biểu mẫu và các kiểu điều khiển lúc thiết kế hình 2.14 minh hoạ.

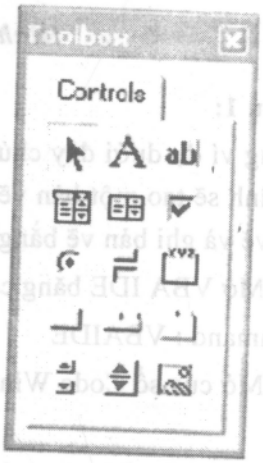
Phần trên cửa sổ là danh sách các đối tượng. Đối tượng được chọn trong danh sách sẽ có thuộc tính hiển thị trong phần bên dưới của của sổ. Ở hình 2.14 biểu mẫu UsesForm1 đang được chọn. Một biểu mẫu có khoảng 40 thuộc tính được hiển thị trong lúc thiết kế, đồng thời ta có thể truy nhập một số thuộc tính khác trong lúc lập trình.




*Hình 2.14.* Cửa sổ Properties.

### 2.3.4. Hộp công cụ (ToolBox)

Để đặt một hộp văn bản hay nút lệnh vào biểu mẫu, đơn giản chỉ là chọn và nhấp chuột. Tất cả các nút lệnh điều khiển nội tại chứa trong *hộp công cụ (ToolBox)* thường được hiển thị ở bên trái màn hình và có biểu tượng 

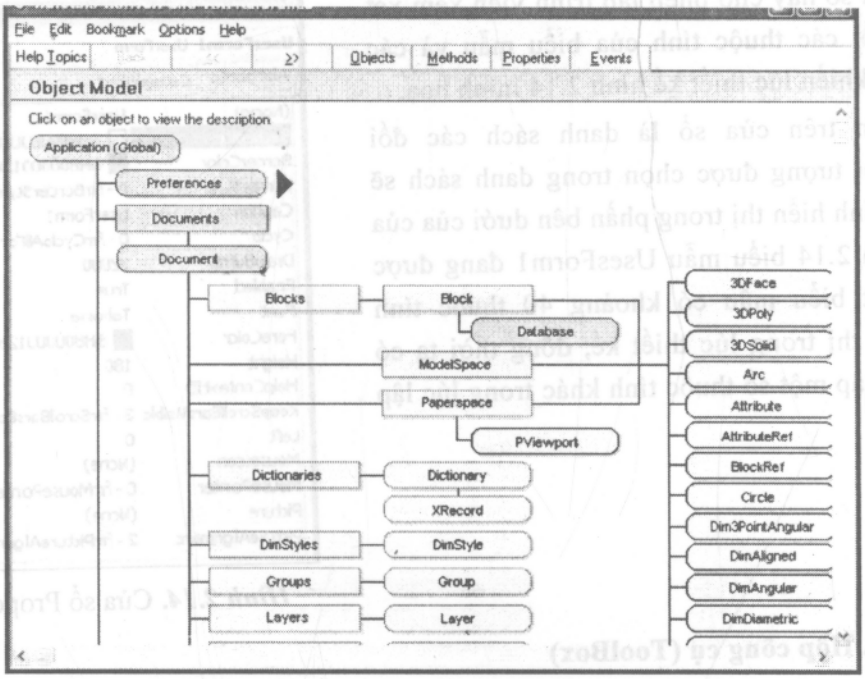



*Hình 2.15.* Hộp công cụ.

## 2.4. CÁC KIỂU ĐỐI TƯỢNG CỦA ACAD

Trong ACAD bao gồm nhiều đối tượng và các thuộc tính của chúng.

- Các đối tượng đồ hoạ như : Đường thẳng, cung tròn, chữ...
- Các kiểu đường và các kiểu kích thước.
- Các cấu trúc như : Layer, các khối Blocks, các Group.
- Hiển thị bản vẽ trên không gian vẽ và không gian giấy.
- Bản vẽ ACAD và các ứng dụng của ACAD.
- v.v...




*Hình 2.16.* Các đối tượng của ACAD.

**Ví dụ 1:**

Trong ví dụ dưới đây chúng ta sẽ viết một chương trình đơn giản trong ACAD VBA. Chương trình sẽ tạo một bản vẽ ACAD mới sau đó thêm một dòng chữ "Hello World " vào trong bản vẽ và ghi bản vẽ bằng VBA. Dưới đây là trình tự các bước:

***B1:*** Mở VBA IDE bằng cách nhập vào dòng lệnh của ACAD Command.

Command : VBAIDE

***B2:*** Mở của sổ Code Window bằng cách chọn Code Option từ View menu trong VBA IDE.

***B3:*** Tạo một thủ tục mới trong dự án bằng cách chọn Procedure Option từ ***Insert menu*** trong VBA IDE.

***B4:*** Khi đó có lời nhắc xuất hiện, nhập tên như HelloWorld, và khai báo ở dạng là Public.

***B5:*** Chọn OK.

***B6:*** Thêm đoạn mã chương trình vào giữa hai lệnh sau:

Public Sub HelloWorld() và End Sub

*' Đoạn mã trên có chức năng mở một bản vẽ mới*

ThisDrawing.Application.Documents.Add

***B7:*** Nhập đoạn mã chương trình dưới đây chèn vào B 6

*' Điểm chèn có kiểu dữ liệu Double*

Dim Diemchen (0 To 2) As Double

*' Khai báo chiều cao chữ có kiểu dữ liệu Double*

Dim Chieucaochu As Double

*'Khai báo biến chữ có kiểu dữ liệu String*

Dim chu As String

Dim Textobj As AcadText

Diemchen(0) = 2

Diemchen(1) = 4

Diemchen(2) = 0

Chieucaochu =1

Chu = " HelloWorld!"

*'Tạo chữ trên không gian vẽ.*

Set Textobj= ThisDrawing.ModelSpace.AddText _

_(chu, Diemchen, Chieucaochu)

***B8:*** Thêm đoạn mã chương trình dưới đây vào B7 khi đó bản vẽ được ghi thành file "Hello.dwg".

ThisDrawing.SaveAs("Hello.dwg")

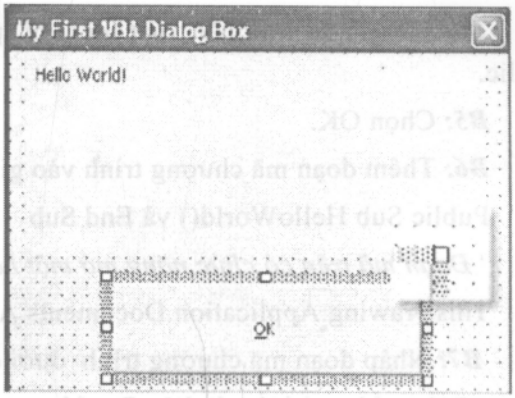
***B9.*** Chạy chương trình bằng cách chọn: Run Sub / UserForm Option từ RUN trên thanh công cụ.

Sau khi chạy chương trình sẽ có dòng chữ "HelloWord" được hiển thị trong MS của bản vẽ CAD. Tên của bản vẽ *Hello.dwg.*

**Ví dụ 2 :**

Chương trình dưới đây với mục đích là làm quen với trình soạn thảo VBA. Trong chương trình sẽ tạo một đối tượng Form và làm việc với đối tượng Form trong chương trình, tác động vào các thuộc tính điều khiển của Form.

*Yêu cầu tạo một Form có giao diện như sau:*




***Hình 2.17.*** Thiết kế giao diện.

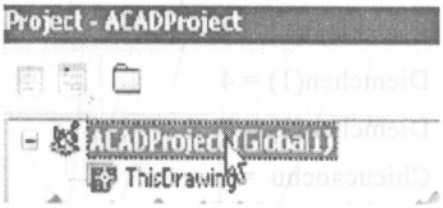
Chạy chương trình khi kích vào nút OK thì thoát khỏi chương trình.

**Các bước thực hiện như sau:**

***B1:*** Dừng tất cả các dự án trong môi trường bằng cách sử dụng lệnh: VBAUNLOAD hoặc VBA Manager.

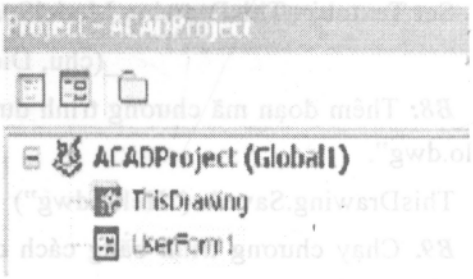
***B2:*** Dùng VBA Manager để tạo một dự án mới và hiển thị trình soạn thảo VBA.

***B3:*** Sau khi chạy trình soạn thảo, chọn tên của dự án như hình 2.18.




***Hình 2.18.***

***B4:*** Sau khi dự án chính được chọn, từ thanh công cụ thêm Form vào ứng dụng như hình 2.19.

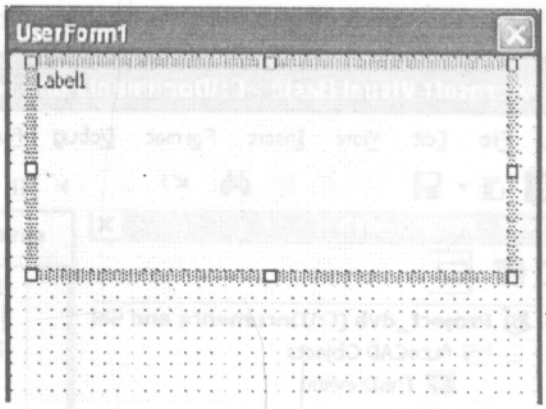



***Hình 2.19.***

***Chú ý: Trong trường hợp hộp thanh công cụ (ToolBox) không hiển thị?***

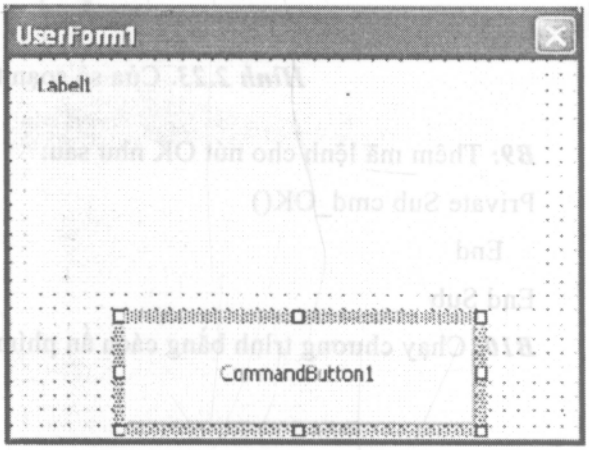
*Nếu như trong trình soạn thảo hộp thanh công cụ không hiện ra, có thể làm hiển thị hộp công cụ bằng cách vào: View menu bar, ToolBox. Hộp công cụ sẽ hiển thị trên màn hình.*

*B5:* Từ ToolBox lựa chọn nhãn (*Label*) để chèn lên Form.



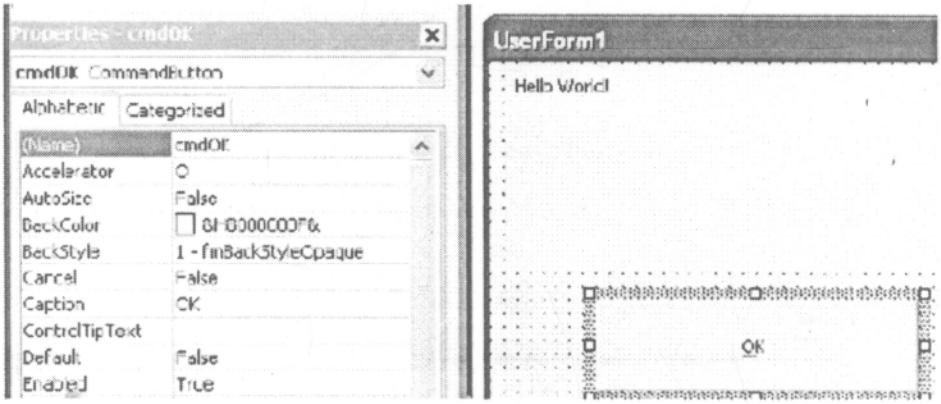

***Hình 2.20.*** Hộp thoại User Form.

*B6:* Từ hộp công cụ thêm nút CommandButton lên Form như hình 2.21.



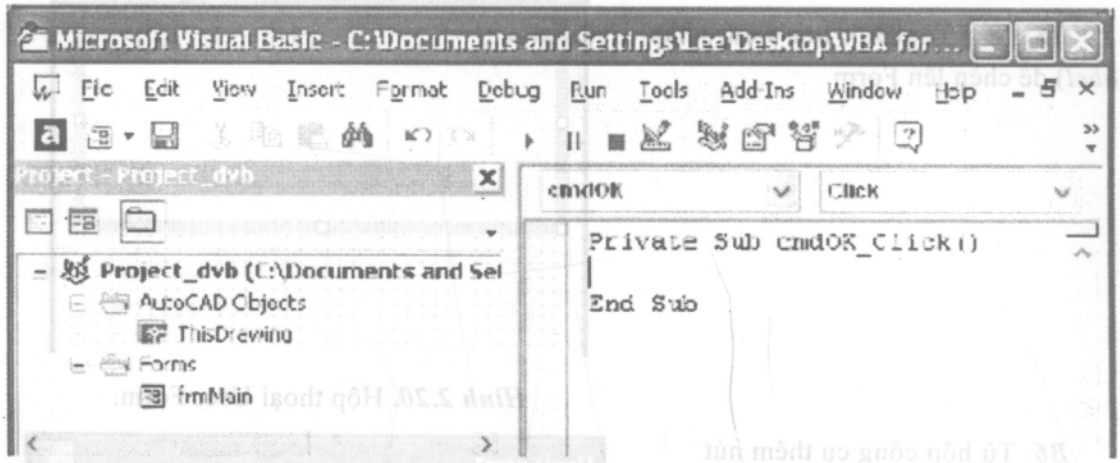

***Hình 2.21.*** Hộp thoại khi thêm nút lệnh Command Button.

*B7:* Mở cửa sổ thuộc tính của các đối tượng và trong cửa sổ thuộc tính đó dùng để thay đổi các thuộc tính của chúng.




***Hình 2.22.*** Thuộc tính của nút lệnh OK.

***B8:*** Viết mã lệnh cho nút OK, bằng cách nhấp đúp chuột vào nút đó như hình 2.23.




***Hình 2.23.*** Cửa sổ soạn thảo chương trình.

***B9:*** Thêm mã lệnh cho nút OK như sau:

```
Private Sub cmd_OK()
   End
End Sub
```

***B10:*** Chạy chương trình bằng cách ấn phím **F5** hoặc từ nút **Run** trên thanh công cụ.

## Chương 3

# TOÁN TỬ VÀ BIỂU THỨC

Chương này trình bày các toán tử, biểu thức, hàm toán học, hàm thời gian của VBA.

## 3.1. CÁC TOÁN TỬ ĐẠI SỐ CỦA VBA

Các toán tử đại số được sử dụng trong câu lệnh của VBA.

***Bảng 3.1.*** Các toán tử đại số của VBA

| Toán tử | Tên | Ý nghĩa | Ví dụ | Kết quả |
|---|---|---|---|---|
| + | Addition | Cộng | 10+5 | 15 |
| - | Negation | Trừ | 0-10 | -10 |
| * | Multiplication | Nhân | 10*5 | 50 |
| / | Division | Chia | 10/5 | 2 |
| \ | Integer Devision | Chia lấy phần nguyên | 11\5 | 2 |
| ^ | Exponentiation | Hàm mũ | 10^5 | 100000 |
| Mod | Moduls | Chia lấy phần dư | 10 Mod 5 | 0 |

## 3.2. TOÁN TỬ SO SÁNH

Sử dụng toán tử so sánh trong biểu thức để so sánh hai hoặc nhiều số, chuỗi Text, các biến hoặc các kết quả hàm.

Nếu câu lệnh là đúng ***(True)*** kết quả của công thức được cho giá trị logic **True**.

Nếu câu lệnh là sai ***(False)*** công thức sẽ trả về giá trị duy nhất **False**. Bảng 3.2 tóm tắt các toán tử so sánh của VBA.

***Bảng 3.2.*** Bảng toán tử so sánh của VBA

| Toán tử | Tên | Ý nghĩa | Ví dụ | Kết quả |
|---|---|---|---|---|
| > | Greater than | Lớn hơn | 10>5 | True |
| < | Less than | Nhỏ hơn | 10<5 | False |
| >= | Greater than or Equal to | Nhỏ hơn hoặc bằng | "a">="b" | False |
| <= | Less than or Equal to | Lớn hơn hoặc bằng | "a"<="b" | True |
| <> | Not or Equal to | Khác | "a"<>"b" | True |

## 3.3. TOÁN TỬ LOGIC

Một biểu thức logic là một biểu thức trả về một kết quả Boolean (***True, False***). VBA nhận một số giá trị tương đương của Boolean.

- Kết quả **False** có thể sử dụng trong biểu thức dưới dạng là 0. Ngược lại có thể sử dụng 0 trong một biểu thức logic dưới dạng **False**.
- Kết quả **True** có thể sử dụng trong một biểu thức dưới dạng là 1.

### 3.3.1. Toán tử And

Sử dụng toán tử **And** để kiểm tra hai toán hạng Boolean để xem chúng có là **True** hoặc **False**.

Ví dụ dưới đây sử dụng toán tử logic **And** để trả về một kết quả từ hai biểu thức.

```
Dim A, B, C, D, MyCheck                 ' Khai báo các biến
A = 10: B = 8: C = 6: D = Null          ' Cho giá trị ban đầu
MyCheck = A > B And B > C               ' Trả về kết quả True
MyCheck = B > A And B > C               ' Trả về kết quả False
MyCheck = A > B And B > D               ' Trả về kết quả Null
```

### 3.3.2. Toán tử Or

Sử dụng toán tử **Or** kiểm tra hai toán hạng Boolean xem một trong trong hai toán hạng là **True** hoặc **False**.

Kết quả = *<giá trị biểu thức 1>* hoặc *<giá trị biểu thức 2>*

**Ví dụ:**

```
Dim A, B, C, D, MyCheck                 ' Khai báo biến
A = 10: B = 8: C = 6: D = Null          ' Định nghĩa các biến
MyCheck = A > B Or B > C                ' Trả về kết quả True
MyCheck = B > A Or B > C                ' Trả về kết quả True
MyCheck = A > B Or B > D                ' Trả về kết quả True
MyCheck = B > D Or B > A                ' Trả về kết quả Null
```

### 3.3.3. Toán tử Xor

**Xor** là toán tử **Or** đặc biệt, dùng kiểm tra hai toán hạng có giá trị đối nhau.

Ví dụ dưới đây sử dụng toán tử logic **Xor** để đưa ra một kết quả từ hai biểu thức.

```
Dim A, B, C, D, MyCheck                 ' Khai báo các biến
A = 10: B = 8: C = 6: D = Null          ' Cho giá trị ban đầu
MyCheck = A > B Xor B > C               ' Trả về kết quả False
MyCheck = B > A Xor B > C               ' Trả về kết quả True
MyCheck = B > A Xor C > B               ' Trả về kết quả False
MyCheck = B > D Xor A > B               ' Trả về kết quả Null
```

### 3.3.4. Toán tử Not

Toán tử **Not** tương đương logic của toán tử phủ định. Toán tử **Not** sẽ trả về đối của một toán hạng. Thứ tự ưu tiên mà VBA sử dụng được xác định bởi các toán tử biểu thức khác nhau.

Bảng giá trị của toán tử **Not**:

| Biểu thức | Kết quả |
|---|---|
| True | False |
| False | True |

**Ví dụ:** Sử dụng toán tử **Not** dùng để phủ định một biểu thức.

```
Dim A, B, C, D, MyCheck          ' Khai báo các biến
A = 10: B = 8: C = 6: D = Null   ' Cho giá trị ban đầu
MyCheck = Not(A > B)             ' Cho kết quả False
MyCheck = Not(B > A)             ' Cho kết quả True
MyCheck = Not(C > D)             ' Cho kết quả Null
```

## 3.4. SỰ ƯU TIÊN TOÁN TỬ

Khi biểu thức đơn giản chỉ có hai giá trị và một toán tử thì không cần xét sự ưu tiên của các toán tử. Tuy nhiên khi biểu thức phức tạp đòi hỏi phải phối hợp nhiều toán tử khi đó phải xét thứ tự ưu tiên của các toán tử *(toán tử nào thực hiện trước, toán tử nào thực hiện sau)*.

V BA thứ tự ưu tiên của các toán tử như sau:

***Bảng 3.3.*** Thứ tự ưu tiên của VBA

| Toán tử | Nội dung | Thứ tự ưu tiên |
|---|---|---|
| ^ | Exponentiation | 1 |
| - | Negation | 2 |
| * and / | Multiplication and Division | 3 |
| \ | Integer Devision | 4 |
| Mod | Moduls | 5 |
| + and - | Addition and Subtraction | 6 |
| & | Concatenation | 7 |
| =<> <=>=<> | Comparison | 8 |
| And Eqv Imp Or Xor Not | Logical | 9 |

## 3.5. CÁC HÀM TOÁN HỌC CỦA VBA

Các toán hạng sử dụng trong các biểu thức số thường là các giá trị số hoặc các biến được khai báo dưới dạng kiểu dữ liệu trong VBA. Tuy nhiên VBA có các hàm toán học được xây dựng sẵn mà các biểu thức được sử dụng dưới dạng toán hạng.

| Hàm | Kết quả trả về |
|---|---|
| Abs (a) | Giá trị tuyệt đối của a. |
| Atn (a) | Actang của a có đơn vị *radians*. |
| Cos (a) | Cosin của a có đơn vị *radians*. |
| Exp (a) | Tăng đến số mũ của a. |
| Fix (a) | Phần nguyên của a. |
| Hex(a) | Giá trị dưới dạng hệ lục phân của a. |
| Int (a) | Phần nguyên của a. |
| Log (a) | Logarit tự nhiên của a. |
| Oct (a) | Giá trị hệ 8, dưới dạng **variant** của a. |
| Rnd (a) | Cho một số ngẫu nhiên của a. |
| Sgn (a) | = 1 khi a > 0<br>= 0 khi a = 0<br>= –1 khi a < 0 |
| Sin (a) | Sin của a có đơn vị *radians*. |
| Tan (a) | Tang của a có đơn vị *radians*. |

**Ví dụ:**

- Sử dụng hàm Abs dùng trả về giá trị của một số.

```
Dim MyNumber
MyNumber = Abs(50.3)          'Trả lại kết quả 50,3
MyNumber = Abs(-50.3)         'Trả lại kết quả 50,3
```

- Dùng hàm **Atn** để tính giá trị của số pi.

```
Dim pi
pi = 4 * Atn(1)
```

- Cách dùng hàm **Sgn**:

```
Dim MyVar1, MyVar2, MyVar3, MySign
MyVar1 = 12: MyVar2 = -2.4: MyVar3 = 0
MySign = Sgn(MyVar1)          ' Cho giá trị 1
MySign = Sgn(MyVar2)          ' Cho giá trị -1
MySign = Sgn(MyVar3)          ' Cho giá trị 0
```

• Ví dụ này sử dụng hàm **Rnd** để tạo một số ngẫu nhiên từ 1 đến 6.

```
        Dim MyValue                    ' khai báo biến MyValue
MyValue = Int((6 * Rnd) + 1)           ' tạo một số ngẫu nhiên từ 1 đến 6
```

• Thủ tục tính **arcsin(x)**:

```
Public Function ArcSin(x As Double) As Double
   Dim tg As Double
   If 1 > x Then
      tg = Atn(x / Sqr(-x * x + 1))
   Else
      tg = 0
   End If
   ArcSin = tg
End Function
```

• Thủ tục tính **arcos(x)**:

```
Public Function ArcCos(x As Double) As Double
   Dim tg As Double
   If 1 > x Then
      tg = Atn(x / Sqr(-x * x + 1) + pi / 2)
   Else
      tg = 0
   End If
   ArcCos = tg
End Function
```

• Thủ tục tính hàm mũ.

```
Public Function Pow(x As Double, n As Integer) As Double
   Dim tg As Double
   Dim i As Double
   tg = 1
   If n = 0 Then
      tg = 1
   Else
      For i = 1 To n
         tg = tg * x
      Next
   End If
   Pow = tg
End Function
```

## 3.6. LÀM VIỆC VỚI CÁC BIỂU THỨC NGÀY THÁNG

Biểu thức ngày tháng là một biểu thức trả về giá trị **Date**. Đối với các toán hạng trong các biểu thức ngày tháng có thể sử dụng một biến được khai báo dưới dạng **Date** hoặc một giá trị **Date** đính kèm ngày tháng trong các dấu thăng như sau:

DateVar = #8/23/05#

Khi làm việc với ngày tháng nên nhớ rằng VBA làm việc với ngày tháng dưới dạng số tuần tự.

VBA cũng có thể trang bị với một số hàm toán học ngày tháng và thời gian.

***Bảng 3.4.*** Các hàm ngày tháng và thời gian của VBA

| Hàm | Giá trị |
|---|---|
| Cdate (exression) | Chuyển đổi giá trị exression thành một giá trị. |
| Date () | Ngày tháng hệ thống hiện hành, dưới dạng Variant. |
| Date$ () | Ngày tháng hệ thống hiện hành, dưới dạng String. |
| DateSerial (year, month, day) | Giá trị ngày, tháng *(Date)* cho year, month và day đặc biệt. |
| DateValue (date) | Giá trị Date cho chuỗi date. |
| Day (date) | Ngày của tháng được cho bởi date. |
| Hour (time) | Thành phần giờ của time. |
| Minute (time) | Thành phần phút của time. |
| Month (date) | Thành phần tháng của date. |
| Now | Ngày tháng và thời gian hệ thống hiện hành. |
| Second (time) | Thành phần giây của time. |
| Time | Thời gian hệ thống hiện hành, dưới dạng Variant. |
| Time$ | Thời gian hệ thống hiện hành, dưới dạng String. |
| Timer | Số giây bắt đầu từ midnight *(nửa đêm)*. |
| TimeSerial (hour, minute, second) | Giá trị ngày tháng cho giờ *(hour)*, phút *(minute )* và giây *(second)* đặc biệt. |
| TimeValue (time) | Giá trị ngày tháng cho chuỗi time. |
| Weekday (date) | Ngày của tuần, dưới dạng một số được cho bởi date. |
| Year (date) | Thành phần năm của date. |

***Ví dụ về sử dụng các hàm ngày tháng:***

• Ví dụ sử dụng hàm **Time** đặt thời gian hệ thống hiện hành.

```
Dim MyTime
MyTime = Time                    ' Thời gian hệ thống hiện hành.
```

• Ví dụ sử dụng hàm **Date** đặt ngày tháng hệ thống hiện hành.

```
Dim MyDate
MyDate = Date                    ' Ngày tháng hệ thống hiện hành.
```

• Ví dụ sử dụng hàm **Hour** tìm thành phần giờ của time.

```
Dim MyTime, MyHour
MyTime = #4:35:17 PM#
MyHour = Hour(MyTime)
```

## 3.7. BIỂU THỨC CHUỖI

Biểu thức chuỗi là biểu thức trả về một giá trị có một kiểu dữ liệu String *(chuỗi)*. Biểu thức chuỗi có thể sử dụng dưới dạng các giá trị chuỗi *(một trong các ký tự được đính kèm trong các dấu ngoặc kép)*, các biến được khai báo dưới dạng String hoặc bất kỳ các hàm xây dựng sẵn nào của VBA sẽ trả về một giá trị chuỗi. Bảng 3.5 tóm tắt các hàm VBA xử lý chuỗi.

***Bảng 3.5.*** Các hàm chuỗi của VBA

| | |
|---|---|
| Asc (string) | Mã ký tự ANSI của ký tự chữ đầu tiên trong String. |
| Chr (charcode) | Ký tự dưới dạng Variant, tương ứng với mã ANSI được cho ra bởi charcode. |
| Chr$ (charcode) | Ký tự, dưới dạng một String, tương ứng với mã ANSI được cho ra bởi charcode. |
| CStr (expression) | Chuyển đổi biểu thức sang dạng giá trị String. |
| InStr (start,string1,string2) | Vị trí ký tự của biến thể đầu tiên của String2 trong String1, bắt đầu ở start. |
| InStr (start,string1,string2) | Vị trí byte của biến thể đầu tiên của String2 trong String1, bắt đầu ở start. |
| InStrRev (string1,string2,start) | Vị trí byte của biến thể cuối cùng của String2 trong String1, bắt đầu ở start. |
| LCase (string) | String được chuyển đổi thành dạng chữ thường dưới dạng một Variant. |

*Bảng 3.5.* (tiếp theo)

| | |
|---|---|
| LCase$ (string) | String được chuyển đổi thành dạng chữ thường dưới dạng một String. |
| Left (string,length) | Length *(số lượng)* ký tự từ String ở bên trái nhất, dưới dạng một Variant. |
| Left$ (string,length) | Length *(số lượng)* ký tự từ String ở bên trái nhất, dưới dạng một Variant. |
| LeftB (string,length) | Length byte ở bên trái nhất từ String, dưới dạng một Variant. |
| LeftB$ (string,length) | Length byte ở bên trái nhất, dưới dạng một String. |
| Len (string) | Số lượng ký tự trongString. |
| LenB (string) | Số byte trong String. |
| LTrim (string) | Một chuỗi, dưới dạng Variant, không có các khoảng trắng ở đầu trong String |
| Ltrim$ (string) | Một chuỗi, dưới dạng String, không có các khoảng trắng ở đầu trong String. |
| Mid (string,start,length) | Length ký tự, dưới dạng một Variant, String bắt đầu từ start. |
| Mid$ (string,start,length) | Length ký tự, dưới dạng một String, từ string bắt đầu từ start. |
| MidB (string,start,length) | Length byte, dưới dạng Variant, String bắt đầu từ start. |
| MidB$ (string,start,length) | Length byte, dưới dạng String, string bắt đầu từ start. |
| Right (string) | Length ký tự bên phải nhất String, dưới dạng một Variant. |
| Right$ (string) | Length ký tự bên phải nhất String, dưới dạng một String. |
| RightB (string) | Length byte bên phải nhất String, dưới dạng một Variant. |
| RightB$ (string) | Length byte bên phải nhất String, dưới dạng một String. |
| RTrim (string) | Một chuỗi, dưới dạng Variant, không có các khoảng trắng ở phần cuối trong String. |
| RTrim$ (string) | Một chuỗi, dưới dạng String, không có các khoảng trắng ở phần cuối trong String. |
| Trim (string) | Một chuỗi, dưới dạng Variant, không có các khoảng trắng ở phần đầu và phần cuối trong String. |
| Trim$ (string) | Một chuỗi, dưới dạng String, không có các khoảng trắng ở phần đầu và phần cuối trong String. |
| Space (Number) | Một chuỗi, dưới dạng Variant, với số khoảng trắng number. |

*Bảng 3.5.* (tiếp theo)

| | |
|---|---|
| Space$ (Number) | Một chuỗi, dưới dạng String, với số khoảng trắng number. |
| Str (Number) | Đổi thành chuỗi dưới dạng một Variant của number. |
| Str$ (Number) | Đổi thành chuỗi dưới dạng một String của number. |
| StrComp (string2,string2,compare) | Giá trị cho biết kết quả của phép so sánh String1 và String2. |
| StrConv (string,conversion) | Chuỗi được chuyển đổi thành dạng khác, dưới dạng được chỉ định bởi conversion *(chẳng hạn vbUpperCase, vbLowerCase và vbProperCase).* |
| String (number,character) | Character dưới dạng một Variant, được lặp lại number lần. |
| String$ (number,character) | Character dưới dạng một String, được lặp lại number lần. |
| UCase (string) | String được chuyển đổi thành dạng chữ hoa, dưới dạng một Variant. |
| Ucase$ (string) | String được chuyển đổi thành dạng chữ hoa, dưới dạng một String. |
| Val (string) | Tất cả các số trong String đều phải là ký tự không số đầu tiên. |

***Ví dụ về sử dụng các hàm của chuỗi:***

• Dùng hàm **CStr** để chuyển từ giá trị số sang kiểu String

```
Dim MyDouble, MyString
MyDouble = 437.324              ' MyDouble kiểu Double.
MyString = CStr(MyDouble)       ' MyString kiểu String "437.324".
```

• Dùng hàm **Ucase** chuyển đổi thành dạng chữ hoa, dưới dạng một String.

```
Dim LowerCase, UpperCase
LowerCase = "Hello World 1234"   ' Ký tự cần chuyển.
UpperCase = UCase(LowerCase)     ' Kết quả "HELLO WORLD 1234".
```

• Dùng hàm **Asc** trả về mã ký tự ANSI của ký tự chữ đầu tiên trong String.

```
Dim MyNumber
MyNumber = Asc("A")             ' Trả về kết quả 65.
MyNumber = Asc("a")             ' Trả về kết quả 97.
MyNumber = Asc("Apple")         ' Trả về kết quả 65.
```

Khi khai báo biến trong một chương trình với từ khoá **Dim** thì biến đó chỉ tồn tại và hoạt động ở trong chương trình đó.

Khi khai báo biến với từ khóa **Public** thì biến đó sẽ tồn tại và có tầm hoạt động trong toàn bộ chương trình.

Khai báo biến cục bộ với từ khoá **Static**, mặc dù biến đó mất đi khi thủ tục chấm dứt, nhưng giá trị của nó vẫn được giữ lại để tiếp tục hoạt động khi thủ tục được gọi trong lần sau.

### 4.1.3. Khai báo ngầm

Khai báo ngầm là không cần khai báo tường minh trước khi sử dụng biến.

*Ví dụ:*

```
Function  Line(point1, point2)
m_Point1= abs(point1)
m_Point2= abs(point2)
```

***Chú ý:*** *Mặc dù cách này rất tiện lợi, nhưng nó có thể gây lỗi nếu nhầm tên biến.*

### 4.1.4. Khai báo biến tường minh

Khai báo biến tường minh để tránh nhầm lẫn khi viết chương trình, để VBA báo lỗi khi gặp một tên biến không khai báo. Ta đặt thêm dòng lệnh :

**Option Explicit**

**Option Explicit** chỉ hoạt động trong một chương trình con. Vì vậy, ta phải thêm dòng này vào mỗi chương trình con của biểu mẫu.

***Tầm hoạt động của biến***

***Bảng 4.1.*** Tầm hoạt động của biến

| Tầm hoạt động | Private | Public |
|---|---|---|
| Thủ tục | Biến chỉ tồn tại hoạt động trong thủ tục. | Không có. |
| Chương trình con | Biến chỉ tồn tại và hoạt động trong chương trình con. | Biến tồn tại và hoạt động trên toàn bộ chương trình. |

### 4.1.5. Khai báo biến Static

Để khai báo các biến cục bộ trong một thủ tục **Static**, đặt từ khoá **Static** vào tên thủ tục như sau:

```
Static Function  Total_Line (m_Lines).
```

Từ khoá **Static** có thể đặt ở đầu thủ tục **Sub** hoặc **Funtion**, kể cả thủ tục xử lý sự kiện hoặc những hàm **Private**.

### 4.1.6. Các kiểu dữ liệu của biến

+ **Kiểu số:** Integer, Long, Double, Curency. Tất cả các biến kiểu số có thể được gán cho nhau và cho biến Variant.

+ **Kiểu Byte:** Được dùng để chứa dữ liệu nhị phân. Tất cả các thao tác trên kiểu Integer có thể thực hiện trên kiểu Byte, ngoại trừ dấu, vì Byte là kiểu không dấu (trong khoảng từ 0-255).

+ **Kiểu Sring:** Ta có thể khai báo chuỗi có chiều dài cố định: Dim EmpName As String * 50. Khi làm việc với chuỗi, cần dùng các hàm Trim và RTrim để cắt bỏ các ký tự trắng không cần thiết.

+ **Kiểu Boolean:** Nên dùng kiểu Boolean được dùng khi có một biến có hai giá trị True/False, Yes/No, On/Off...

+ **Kiểu Object:** Chứa một địa chỉ 4 byte (32 bit) trỏ đến đối tượng trong ứng dụng hiện hành hoặc các ứng dụng khác. Khi khai báo biến đối tượng nên chỉ ra tên lớp tường minh như TextBox, Database thay vì Control hay Object.

+ **Kiểu Variant:** Có thể chứa mọi loại dữ liệu: (Chuỗi, số, thậm chí là mảng... ).

+ **Kiểu mảng (Array):** Mảng là một xâu các biến có cùng tên và cùng kiểu dữ liệu. Mảng có biên trên và biên dưới, các thành phần trong mảng là liên tục giữa hai biến.

## 4.2. LÀM VIỆC VỚI HẰNG

Hằng dùng để chứa dữ liệu tạm thời nhưng nó không thay đổi trong suốt thời gian chương trình hoạt động. Sử dụng hằng số làm cho chương trình sáng sủa và dễ đọc hơn.

### 4.2.1. Khai báo hằng

[Public| Private] Const <*Tên hằng* >[As <*Kiểu dữ liệu*>] =<*Biểu thức*>

*Ví dụ:*

```
Const  pi = 3.1416
Public Const Max_Radiant =100
```

### 4.2.2. Tầm hoạt động của hằng

Cũng giống như biến, hằng có tầm hoạt động tương tự như biến:

- Hằng khai báo trong thủ tục chỉ hoạt động trong thủ tục.
- Hằng khai báo trong một chương trình con thì chỉ hoạt động trong chương trình con đó.

## 4.3. CẤU TRÚC CHỌN

### 4.3.1. Cấu trúc: If... Then

- Trong trường hợp một dòng lệnh:

**If** <*Điều kiện*>**Then**<*dòng lệnh*>

- Nhiều dòng lệnh

**If** <*Điều kiện*>Then

<*Các dòng lệnh*>

**End if**

Dạng cơ bản nhất của lựa chọn là (**True/False**) cũng có thể thấy dưới dạng là (**Yes / No**) hoặc (**On / Off**). Trong trường hợp này chương trình sẽ xét một điều kiện cụ thể, xác định có đúng (**True**) hoặc sai (**False**) và hoạt động theo đó. Nếu điều kiện là **True** VBA thực hiện tất cả các dòng lệnh sau từ khoá **Then**.

***Ví dụ:***

```
If i < 90 Then
  P[i] = 10*i
If j >90 Then
  P[j] = 10*j
```

***Chú ý:*** *Nếu trường hợp một dòng lệnh thì không có **End if**.*

### 4.3.2. Cấu trúc: If... Then...Else

**If** < *Điều kiện 1* > **Then**

*(Khối lệnh 1)*

**ElseIf** <*điều kiện 2*> **Then**

*(Khối lệnh 2)*

...

**Else** *(Khối lệnh n)*

**End if**

Với cấu trúc trên VBA sẽ kiểm tra điều kiện thứ nhất. Nếu điều kiện đó sai, VBA sẽ kiểm tra điều kiện thứ hai cho đến khi điều kiện đúng. VBA thực hiện khối lệnh tương ứng và sau đó thực hiện chương trình ngay sau **End if**.

***Ví dụ:***

```
If strKeyWord = "X" Then
    blnXaxis = True
```

```
    Else
        blnXaxis = False
    End If
```

### 4.3.3. Select Case

**Select Case** được dùng trong trường hợp chương trình của bạn có quá nhiều **ElseIf** được dùng, giúp cho chương trình sáng sủa, dễ đọc và dễ tìm lỗi chương trình. Biểu thức để so sánh được tính toán một lần vào đầu cấu trúc. Sau đó VBA so sánh kết quả biểu thức với từng **Case**. Nếu bằng khi đó chương trình mới thực hiện khối lệnh trong **Case** đó.

**Select Case** *<Biểu thức kiểm tra>*

**Case** *<Danh sách biểu thức 1>*

*(Khối lệnh 1)*

**Case** *<Danh sách biểu thức 2>*

*(Khối lệnh 2)*

.....

**Case Else**

*(Khối lệnh n)*

**End Select**

Trong đó mỗi danh sách biểu thức chứa một hoặc nhiều giá trị. Các giá trị cách nhau bằng dấu phẩy và mỗi khối lệnh có thể chứa từ một cho đến nhiều dòng lệnh.

***Ví dụ:***

```
Function Bonus(performance, salary)
  Select Case performance
    Case 1
      Bonus = salary * 0.1
    Case 2, 3
      Bonus = salary * 0.09
    Case 4 To 6
      Bonus = salary * 0.07
    Case Is > 8
      Bonus = 100
    Case Else
      Bonus = 0
  End Select
End Function
```

## 4.4. CẤU TRÚC LẶP

Vòng lặp cho phép thi hành một đoạn chương trình nhiều lần.

### 4.4.1. Vòng lặp Do... Loop

Thi hành một khối lệnh với số lần lặp không định trước, trong đó một biểu thức điều kiện dùng so sánh quyết định vòng lặp có tiếp tục hay không. Điều kiện phải trả về False 0 hoặc True # 0.

***Do*... Loop** có 4 cú pháp khác nhau:

*** Cú pháp thứ nhất:**

**Do While** *<Điều kiện>*

*<Khối lệnh>*

**Loop**

Kiểm tra điều kiện trước khi vào vòng lặp. Thực hiện câu lệnh chỉ khi điều kiện là True.

***Ví dụ 1:***

Trong thủ tục này đã kiểm tra điều kiện trước khi vào vòng lặp chương trình.

```
Sub FirstWhile()
   counter = 0
   myNum = 20
   Do While myNum > 10
      myNum = myNum – 1
      counter = counter + 1
   Loop
   MsgBox "The loop made" & counter & "repetitions."
End Sub
```

*** Cú pháp thứ hai:**

**Do** *<Khối lệnh>*

**Loop While** *<Điều kiện>*

Kiểm tra điều kiện sau khi chạy vòng lặp và thực hiện khối lệnh khi điều kiện là True.

***Ví dụ 2:***

Trong thủ tục chạy chương trình trước khi kiểm tra điều kiện:

```
Sub LastWhile()
   counter = 0
   myNum = 9
```

```
    Do
        myNum = myNum - 1
        counter = counter + 1
    Loop While myNum > 10
    MsgBox "The loop made " & counter & " repetitions."
End Sub
```

*** Cú pháp thứ ba:**

+ **Do Until** *<Điều kiện>*

*<Khối lệnh>*

**Loop**

***Ví dụ 3:***

Sau kiểm tra điều kiện trước khi vào vòng lặp và thực thi khối lệnh khi điều kiện là **False**:

```
Sub ExitExample()
    counter = 0
    myNum = 9
    Do Until myNum = 10
        myNum = myNum – 1
        counter = counter + 1
        If myNum < 10 Then Exit Do
    Loop
    MsgBox "The loop made " & counter & " repetitions."
End Sub
```

*** Cú pháp thứ tư:**

**Do**

*< Khối lệnh >*

**Loop Until** *<Điều kiện>*

***Ví dụ 4:***

Sau lặp trong khi điều kiện là **False** và có ít nhất một lần thi hành khối lệnh:

```
Sub LastUntil()
    counter = 0
    myNum = 1
    Do
        myNum = myNum + 1
```

```
    counter = counter + 1
  Loop Until myNum = 10
  MsgBox "The loop made " & counter & " repetitions."
End Sub
```

### 4.4.2. For...Next

Đây là vòng lặp thông thường nhất, sử dụng vòng lặp này khi đã biết chính xác số lần lặp.

**For** *<Biến đếm >=<Điểm đầu>* **To** *<Điểm cuối>* [Step *<Bước nhảy>*]

*<Khối lệnh>*

**Next**

> ***Chú ý:***
>
> *Biến đếm, điểm đầu, điểm cuối và bước nhảy là những giá trị số. Bước nhảy có thể là dương hoặc âm. Nếu bước nhảy là dương, điểm đầu phải nhỏ hơn hoặc bằng điểm cuối. Nếu không khối lệnh sẽ không thi hành, nếu bước nhảy là âm, điểm đầu phải lớn hơn hoặc bằng điểm cuối. Nếu bước nhảy (Step) không được chỉ ra thì mặc định là 1*

Đoạn mã chương trình sau sẽ dùng vòng lặp **For... Next** dùng để tính tổng của một dãy số

```
Sub TwosTotal()
  For j = 2 To 10 Step 2
    total = total + j
  Next j
  MsgBox "The total is " & total
End Sub
```

### 4.4.3. For Each...Next

Tương tự như vòng lặp **For.. Next**, nhưng lặp khối lệnh theo số phần tử của một tập các đối tượng hay một mảng thay vì số lần lặp xác định. Dùng trong trường hợp khi không biết chính xác bao nhiêu phần tử trong tập hợp.

**For Each** *<Phần tử >* **In** *<Nhóm>*

*<Khối lệnh>*

**Next** *<Phần tử>*

***Ví dụ:***

```
Sub RoundToZero()
  For Each myObject in myCollection
```

```
        If Abs(myObject.Value) < 0.01 Then myObject.Value = 0
    Next
End Sub
```

#### 4.4.4. Vòng lặp While...Wend

Cũng giống như vòng lặp **Do ..While** nhưng không thể thoát được vòng lệnh bằng lệnh **Exit**. Vì vậy vòng lặp kiểu này chỉ thoát khi biểu thức điều kiện sai.

***Ví dụ:***

Trong đoạn mã chương trình sau sử dụng vòng lặp **While ... Wend** dung để tăng biến đếm. Bên trong vùng lặp được thi hành đến khi giá trị điều kiện thành **True**.

```
Dim Counter                 ' Khai báo biến đếm
Counter = 0                 ' Gán biến đếm ban đầu bằng 0
While Counter < 20          ' Kiểm tra gía trị của biến đếm
   Counter = Counter + 1    ' Tăng biến đếm
Wend                        ' Kết thúc khi biến đếm > 19
Debug.Print Counter         ' Thực hiện việc in số 20 trên màn hình
```

#### 4.4.5. Câu lệnh Go To

- Câu lệnh Go To dùng để bẫy lỗi.

*On Error GoTo ErrorHandler*

Khi có lỗi chương trình sẽ tự động nhảy đến nhãn **ErrorHandler** và thi hành các dòng lệnh ở đó.

| |
|---|
| ***Chú ý:*** *Trong quá trình lập trình ta có thể lồng các cấu trúc với nhau* |

- Thoát khỏi cấu trúc:

Lệnh **Exit** cho phép thoát khỏi vòng lặp **For, Do loop**, thủ tục hoặc hàm. **Exit For** dùng cho vòng lặp **Do...Loop.**

```
Public Sub SubName

  Exit Sub

End Function
```

### 4.5. THAO TÁC VÀ XỬ LÝ DỮ LIỆU

#### 4.5.1. Sử dụng đối tượng DAO (*Data Access Object*)

Sử dụng đối tượng **DAO** để kết nối và xử lý dữ liệu, đặc biệt khi sử dụng **DAO** cho

phép ta có thể thao tác với nhiều loại cơ sở dữ liệu khác nhau bao gồm:

- Access, dBase, FoxPro, và Paradox
- Microsoft SQL Server và Oracle

**DAO** cung cấp các điều khiển hoàn chỉnh để có thể kết nối cơ sở dữ liệu từ VBA.

- Tạo một cơ sở dữ liệu mới hoặc có thể chỉnh sửa một cơ sở dữ liệu hiện thời.
- Thêm các bảng vào cơ sở dữ liệu, định quan hệ giữa các bảng đó, khởi tạo và chạy các truy vấn.
- Thêm, chỉnh sửa và xóa các bản ghi.
- Bảo mật cơ sở dữ liệu.

Ba bước cơ bản khi sử dụng **DAO trong VBA**

- Khai báo thư viện **DAO**.
- Mở cơ sở dữ liệu định làm việc.
- Viết chương trình với các đối tượng của mô hình **DAO**.

### 4.5.2. Khai báo thư viện DAO

Để tạo liên kết (*tham chiếu*) đến các thư viện của các đối tượng ứng dụng khác:

***B1***: Trong trình soạn thảo VBA IDE, mở Tools menu và lựa chọn References.

***B2***: Tìm và lựa chọn thư viện Microsoft DAO Object Library.

***B3***: Kích OK để đóng hộp thoại.

### 4.5.3. Mở cơ sở dữ liệu định làm việc

Để làm việc với dữ liệu trong một cơ sở dữ liệu thì phải mở cơ dở dữ liệu từ VBA. Để mở cơ sở dữ liệu sử dụng phương thức **OpenDatabase** trong không gian làm việc của đối tượng mô hình **DAO**.

Đoạn mã đưới đây mở một cơ sở dữ liệu có tên là :"bang1"

```
Dim db As Database
Set db = DBEngine.Workspaces(0).OpenDatabase("C:\bang1")
```

### 4.5.4. Viết chương trình với các đối tượng mô hình DAO

- Khai báo đối tượng **DAO**

***Ví dụ:***

```
Dim mDb As Database, mRc As Recordset
```

Bây giờ đã tham chiếu được đến thư viện **DAO** đã tạo, mở một cơ sở dữ liệu, chỉnh sửa hoặc truy vấn dữ liệu trong cơ sở dữ liệu đó. Tất cả các đối tượng, phương thức, thuộc tính được định nghĩa bởi đối tượng mô hình **DAO**, các công việc còn lại là thực hiện các truy vấn dữ liệu trong cơ sở dữ liệu.

### 4.5.5. Thủ tục tra cứu cơ sở dữ liệu theo phương pháp ADODB

```
' Khai báo các biến.
Public cnWorkShop As Connection
Public rsItemMaster As Recordset
Public rsIUnit As Recordset
' Thủ tục tra cứu cơ sở dữ liệu.
Public Sub connectDatabase()
Set cnWorkShop = New ADODB.Connection
With cnWorkShop
   .Provider = "Microsoft.JET.OLEDB.4.0"
   .ConnectionString = App.Path & "\Accounts.mdb" 'Write your access
    Database name
   .Open
End With
Set rsItemMaster = New ADODB.Recordset
rsItemMaster.Open "Item", cnWorkShop, adOpenKeyset, adLockOptimistic
Set rsIUnit = New ADODB.Recordset
rsIUnit.Open "Unit", cnWorkShop, adOpenKeyset, adLockOptimistic
End Sub
```

## Chương 5
# QUẢN LÝ MÔI TRƯỜNG CAD

Chương này trình bày các thao tác trong môi trường ACAD, ví dụ như mở bản vẽ, ghi bản vẽ, các chế độ quan sát bản vẽ v.v...

## 5.1. KHỞI TẠO, MỞ, GHI VÀ ĐÓNG BẢN VẼ

### 5.1.1. Khởi tạo một bản vẽ mới

Để tạo một bản vẽ mới sử dụng phương thức **Add**.

*Ví dụ:*

```
Sub NewDrawing ()
    Dim docObj As AcadDocument
    Set docObj  = ThisDrawing.Application.Document.Add
End Sub
```

### 5.1.2. Mở một bản vẽ

Để mở một bản vẽ có sẵn sử dụng phương thức **Open** và có thể đổi tên File hoặc đường dẫn đến bản vẽ ACAD trong hệ thống dữ liệu.

### 5.1.3. Ghi một bản vẽ

Để ghi một bản vẽ có thể sử dụng cả hai phương thức **Save** hoặc **SaveAs**.

*Ví dụ:* Đoạn mã chương trình dưới đây sẽ ghi bản vẽ dưới tên hiện thời và ghi lại dưới một tên mới.

```
Sub SaveActiveDrawing()
    'Ghi bản vẽ dưới tên hiện thời
    ThisDrawing.Save
    'Ghi bản vẽ với tên mới
    ThisDrawing.SaveAs "MyDrawing.dwg"
End Sub
```

Trong quá trình làm việc, để kiểm tra bản vẽ hiện thời trước khi người lập trình muốn thoát khỏi chương trình và kiểm tra lại thông tin đã chắc chắn ghi lại những thay đổi trong bản vẽ đó. Đây là cách an toàn trước khi thoát khỏi ACAD hoặc khi bắt đầu một bản vẽ mới. Sử dụng thuộc tính **Saved** *(ghi)* để đảm bảo chắc chắn bản vẽ hiện thời không chứa bất kỳ sự thay đổi **Unsaved** *(không ghi)* nào.

### 5.1.4. Kiểm tra nếu một bản vẽ không được ghi

Đoạn mã chương trình dưới đây dùng để kiểm tra một bản vẽ đã được ghi lại hay không ghi trước khi muốn thoát khỏi chương trình ACAD.

```
Sub TestIfSave()
    If Not (ThisDrawing.Saved) Then
    If MsgBox(" Do you wish to save this Drawing ?", _ vbaYesNo ) = vbaYes
    Then ThisDrawing.Save
    End if
    End if
End Sub
```

## 5.2. ĐIỀU KHIỂN CỦA SỔ ỨNG DỤNG

### 5.2.1. Trạng thái cửa sổ

Cách sử dụng thuộc tính này có dạng như sau :

Object.WindowState

**Trong đó :**

| | |
|---|---|
| Object | Các đối tượng thuộc Application hoặc Document. |
| WindowState | Các biến điều khiển của cửa sổ gồm: |
| - acMin | *Cửa sổ ở trạng thái nhỏ nhất.* |
| - acMax | *Cửa sổ ở trạng thái lớn nhất.* |
| - acNorm | *Cửa sổ ở trạng thái bình thường.* |

Đoạn mã chương trình dưới đây dùng để thao tác với cửa sổ ứng dụng.

```
Sub Example_WindowState()
  ACAD application.
   'Khai báo biến trạng thái
   Dim CurrentState As String
   'Các lựa chọn cửa sổ ứng dụng
   Select Case WindowState
     Case acMin: CurrentState = "Minimized"
     Case acMax: CurrentState = "Maximized"
     Case acNorm: CurrentState = "Normal Size"
  End Select
     MsgBox "ACAD is now: " & CurrentState
  End Sub
```

### 5.2.2. Thay đổi vị trí và cỡ của cửa sổ ứng dụng

#### *5.2.2.1. Vị trí của cửa sổ ứng dụng*

Để thay đổi vị trí của cửa sổ ứng dụng khi thiết kế chương trình cần sử dụng các thuộc tính như: **WindowLeft**, **WindowTop**, **Width** và **Height**.

Đoạn mã chương trình sau sẽ hiển thị cửa sổ nên góc trái màn hình với chiều rộng là 400 và chiều cao là 400 pixel.

```
Sub PositionApplicationWindow()
    ThisDrawing.Application.WindowTop =0
    ThisDrawing.Application.WindowLeft =0
    ThisDrawing.Application.width = 400
    ThisDrawing.Application.height = 400
End Sub
```

#### *5.2.2.2. Phóng to và thu nhỏ của sổ ACAD*

Cửa sổ ACAD có thể phóng to, thu nhỏ bằng cách sử dụng thuộc tính **WindowState Property**.

*a/ Làm cho cửa sổ ứng dụng lớn nhất*

```
ThisDrawing Application.WindowState = acMax
```

*b/ Làm cho cửa sổ ứng dụng nhỏ nhất*

```
ThisDrawing Application.WindowState = acMin
```

#### *5.2.2.3. Tìm kiếm trạng thái hiện thời của cửa sổ ACAD*

Để tìm trạng thái hiện thời của cửa sổ màn hình bằng cách sử dụng thuộc tính **WindowState**.

Đoạn mã chương trình dưới đây sẽ truy vấn trạng thái của cửa sổ ứng dụng và hiển thị trạng thái lên hộp thoại sử dụng.

```
Sub VBA_CurrenWindowState()
    Dim  CurrenWindowState As Integer
    Dim msg As String
    CurrenWindowState = ThisDrawing.Application.WindowState
    Msg= Choose(CurrWindowState,"nomal",_"minimized","maximized")
    MsgBox "The application window is "+ msg
End Sub
```

#### *5.2.2.4. Tắt cửa sổ ứng dụng*

Để làm cho của sổ ứng dụng không hiển thị lên màn hình sử dụng thuộc tính **Visible**.

Đoạn mã chương trình dưới đây sử dụng thuộc tính Visible để ẩn cửa sổ ứng dụng:

```
ThisDrawing Application.Visible = False.
```

## 5.3. ĐIỀU KHIỂN CỬA SỔ BẢN VẼ

Như cửa sổ ứng dụng của ACAD cũng có thể thu nhỏ, phóng to, thay đổi vị trí, kích cỡ hoặc kiểm tra trạng thái của bất kỳ cửa sổ tài liệu nào. Ngoài ra cũng có thể thay đổi cách hiển thị của bản vẽ trong cửa sổ bằng cách sử dụng các phương thức như: views, viewports, và zooming.

Các tiện ích của ACAD cung cấp nhiều cách để hiển thị và quan sát bản vẽ. Người sử dụng có thể điều khiển sự hiển thị của bản vẽ để di chuyển nhanh đến một vùng khác của bản vẽ. Trong khi kiểm tra mọi tác động thay đổi đó thì vẫn có thể được phóng to, thu nhỏ và di chuyển các đối tượng trong bản vẽ v.v...

### 5.3.1. Thay đổi vị trí và cỡ của cửa sổ tài liệu

Sử dụng đối tượng **Document** để sửa đổi vị trí và cỡ của bất kỳ cửa sổ tài liệu nào.

Đoạn mã chương trình sau sử dụng thuộc tính: Width & Height để thiết lập cửa sổ tài liệu với chiều rộng là 500 và chiều cao là 500.

```
ThisDrawing.Width = 500
ThisDrawing.Height = 500
```

### 5.3.2. Phóng to và thu nhỏ một cửa sổ tài liệu

Cửa sổ tài liệu có thể phóng to thu nhỏ được bằng cách sử dụng thuộc tính **WindowState**.

***Phóng to cửa sổ tài liệu:***

```
ThisDrawing.WindowState = acMax
```

***Thu nhỏ cửa sổ tài liệu:***

```
ThisDrawing.WindowState = acMin
```

### 5.3.3. Tìm kiếm trạng thái hiện thời của một cửa sổ tài liệu

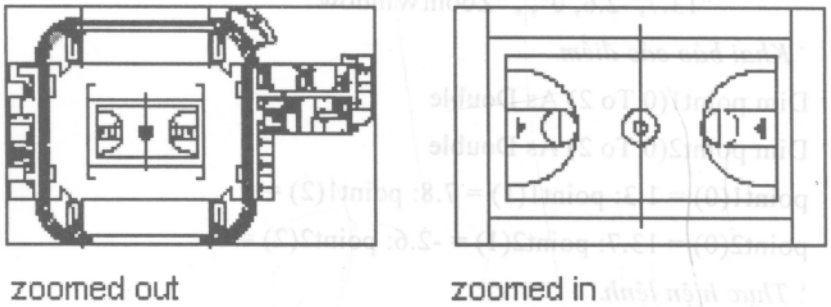
Bạn có thể tìm thấy trạng thái hiện thời của cửa sổ tài liệu bằng cách sử dụng thuộc tính **WindowState**.

Tìm kiếm trạng thái hiện thời của một cửa sổ tài liệu

```
Sub Vidu_CurrenWindowState()
    Dim CurrenWindowState As Integer
    Dim msg As String
    CurrenWindowState = ThisDrawing.WindowState
    Msg= Choose(CurrWindowState,"nomal",_"minimized","maximized")
    MsgBox "The document window is "+ msg
End Sub
```

### 5.3.4. Sử dụng Zoom

Để quan sát rõ ràng bản vẽ, cách phổ biến nhất là sử dụng các lựa chọn của **Zoom**. Với lựa chọn này giúp người sử dụng có thể tăng hoặc giảm kích cỡ hình ảnh hiển thị trong màn hình hiển thị *(không gian vẽ hoặc không gian giấy)* như hình 5.1.



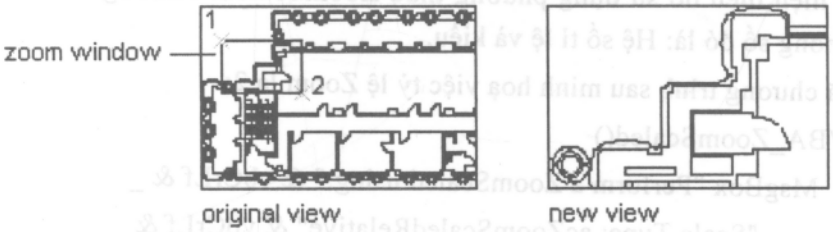

***Hình 5.1.*** Lệnh Zoom.

**Zooming** không thay đổi các cỡ của bản vẽ, mà chỉ thay đổi độ quan sát trong không gian vẽ hoặc không gian giấy.

> ***Chú ý:***
>
> *Phóng to hình ảnh để quan sát các thành phần trong bản vẽ sử dụng **Zooming in.** Thu nhỏ hình ảnh để quan sát hết tổng thể của cả bản vẽ sử dụng **Zooming out***

***Lựa chọn một phần của cửa sổ để Zoom***

Người dùng có thể phóng to, thu nhỏ một vùng của bản vẽ bằng cách chỉ rõ vùng ấy.




***Hình 5.2.*** Zoom phần lựa chọn.

Để phóng to phần lựa chọn, phải sử dụng một trong hai phương thức sau:

Đó là: **ZoomWindow** hoặc **ZoomPickWindow**. Trong đó ZoomWindow cho phép chỉ rõ hai điểm trên cửa sổ. **ZoomPickWindow** cần đến lựa chọn hai điểm. Khi có hai điểm được lựa chọn sẽ trở thành **ZoomWindow**.

Đoạn mã chương trình sau dùng để phóng to, thu nhỏ bản vẽ được chọn bởi hai điểm.

```
Sub ZoomWindow()
        MsgBox "Perform a ZoomWindow with:" & vbCrLf & _
                "1.3, 7.8, 0" & vbCrLf & _
                "13.7, -2.6, 0", , "ZoomWindow"
        ' Khai báo các điểm.
        Dim point1(0 To 2) As Double
        Dim point2(0 To 2) As Double
        point1(0) = 1.3: point1(1) = 7.8: point1(2) = 0
        point2(0) = 13.7: point2(1) = -2.6: point2(2) = 0
        ' Thực hiện lệnh.
        ThisDrawing.Application.ZoomWindow point1, point2
        MsgBox "Perform a ZoomPickWindow", , "ZoomPickWindow"
        ThisDrawing.Application.ZoomPickWindow
End Sub
```

### 5.4. XÁC ĐỊNH TỈ LỆ VIEW (các phần hình ảnh của bản vẽ hiện hành)

Nếu cần tăng hay giảm sự phóng to hình ảnh theo một tỷ lệ chính xác.

**Các cách thực hiện:**

- Quan hệ theo giới hạn bản vẽ.
- Quan hệ đến chế độ quan sát hiện tại.
- Quan hệ đến đơn vị của vùng hiển thị.

Để thực hiện điều đó sử dụng phương thức **ZoomScaled**. Phương thức này bắt phải nhập vào hai thông số đó là: Hệ số tỉ lệ và kiểu.

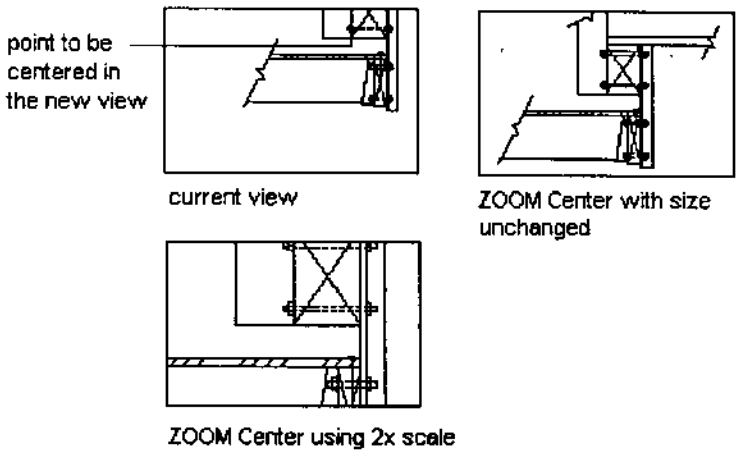
Đoạn mã chương trình sau minh hoạ việc tỷ lệ Zoom là 2:

```
Sub VBA_ZoomScaled()
        MsgBox "Perform a ZoomScaled using:" & vbCrLf & _
            "Scale Type: acZoomScaledRelative" & vbCrLf & _
            "Scale Factor: 2", , "ZoomScaled"
        Dim scalefactor As Double
        Dim scaletype As Integer
        ' Đặt tỷ lệ Zoom
        scalefactor = 2
        scaletype = acZoomScaledRelative
```

```
    'Thi hành trên bản vẽ
    ThisDrawing.Application.ZoomScaled scalefactor, scaletype
End Sub
' Kết thúc.
```

## 5.5. ZOOM VỀ TÂM MÀN HÌNH ĐỒ HOẠ

Để có thể di chuyển một điểm đặc biệt trong bản vẽ về tâm của màn hình. Phương thức **ZoomCenter** được sử dụng hữu ích cho kích thước yêu cầu của một đối tượng và mang đối tượng đó về tâm của màn hình quan sát như hình 5.3.




***Hình 5.3.*** Zoom về tâm của màn hình.

Đoạn mã dưới đây sử dụng phương thức **ZoomCenter** để hiển thị sự quan sát toàn bộ đối tượng và phóng to đối tượng lên hai lần.

```
Sub  ZoomCenter()
    MsgBox "Perform a ZoomCenter using:" & vbCrLf & _
        "Center 3, 3, 0" & vbCrLf & _
        "Magnification: 10", , "ZoomCenter"
    Dim Center(0 To 2) As Double
    Dim magnification As Double
    Center(0) = 3: Center(1) = 3: Center(2) = 0
    'Chọn tỷ lệ phóng to là 10
    magnification = 10
    ThisDrawing.Application.ZoomCenter Center, magnification
End Sub.
```

## 5.6. HIỂN THỊ GIỚI HẠN BẢN VẼ VÀ PHẠM VI CỦA ĐỐI TƯỢNG

Để hiển thị một quan sát cơ bản cả khung bản vẽ, hoặc phạm vi của đối tượng trong bản vẽ. Sử dụng phương thức **ZoomAll**, **ZoomExtent**, hoặc **ZoomPrevious**.

**ZoomAll** hiển thị toàn bộ bản vẽ. Nếu đối tượng được vẽ trong giới hạn, **ZoomAll** sẽ hiển thị cả giới hạn thì phương của nó được thể hiện như sau :

object.ZoomAll ()

**Trong đó:**

object: là các đối tượng đồ hoạ trong bản vẽ.

**ZoomExtent** dùng để phóng to hoặc thu nhỏ bản vẽ khi quan sát, tuy nhiên không phải ở thời điểm quan sát.

Phương của nó được thể hiện như sau:

object.ZoomExtents

Đoạn mã chương trình dưới đây dùng để **Zoom** toàn bộ bản vẽ trên màn hình.

```
Sub VBA_ZoomAll()
    'Hiện thông báo thu nhỏ màn hình.
    MsgBox "Perform a ZoomAll", , "ZoomAll"
    'Thực hiện việc thu nhỏ màn hình.
    ThisDrawing.Application.ZoomAll
    MsgBox "Perform a ZoomExtents", , "ZoomExtents"
    ThisDrawing.Application.ZoomExtents
End Sub
```

## 5.7. TẠO TÊN PHẢN ẢNH

Phần ảnh quan sát sẽ được đặt tên nếu bạn tạo ra nó. Để tạo một phần ảnh mới sử dụng phương thức **Add** để thêm một phần ảnh mới vào tập hợp các phần ảnh quan sát.

Tên của phần ảnh có thể lên tới 255 kí tự, kiểu số, có thể là các kí tự đặc biệt ($), (-) và ( _ ).

Khi ghi lại bản vẽ, thì vị trí, tỉ lệ của phần ảnh quan sát cũng sẽ được ghi lại.

Đoạn mã chương trình dưới đây sẽ đặt tên cho một phần ảnh mới có tên là "View1".

```
Sub VBA_AddView()
    Dim viewObj As AcadView
    Set viewObj = ThisDrawing.Views.Add("View1")
End Sub
```

## 5.8. XÓA PHẦN ẢNH

Để xóa một tên phần ảnh, đơn giản nhất sử dụng phương thức **Delete**.

Cấu trúc cú pháp như sau:

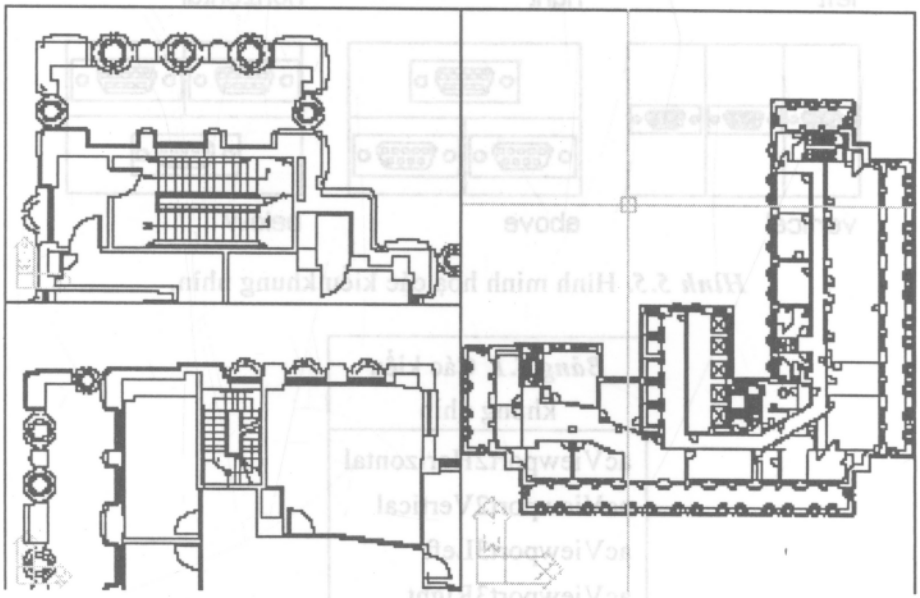
viewObj.Delete

Đoạn mã chương trình dưới đây sẽ xóa một phần ảnh mới có tên là "View1".

```
Sub VBA_DeleteView()
    Dim viewObj As AcadView
    Set viewObj = ThisDrawing.Views("View1")
    'Xoá phần ảnh
    viewObj.Delete
End Sub
```

## 5.9. ĐỐI TƯỢNG KHUNG NHÌN (*VIEWPORT*)

Trong ACAD có hai loại khung nhìn: ***Khung nhìn tĩnh*** và ***khung nhìn động***. Khung nhìn tĩnh chỉ tồn tại trong không gian vẽ, khung nhìn động chỉ tồn tại trong không gian giấy.



***Hình 5.4.*** Đối tượng khung nhìn.

Mỗi khung nhìn có một chỉ số định danh do ACAD quản lý.

Biến hệ thống **TILEMODE**, điều khiển việc sử dụng không gian giấy hay không gian vẽ.

- Nếu TILEMODE = 1 *Khi đó ta sử dụng không gian vẽ.*
- Nếu TILEMODE = 0 *Khi đó ta sử dụng không gian giấy.*

### 5.9.1. Tách thành nhiều khung nhìn

Trong bản vẽ ta có thể tách ra thành nhiều khung nhìn với các kiểu và kích cỡ khác nhau để quan sát các đối tượng trong bản vẽ đó như hình 5.5.

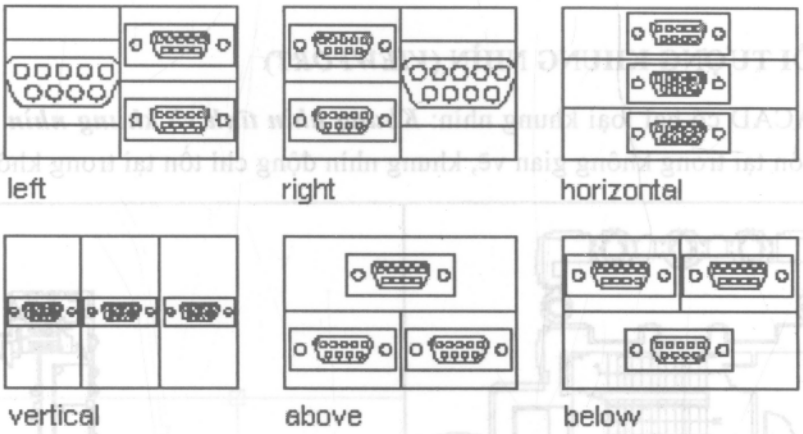
Lệnh **Split** được dùng để tách khung nhìn thành nhiều khung nhìn khác nhau.

**Cú pháp:**

object.Split NumWins

**Trong đó:**

| | |
|---|---|
| object | Đối tượng khung nhìn. |
| NumWins | Các kiểu khung nhìn như bảng 5.1. |




*Hình 5.5.* Hình minh hoạ các kiểu khung nhìn.

| *Bảng 5.1.* Các kiểu khung nhìn |
|---|
| acViewport2Horizontal |
| acViewport2Vertical |
| acViewport3Left |
| acViewport3Right |
| acViewport3Horizontal |
| acViewport3Vertical |
| acViewport3Above |
| acViewport3Below |
| acViewport4 |

Đoạn mã chương trình dưới đây tạo một khung nhìn mới và tách ra thành hai cửa sổ như hình 5.6.

***Hình 5.6.*** Khung nhìn với hai cửa sổ.

```
' Tên file SplitAViewport.dvb.
  Sub SplitAViewport()
    'Khai báo đối tượng khung nhìn.
    Dim vportObj As AcadViewport
    'Tạo đối tượng khung nhìn mới.
    Set vportObj = ThisDrawing.Viewports.Add("TEST_VIEWPORT")
    'Chia cửa sổ thành 2 phần.
    vportObj.Split acViewport2Horizontal
    ThisDrawing.ActiveViewport = vportObj
  End Sub
' Kết thúc.
```

### 5.9.2. Tạo khung nhìn tĩnh hiện thời

Để tạo đối tượng khung nhìn tĩnh hiện thời thì cần phải nhập điểm và lựa chọn đối tượng trong khung nhìn hiện thời. Sử dụng các thuộc tính sau:

+ Thuộc tính: **LowerLeftCorner**:

Hàm có dạng như sau :

Object.LowerLeftCorner

**Trong đó:**

| | |
|---|---|
| Object | Đối tượng khung nhìn. |
| LowerLeftCorner | Biến mảng có hai phần tử và có kiểu dữ liệu Double. |

+ Thuộc tính: **UpperRightCorner**

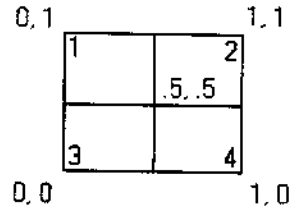
Hàm có dạng như sau :

Object.UpperRightCorner

**Trong đó:**

| | |
|---|---|
| Object | Đối tượng khung nhìn. |
| UpperRightCorner | Biến mảng có hai phần tử và có kiểu dữ liệu Double. |

*Ví dụ:*




***Hình 5.7.***

Viewport 1—LowerLeftCorner = (0, .5), UpperRightCorner = (.5, 1)

Viewport 2—LowerLeftCorner = (.5, .5), UpperRightCorner = (1, 1)

Viewport 3—LowerLeftCorner = (0, 0), UpperRightCorner = (.5, .5)

Viewport 4—LowerLeftCorner = (.5, 0), UpperRightCorner = (1, .5)

Các ví dụ sau tạo các khung nhìn khác nhau:

*Ví dụ 1:* Đoạn mã chương trình sau tạo một khung nhìn mới. Sau đó tách khung nhìn thành bốn cửa sổ và hiển thị toạ độ góc trái của mỗi của sổ.

```
Sub VBA_LowerLeftCorner()
  Dim newViewport As AcadViewport
  Set newViewport = ThisDrawing.Viewports.Add("TESTVIEWPORT")
  ThisDrawing.ActiveViewport = newViewport
  newViewport.Split acViewport4
  ThisDrawing.ActiveViewport = newViewport
  Dim entry As AcadViewport
  Dim lowerLeft As Variant
  For Each entry In ThisDrawing.Viewports
    entry.GridOn = True
    ThisDrawing.ActiveViewport = entry
    lowerLeft = entry.LowerLeftCorner
    MsgBox "The lower left corner of this viewport is " & lowerLeft(0) & ", " &
     lowerLeft(1), , "LowerLeftCorner Example"
    entry.GridOn = False
  Next
End Sub
```

*Ví dụ 2 :* Đoạn mã chương trình sau tạo một khung nhìn mới. Sau đó tách khung nhìn thành 4 cửa sổ và hiển thị toạ độ góc phải của mỗi của sổ.

```
Sub Example_UpperRightCorner()
    Dim newViewport As AcadViewport
    Set newViewport = ThisDrawing.Viewports.Add("TESTVIEWPORT")
    ThisDrawing.ActiveViewport = newViewport
    newViewport.Split acViewport4
    ThisDrawing.ActiveViewport = newViewport
    Dim entry As AcadViewport
    Dim UpperRight As Variant
    For Each entry In ThisDrawing.Viewports
        entry.GridOn = True
        ThisDrawing.ActiveViewport = entry
        UpperRight = entry.UpperRightCorner
        MsgBox "The upper right corner of this viewport is " & UpperRight(0) & ", " &
UpperRight(1), , "UpperRightCorner Example"
        entry.GridOn = False
    Next
End Sub
' Kết thúc.
```

***Hình 5.8***. Kết quả chương trình.

## Chương 6

# VẼ CÁC ĐỐI TƯỢNG CƠ BẢN TRONG MÔI TRƯỜNG ACAD

Các chi tiết đồ hoạ được trình bày trong bản vẽ như đoạn thẳng, đường thẳng, cung tròn, đường tròn, đa giác, đường Spline... được gọi *là các đối tượng trong bản vẽ ACAD.*

Để đưa các đối tượng đồ hoạ vào bản vẽ ta sử dụng phương thức **Add**. Đối tượng trong bản vẽ ACAD được thể hiện trên hai không gian, đó là :

- Không gian vẽ (**MS**).
- Không gian giấy (**PS**).

Các đối tượng trong bản vẽ ACAD có thể được thể hiện trên **PS** hoặc **MS** là môi trường để thực hiện các thiết kế, **MS** chứa đựng tất cả các kiểu chữ, kiểu đường... để tạo nên bản vẽ hoàn thiện. **PS** là môi trường 2D được sử dụng để quan sát các thông tin của không gian vẽ trước khi in bản vẽ. **MS** bao gồm các đối tượng: kiểu đường, kiểu chữ có thể tạo nên bản vẽ và các chế độ quan sát được gọi là khung nhìn.

Khi một đối tượng được thêm vào bản vẽ, thì nó cũng có thể được thêm vào **MS** hoặc trong **PS**.

Để đưa các đối tượng vào **MS** hay **PS** dùng các hàm tham chiếu tương ứng với mỗi không gian ta có một hàm tham chiếu:

+ Hàm tham chiếu đến MS: ***ThisDrawing.ModelSpace***

+ Hàm tham chiếu đến PS: ***ThisDrawing.PaperSpace***

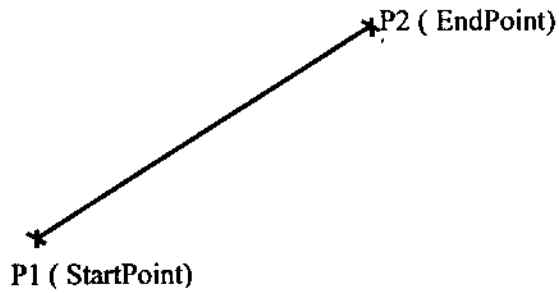
Dưới đây chúng ta tìm hiểu các lệnh vẽ đối tượng một cách tự động trong môi trường ACAD bằng VBA.

### 6.1. VẼ ĐỐI TƯỢNG LÀ ĐOẠN THẲNG

Để vẽ đoạn thẳng đi qua hai điểm P1 và P2 tương ứng với các điểm *StartPoint* và *EndPoint* ta dùng lệnh **AddLine**.

**Cú pháp:**

Object = Space.AddLine
( *StartPoint, EndPoint* )

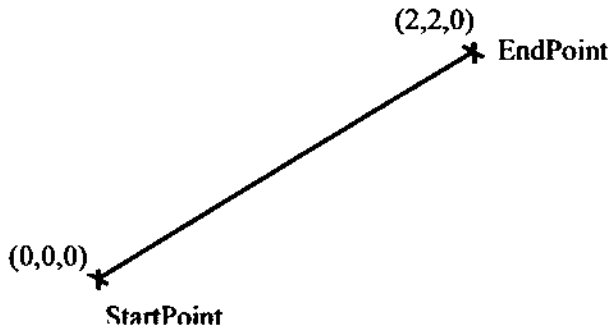



***Hình 6.1.*** Vẽ đoạn thẳng.

**Trong đó:**

| | |
|---|---|
| Object | Đối tượng đoạn thẳng. |
| Space | Không gian vẽ hoặc không gian giấy. |
| *StartPoint* | Là điểm đầu của đoạn thẳng gồm có ba phần tử (x, y, z) có kiểu dữ liệu Double. |
| *EndPoint* | Là điểm cuối của đoạn thẳng gồm ba phần tử (x, y, z) có kiểu dữ liệu Double. |

Đoạn mã chương trình dưới đây là thủ tục vẽ một đoạn thẳng đi qua hai điểm, sau đó áp dụng với thông số đầu vào cho trước là: Điểm đầu P1 (0,0,0) và điểm cuối P2 (2,2,0) trong không gian vẽ như hình 6.2.




***Hình 6.2.*** **Đoạn thẳng đi qua hai điểm $P_1(0,0,0)$ và $P_2(2,2,0)$.**

```
; Tên file VBA_AddLine.dvb
 ' Khai báo biến điểm đầu và điểm cuối.
 Public StartPoint(0 To 2) As Double
 Public EndPoint(0 To 2) As Double
 ' Thủ tục vẽ đoạn thẳng qua hai điểm.
 Public Sub Addline(StartPoint() As Double, EndPoint() As Double)
    Dim Add_line As AcadLine
    Dim startPoint1(2) As Double
    Dim endPoint1(2) As Double
    startPoint1(0) = StartPoint(0)
    startPoint1(1) = StartPoint(1)
    startPoint1(2) = StartPoint(2)
    endPoint1(0) = EndPoint(0)
    endPoint1(1) = EndPoint(1)
    endPoint1(2) = EndPoint(2)
    Set Add_line = ThisDrawing.ModelSpace.Addline(startPoint1, endPoint1)
    Add_line.Update
```

```
End Sub
Sub VBA_Addline()
    ' Định nghĩa các toạ độ điểm đầu và cuối của đoạn thẳng.
    StartPoint(0) = 0: StartPoint(1) = 0: StartPoint(2) = 0
    EndPoint(0) = 2: EndPoint(1) = 2: EndPoint(2) = 0
    ' Tạo đoạn thẳng trong bản vẽ bằng cách gọi thủ tục AddLine vẽ đoạn đường
    thẳng ở trên.
    Call Addline(StartPoint, EndPoint)
    ' Thu toàn bộ bản vẽ về màn hình đồ hoạ..
    ZoomAll
End Sub
; Kết thúc.
```

> ***Chú ý:***
>
> *Dòng lệnh **Add_line.Update** dùng để chắc chắn rằng đoạn thẳng đó xuất hiện trên bản vẽ, đôi khi đối tượng không xuất hiện đến khi **update** lại đoạn thẳng hoặc bản vẽ mới xuất hiện trên màn hình*

## 6.2. VẼ ĐỐI TƯỢNG GỒM NHIỀU ĐOẠN THẲNG LIÊN TIẾP (đường polyline)

Để tạo một đối tượng gồm nhiều đoạn thẳng liên tiếp đi qua các điểm chúng ta sử dụng lệnh **AddLightweightPolyline**.

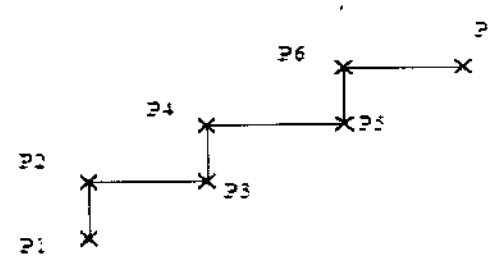
**Cú pháp:**

Object = Space.AddLightweightPolyline (*VerticesList* )

**Trong đó:**

| | |
|---|---|
| Object | Đối tượng gồm nhiều đoạn thẳng liên tiếp. |
| Space | Không vẽ hoặc không gian giấy. |
| *VerticesList* | Đỉnh mà các đoạn thẳng đi qua có kiểu dữ liệu Double. |

Đoạn mã chương trình dưới đây tạo một đối tượng gồm nhiều đoạn thẳng liên tiếp đi qua các điểm, khi muốn kết thúc lệnh bằng cách ấn phím [Enter] như hình 6.3.




***Hình 6.3.*** Tạo một đường Polyline đi qua 7 điểm.

```
; Tên file VBA_AddLightweightPolyline.dvb
 Sub VBA_AddLightweightPolyline()
  ' Khai báo đối tượngcó kiểu  đường LightweightPolyline.
 Dim objLWPline As AcadLWPolyline
 Dim dblPoints() As Double
 Dim intCounter As Integer
 Dim varPoint As Variant
 'Biến đếm ban đầu bằng 0.
 intCounter = 0
 'Tắt kiểm tra lỗi
 On Error Resume Next
 'Toạ độ điểm đầu
 varPoint = ThisDrawing.Utility.GetPoint(, vbCrLf & "Start point: ")
 'Tiếp tục lựa chọn điểm tiếp theo đến khi presses [Enter].
 While Err.Number = 0
 'Tăng kích thước của mảng.
 ReDim Preserve dblPoints(intCounter + 1)
 'Toạ độ X
 dblPoints(intCounter) = varPoint(0)
 'Toạ độ Y
 dblPoints(intCounter + 1) = varPoint(1)
 'Nhận điểm kế tiếp
 varPoint = ThisDrawing.Utility.GetPoint(varPoint, vbCrLf & "Next point: ")
 'Tăng giá trị biến đếm lên 2
 intCounter = intCounter + 2
 Wend
 'Xoá lỗi
 Err.Clear
 'Vẽ đường lightweight polyline trong không gian vẽ.
 Set objLWPline = ThisDrawing.ModelSpace.AddLightweightPolyline(dblPoints)
 ' Khôi phục lại đối tượng để nó xuất hiện trong bản vẽ.
 objLWPline.Update
 End Sub
; Kết thúc.
```

## 6.3. TẠO ĐỐI TƯỢNG LÀ CÁC ĐOẠN LIÊN TIẾP THẲNG SONG SONG (lệnh Mline)

Để vẽ một đối tượng gồm nhiều đoạn thẳng liên tiếp song song và cách đều nhau một khoảng xác định hình 6.4 ta sử dụng lệnh **AddMline**.

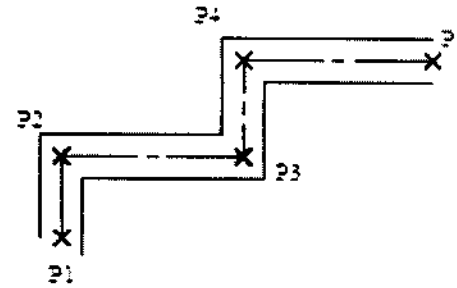
**Cú pháp:**

Object = space.AddMLine(*VertexList*)

**Trong đó:**

| | |
|---|---|
| Object | Đối tượng đường Mline. |
| space | Không gian vẽ hoặc không gian giấy. |
| *VertexList* | Biến mảng có kiểu dữ liệu Double. |

Đoạn mã chương trình dưới đây vẽ đường Mline đến khi bạn nhấn phím [Enter]. Đường Mline được tạo ra trong không gian vẽ với toạ độ các điểm được nhập từ người sử dụng như hình 6.3.




***Hình 6.4.*** Tạo một đối tượng gồm nhiều đoạn thẳng song song.

```
; Tên file VBA_AddMLine.dvb
 Sub VBA_AddMLine()
    'Khai báo đối tượng đường Mline trong VBA.
    Dim objMline As AcadMline
    Dim dblPoints() As Double
    Dim intCounter As Integer
    Dim varPoint As Variant
    'Định nghĩa điểm ban đầu.
    intCounter = 0
    'Dừng chương trình khi có lỗi.
    On Error Resume Next
    'Gán giá trị cho điểm đầu tiên.
    varPoint = ThisDrawing.Utility.GetPoint(, vbCrLf & "Start point: ")
    'Tiếp tục thực hiện khi ấn [ Enter].
```

```
    While Err.Number = 0
    ReDim Preserve dblPoints(intCounter + 2)
    'Thiết lập toạ độ X.
    dblPoints(intCounter) = varPoint(0)
    ' Thiết lập toạ độ Y.
    dblPoints(intCounter + 1) = varPoint(1)
    ' Thiết lập toạ độ Z.
    dblPoints(intCounter + 2) = varPoint(2)
    ' Thiết lập  điểm kế tiếp.
    varPoint = ThisDrawing.Utility.GetPoint(varPoint, vbCrLf & "Next point: ")
    'Tăng số lần đếm.
    intCounter = intCounter + 3
    Wend
    'Xoá lỗi.
    Err.Clear
    'Khởi tạo đường Mline trong không gian vẽ.
    Set objMline =   ThisDrawing.ModelSpace.AddMLine(dblPoints)
    'Khôi phục lại đối tượng để nó xuất hiện trong bản vẽ.
    objMline.Update
  End Sub
; Kết thúc.
```

## 6.4. VẼ ĐỐI TƯỢNG LÀ ĐƯỜNG THẲNG ĐI QUA HAI ĐIỂM

Để vẽ một đối tượng là đường thẳng vô tận đi qua hai điểm ta sử dụng lệnh **AddXLine**.

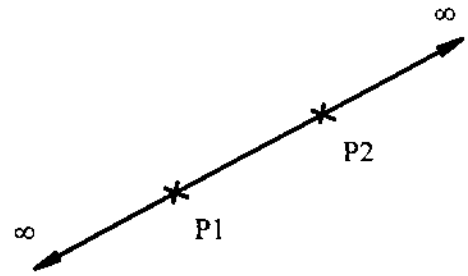
**Cú pháp**:

Object = Space.AddXLine(*LocatePt*, *ThruPt*)

**Trong đó:**

| | |
|---|---|
| Object | Đối tượng đường XLine. |
| Space | Không gian vẽ hoặc không gian giấy. |
| *LocatePt* | Điểm thứ nhất của đường thẳng đi qua gồm ba phần tử (x, y, z) có kiểu dữ liệu Double. |
| *ThruPt* | Điểm thứ hai của đường thẳng đi qua gồm ba phần (x, y, z) có kiểu dữ liệu Double. |

Đoạn mã chương trình dưới đây tạo đường thẳng đi qua hai điểm P1 & P2 trong không gian vẽ, các thông số đầu vào được nhập từ bàn phím, hình 6.5 minh hoạ.




***Hình 6.5.*** Đường X line.

```
; Tên file VBA_AddXLine.dvb
 Sub VBA_AddXLine()
 ' Khai bảo đối tượng đường Xline.
 Dim objXline As AcadXline
 Dim varLocPt As Variant
 Dim varThruPt As Variant
 ' Nhận các điểm từ bàn phím.
 varLocPt = ThisDrawing.Utility.GetPoint(, vbCrLf & "Location point: ")
 varThruPt = ThisDrawing.Utility.GetPoint(varLocPt, vbCrLf & "Through point: ")
 ' Khởi tạo đường Xline trong không gian vẽ.
 Set objXline = ThisDrawing.ModelSpace.AddXLine(varLocPt, varThruPt)
 ' Khôi phục lại đối tượng trong bản vẽ.
 objXline.Update
 End Sub
; Kết thúc.
```

## 6.5. TẠO ĐỐI TƯỢNG LÀ ĐƯỜNG THẲNG BỊ CHẶN MỘT ĐẦU

Để tạo ra một đối tượng là đường thẳng có một phía bị hạn chế và phía còn lại là vô tận ta sử dụng lệnh **Ray** của ACAD (hình 6.6).

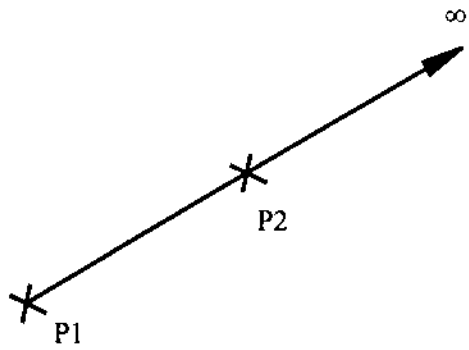
**Cú pháp:**

Object = space.AddRay(*Point1, Point2*)

**Trong đó:**

| | |
|---|---|
| Object | Đối tượng đường là đường Ray. |
| Space | Không gian vẽ hoặc không gian giấy. |
| *Point1* | Điểm thứ nhất của đường thẳng *( điểm hạn chế)*, là biến mảng gồm 3 phần tử có kiểu dữ liệu Double. |
| *Point2* | Điểm thứ hai của đường thẳng đi qua, là biến mảng gồm 3 phần tử có kiểu dữ liệu Double. |

Đoạn mã chương trình dưới đây vẽ một một đường **Ray** trong không gian vẽ có điểm đầu P1 là vị trí bắt đầu, điểm thứ hai P2 là điểm mà đường thẳng đi qua như hình 6.6.




***Hình 6.6.*** Đường thẳng bị hạn chế một đầu.

```
; Tên file VBA_AddRay.dvb
Sub VBA_AddRay()
' Khai bảo kiểu đường Ray trong VBA.
Dim objRay As AcadRay
Dim varStartPt As Variant
Dim varThruPt As Variant
' Nhập điểm từ người sử dụng.
varStartPt = ThisDrawing.Utility.GetPoint(, vbCrLf & "Start point: ")
varThruPt = ThisDrawing.Utility.GetPoint(varStartPt, vbCrLf & "Through point: ")
'Tạo cấu trúc đường  line trong không gian vẽ.
Set objRay = ThisDrawing.ModelSpace.AddRay(varStartPt, varThruPt)
objRay.Update
End Sub
; Kết thúc.
```

## 6.6. TẠO ĐỐI TƯỢNG LÀ MỘT ĐƯỜNG CONG TỰ DO

Để tạo đối tượng là một đường cong tự do (*Spline*) đi qua các điểm khác nhau ta sử dụng lệnh **AddSpline**.

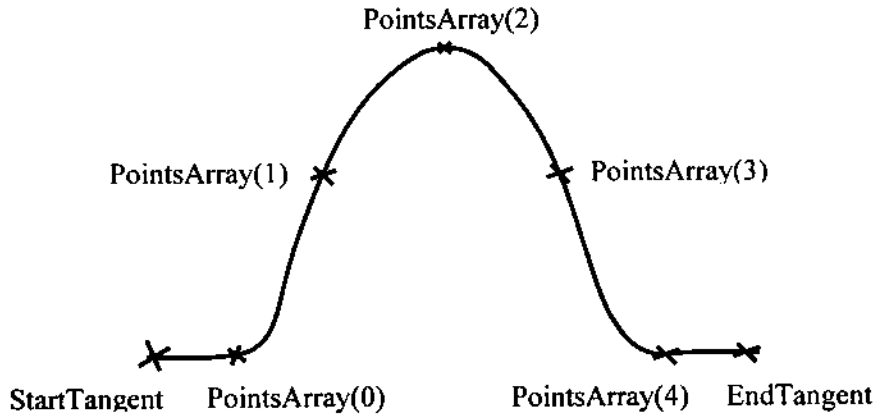
**Cú pháp:**

Object = space.AddSpline(*PointsArray, StartTangent, EndTangent)*

Trong đó:

| | |
|---|---|
| Object | Đối tượng là đường Spline. |
| space | Không gian vẽ hoặc không gian giấy. |
| *StartTangen* | Điểm bắt đầu của đường cong gồm có ba phần tử có kiểu dữ liệu Double là vector tiếp tuyến của đường Spline tại điểm đầu tiên. |
| *PointsArra* | Là các điểm trên Spline có kiểu dữ liệu Double. |
| *EndTangen* | Điểm cuối cùng của đường cong gồm ba phần tử có kiểu dữ liệu Double, là vector tiếp tuyến của đường Spline tại điểm cuối cùng. |

Đoạn mã chương trình dưới đây sẽ vẽ một đường cong tự do đến khi người sử dụng ấn phím [Enter]. Một đường Spline được tạo ra trong không gian vẽ với các thông số được nhập vào như hình 6.7.




***Hình 6.7.*** Đường cong tự do Spline.

```
; Tên file VBA_AddSpline.dvb
 Sub  VBA_AddSpline()
    'Khai báo đối tượng đường Spline trong VBA.
    Dim objSpline As AcadSpline
    Dim dblFitPts() As Double
    Dim intCounter As Integer
    Dim varTanPt1 As Variant
    Dim varTanPt2 As Variant
    Dim varFitPt As Variant
```

```
  'Biến đếm ban đầu được gán bằng 0.
  intCounter = 0
  'Dừng khi có lỗi.
  On Error Resume Next
  'Nhập điểm đầu tiên.
  varFitPt = ThisDrawing.Utility.GetPoint(, vbCrLf & "First point: ")
 'Tiếp tục lựa chọn các điểm cho đến khi ấn phím [Enter].
 While Err.Number = 0
 ReDim Preserve dblFitPts(intCounter + 2)
 'Toạ độ X.
 dblFitPts(intCounter) = varFitPt(0)
 'Toạ độ Y.
 dblFitPts(intCounter + 1) = varFitPt(1)
 'Toạ độ Z.
 dblFitPts(intCounter + 2) = varFitPt(2)
 ' Nhập điểm kế tiếp.
 varFitPt = ThisDrawing.Utility.GetPoint(varFitPt, vbCrLf & "Next point: ")
 intCounter = intCounter + 3
 Wend
 ' Xóa lỗi.
 Err.Clear
 'Nhập điểm start tangent .
 varTanPt1 = ThisDrawing.Utility.GetPoint(, vbCrLf & "Start tangent point: ")
 'Nhập điểm end tangent.
 VarTanPt2 = ThisDrawing.Utility.GetPoint(, vbCrLf & "End tangent point: ")
 ' Tạo đường Spline với các thông số đầu vào.
 Set objSpline = ThisDrawing.ModelSpace.AddSpline(dblFitPts, varTanPt1, _
 varTanPt2)
 ' Khôi phục lại đối tượng trong bản vẽ.
 objSpline.Update
End Sub
; Kết thúc.
```

## 6.7. VẼ ĐỐI TƯỢNG LÀ ĐƯỜNG TRÒN

Để vẽ đối tượng là đường tròn trong bản vẽ ACAD ta sử dụng lệnh **AddCircle**.

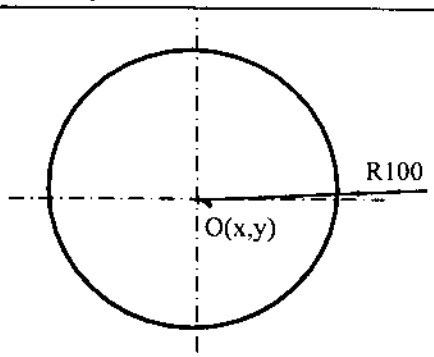
**Cú pháp:**

Object = space.AddCircle(*Center, Radius)*

**Trong đó:**

| | |
|---|---|
| Object | Đối tượng là đường tròn. |
| space | Không gian vẽ hoặc không gian giấy. |
| *Center* | Tâm của đường tròn gồm 3 phần tử (x, y, z) có kiểu dữ liệu Double. |
| *Radius*: | Bán kính của đường tròn có kiểu dữ liệu Double. |

Đoạn mã chương trình dưới đây tạo một thủ tục vẽ đường tròn trong không gian vẽ với tâm O(x, y, z) và bán kính R. như hình 6.8 minh hoạ.




***Hình 6.8.*** Đối tượng là đường tròn.

```
; Tên file VBA_AddCircle.dvb
' Khai báo tâm và bán kính đường tròn có kiểu Public.
Public center(2) As Double
Public  r As Double
' Thủ tục vẽ đường tròn với đầu vào, với thủ tục này ta có thể gọi ở  bất kỳ chỗ nào
trong chương trình .
Public Sub AddCircle(center() As Double, r As Double)
   Dim Cir As AcadCircle
   Dim Center3D(2) As Double
   Center3D(0) = center(0)
   Center3D(1) = center(1)
   Center3D(2) = 0
   Set Cir = ThisDrawing.ModelSpace.AddCircle(Center3D, r)
End Sub
```

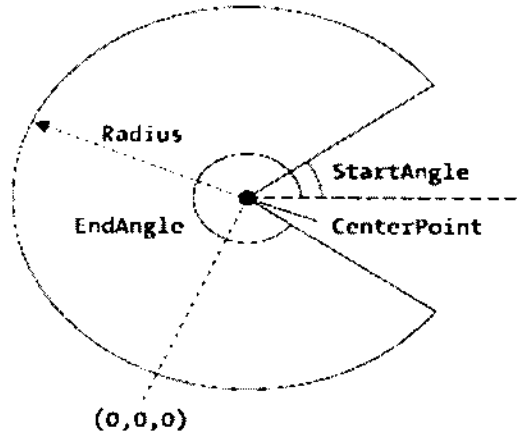
```
Sub VBA_AddCircle()
  ' Định nghĩa đường tròn.
  center(0) = 0: center(1) = 0: center(2) = 0
  r = 100
  ' Gọi thủ tục vẽ đường tròn được khai báo ở trên.
  Call AddCircle(center, r)
  ZoomAll
End Sub
; Kết thúc.
```

## 6.8. TẠO ĐỐI TƯỢNG LÀ MỘT CUNG TRÒN

Để tạo ra một đối tượng là cung tròn trong bản vẽ với các thông số *(tâm, bán kính, điểm đầu, điểm cuối).*

**Cú pháp:**

Object = space.AddArc*(Center, Radius, StartAngle, EndAngle)*

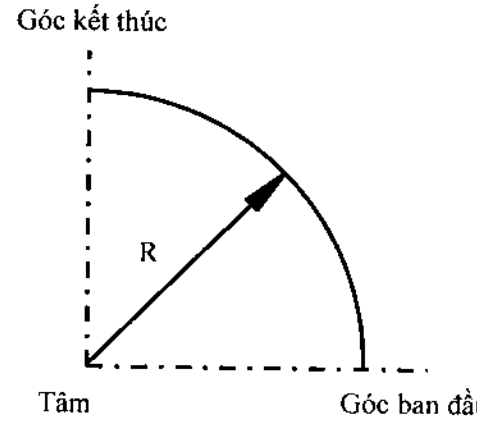



***Hình 6.9.*** Minh hoạ lệnh vẽ cung tròn.

**Trong đó:**

| | |
|---|---|
| Object | Đối tượng cung tròn. |
| Space | Không gian vẽ hoặc không gian giấy. |
| *Center* | Tâm của cung tròn gồm 3 phần tử (x, y, z) có kiểu dữ liệu Double. |
| *Radius* | Bán kính của cung tròn có kiểu dữ liệu là Double. |
| *StartAngle, EndAngle* | Góc ban đầu, và góc kết thúc của cung tròn có kiểu dữ liệu Double (*đơn vị radians*). |

Đoạn mã chương trình dưới đây sẽ vẽ một cung tròn trong không gian vẽ với các thông số: điểm tâm, góc ban đầu, góc cuối, bán kính. Bán kính được nhập vào từ bàn phím, còn các thông số khác có thể được nhập vào từ bàn phím hoặc dùng chuột hình 6.10 minh hoạ.




***Hình 6.10.*** Cung tròn.

```
; Tên file Vidu_AddArc.dvb
  Sub  Vidu_AddArc()
   'Khai báo đối tượng cung tròn.
   Dim objArc As AcadArc
   Dim varCtrPt As Variant
   Dim dblRadius As Double
   Dim dblStartAngle As Double
   Dim dblEndAngle As Double
   ' Nhập toạ độ tâm của cung tròn.
   varCtrPt = ThisDrawing.Utility.GetPoint (,vbCrLf & "Center point: ")
   ' Nhập góc ban đầu của cung tròn.
   dblStartAngle = ThisDrawing.Utility.GetAngle(varCtrPt, vbCrLf & "Start angle: ")
   ' Nhập góc cuối của cung tròn.
    dblEndAngle = ThisDrawing.Utility.GetAngle(varCtrPt, vbCrLf & "End angle: ")
   ' Nhập bán kính của cung tròn.
   dblRadius = ThisDrawing.Utility.GetReal(vbCrLf & "Radius: ")
   ' Tạo cung tròn với các thông số đầu vào .
   Set objArc = ThisDrawing.ModelSpace.AddArc(varCtrPt, dblRadius, _
   dblStartAngle, dblEndAngle)
   'Khôi phục lại đối tượng để nó xuất hiện trong bản vẽ.
   objArc.Update
    End Sub
  ; Kết thúc.
```

## 6.9. TẠO ĐƯỜNG ELLIPSE

Lệnh **AddEllipse** dung để vẽ đường Ellipse trong bản vẽ.

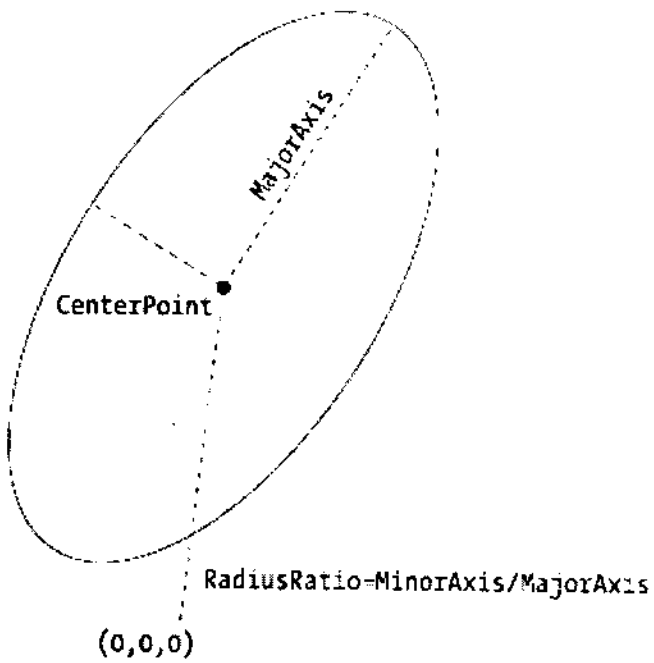
**Cú pháp:**

Object =space.AddEllipse*(Center, MajorAxis, RadiusRatio)*

**Trong đó:**

| | |
|---|---|
| Object | Đối tượng đường Ellipse. |
| space | Không gian vẽ hoặc không gian giấy. |
| *Center* | Tâm của Ellip gồm ba phần tử (x, y, z) có kiểu dữ liệu Double. |
| *MajorAxis* | Chiều dài trục chính của Ellip có kiểu dữ liệu Double. |
| *RadiusRatio* | Tỉ số giữa Minor/Major như hình 6.11 có kiểu dữ liệu Double. |

Đoạn mã chương trình dưới đây sẽ tạo một Ellipse trong không gian vẽ với các thông số đầu vào được nhập từ người sử dụng như hình 6.11.




***Hình 6.11.*** Đường Ellipse.

```
; Tên file VBA_AddEllipse.dvb
 Sub VBA_AddEllipse()
   ' Khai báo đối tượng đường Ellipse.
   Dim objEllipse As AcadEllipse
   Dim varCtrPt As Variant
   Dim varAxisPt As Variant
   Sub Ch05_AddEllipse()
```

```
'Khai báo đối tượng đường Ellipse.
Dim dblRatio As Double
'Nhập toạ độ tâm ellipse.
varCtrPt = ThisDrawing.Utility.GetPoint(, vbCrLf & "Center point: ")
'Nhập chiều dài trục chính.
Dim dblRatio As Double
varAxisPt = ThisDrawing.Utility.GetPoint(varCtrPt, vbCrLf & "Major axis point: ")
'Nhập tỉ số.
varAxisPt = ThisDrawing.Utility.GetPoint(varCtrPt, vbCrLf & "Major axis point: ")
dblRatio = ThisDrawing.Utility.GetReal(vbCrLf & "Ratio: ")
Set objEllipse = ThisDrawing.ModelSpace.AddEllipse(varCtrPt, varAxisPt,_
dblRatio)
'Khôi phục lại đối tượng để nó xuất hiện trong bản vẽ.
objEllipse.Update
End Sub
; Kết thúc.
```

## 6.10. TẠO ĐỐI TƯỢNG LÀ HÌNH CHỮ NHẬT

Để vẽ hình chữ nhật trong VBA có thể sử dụng nhiều cách khác nhau. Sau đây là cách dùng đường **LightWeightPolyline** để vẽ.

Trong đoạn chương trình con này tác giả đã xây dựng một chương trình con sẵn để vẽ hình chữ nhật với toạ độ điểm đầu và điểm cuối của hình chữ nhật.

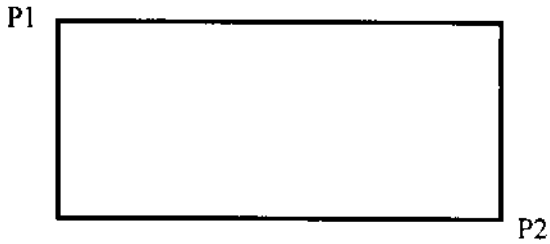
**Cú Pháp:**

Rectang*(FirstPoint() As Double, EndPoint() As Double)*

**Trong đó:**

| | |
|---|---|
| *FirstPoint* | Điểm đầu với ba thông số (x, y, z) có kiểu dữ liệu là Double. |
| *EndPoint* | Điểm cuối với ba thông số (x, y, z) có kiểu dữ liệu là Double. |

Đoạn mã chương trình sau tạo một thủ tục vẽ hình chữ nhật trong không gian vẽ với điểm đầu P1 và điểm cuối P2 như hình 6.11.




***Hình 6.12.*** Hình chữ nhật.

```
; Tên file VBA_Rectang.dvb
 ' Thủ tục vẽ hình chữ nhật với điểm đầu và điểm cuối.
 Public Sub Rectang(FirstPoint() As Double, EndPoint() As Double)
   Dim plineObj As AcadLWPolyline
   Dim points(0 To 7) As Double
   ' Định nghĩa các điểm đầu và điểm cuối.
   points(0) = FirstPoint(0): points(1) = FirstPoint(1)
   points(2) = EndPoint(0): points(3) = FirstPoint(1)
   points(4) = EndPoint(0): points(5) = EndPoint(1)
   points(6) = FirstPoint(0): points(7) = EndPoint(1)
   Set plineObj = ThisDrawing.ModelSpace.AddLightWeightPolyline(points)
   plineObj.Closed = True
   ThisDrawing.Regen (True)
 End Sub
 'Chương trình vẽ hình chữ nhật với điểm đầu, điểm cuối cho trước, sử dụng thủ
 tục Rectang ở trên.
 Sub VBA_Rectang()
  Dim FirstPoint(2) As Double
  Dim EndPoint(2) As Double
  ' Định nghĩa các điểm.
  FirstPoint(0) = 0: FirstPoint(1) = 0: FirstPoint(2) = 0
  EndPoint(0) = 50: EndPoint(1) = 25: EndPoint(2) = 0
  ' Gọi thủ tục vẽ hình chữ nhật.
  Call Rectang(FirstPoint, EndPoint)
  ZoomAll
 End Sub
; Kết thúc.
```

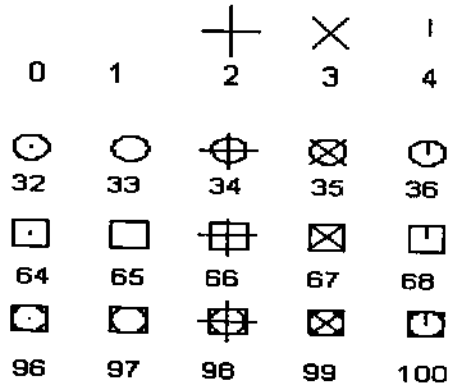
## 6.11. LÀM VIỆC VỚI CÁC ĐỐI TƯỢNG ĐIỂM

### a. Khái quát

Trong các bản vẽ đối tượng điểm có thể được sử dụng như các nút hoặc các điểm tham chiếu ví dụ như lấy hai điểm làm trục lấy đối xứng các đối tượng, có thể thiết lập kiểu và kích thước điểm liên quan đến chế độ màn hình hoặc các hệ đơn vị (inch, mm vv...) mà bạn đang sử dụng.

Trong đó :

PDMODE, PDSIZE: Là 2 biến hệ thống điều khiển sự xuất hiện của đối tượng điểm. Các giá trị của PDMODE dưới đây sẽ tương ứng với các kiểu điểm như sau :




***Hình 6.13.*** Các kiểu điểm.

PDSIZE: Biến điều khiển kích cỡ của điểm.

Để thiết lập PDMODE và PDSIZE, sử dụng phương thức: **SetVariable**

**b. Thao tác với đối tượng điểm**

Để tạo ra đối tượng điểm sử dụng biến điều khiển PDMODE ta sử dụng lệnh **AddPoint.**

**Cú pháp**:

Object = Space.AddPoint(*Point*)

**Trong đó**:

| | |
|---|---|
| Object | Là đối tượng điểm. |
| Space | Không gian vẽ hoặc không gian giấy. |
| *Point* | Biến mảng gồm ba phần tử (x, y, z) có kiểu dữ liệu Double. |

Đoạn mã chương trình dưới đây sẽ tạo ra một điểm trong không gian vẽ tại vị trí do người dùng lựa chọn trên bản vẽ như hình 6.14.

Point

***Hình 6.14.*** Đối tượng điểm.

```
; Tên file VBA_AddPoint.dvb
Sub VBA_AddPoint()
 'Khai báo đối tượng điểm trong VBA.
 Dim objPoint As AcadPoint
```

```
  Dim varLocPt As Variant
  'Nhập vị trí của điểm.
  varLocPt = ThisDrawing.Utility.GetPoint(,  vbCrLf & "Point location: ")
' Khởi tạo điểm.
   Set objPoint = ThisDrawing.ModelSpace.AddPoint(varLocPt)
' Khôi phục lại bản vẽ.
objPoint.Update
  End Sub
; Kết thúc.
```

## 6.12. VẼ ĐỐI TƯỢNG LÀ ĐA GIÁC CÓ TÔ ĐẶC Ở BÊN TRONG

Để vẽ đối tượng là đa giác được tô 2D có tô đặc ở trong.

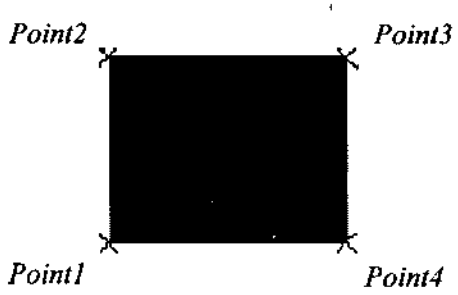
**Cú pháp:**

Object = Space.AddSolid(*Point1, Point2, Point3, Point4*)

**Trong đó:**

| | |
|---|---|
| Object | Đối tượng đa giác được tô 2D. |
| Space | Không gian vẽ hoặc không gian giấy. |
| *Point1* | Điểm đầu tiên của đa giác được tô gồm có ba phần tử (x, y, z) và có kiểu dữ liệu là Double. |
| *Point2* | Điểm thứ hai của đa giác được tô gồm có ba phần tử (x, y, z) và có kiểu dữ liệu là Double. |
| *Point3* | Điểm thứ ba của đa giác được tô gồm có ba phần tử (x, y, z) và có kiểu dữ liệu là Double. |
| *Point4* | Điểm thứ tư của đa giác được tô gồm có ba phần tử (x, y, z) và có kiểu dữ liệu là Double. |

Đoạn mã chương trình dưới đây sẽ tạo một đa giác được tô 2D trong không gian vẽ, với các thông số đầu vào do người sử dụng nhập vào như hình 6.15.




***Hình 6.15.*** Đa giác được tô 2D.

```
; Tên file VBA_AddSolid.dvb.
  Sub VBA_AddSolid()
    ' Khai báo đối tượng điểm trong VBA
    Dim objSolid As AcadSolid
    ' Khai báo các biến đầu vào.
    Dim varPt1 As Variant
    Dim varPt2 As Variant
    Dim varPt3 As Variant
    Dim varPt4 As Variant
    'Nhập vị trí các điểm từ bàn phím.
    varPt1 = ThisDrawing.Utility.GetPoint(, vbCrLf & "First point: ")
    varPt2 = ThisDrawing.Utility.GetPoint(varPt1, vbCrLf & "Second point: ")
    varPt3 = ThisDrawing.Utility.GetPoint(varPt2, vbCrLf & "Third point: ")
    varPt4 = ThisDrawing.Utility.GetPoint(varPt3, vbCrLf & "Fourth point: ")
    'Tạo miền đa giác tô từ các thông số đầu vào.
    Set objSolid = ThisDrawing.ModelSpace.AddSolid(varPt1, varPt2, varPt3,
_varPt4)
    ' Đặt màu tô cho đối tượng.
    objSolid.Color = acRed
    objSolid.Update
  End Sub
; Kết thúc.
```

## 6.13. VIẾT CHỮ TRONG BẢN VẼ

Trong phần này sẽ trình bày về việc tạo các đối tượng chữ trong bản vẽ của ACAD. Có hai kiểu Text trong ACAD đó là: ***Tạo chữ trên một dòng và chữ trên nhiều dòng***.

### 6.13.1. Kiểu chữ trên một dòng

Lệnh **AddText** là cách khởi tạo một dòng chữ.

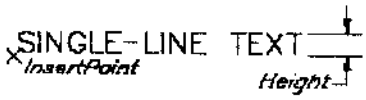
**Cú pháp:**

*Object* = *Space*.AddText(*TextString*, *InsertPoint*, *Height*)

**Trong đó:**

| | |
|---|---|
| *Object* | Đối tượng chữ. |
| *Space* | Không gian vẽ hoặc không gian giấy. |
| *TextStrin* | Là một dạng ký tự có kiểu dữ liệu là String. |
| *InsertPoin* | Vị trí để chèn chữ vào trong bản vẽ gồm có ba phần tử (x, y, z) có kiểu dữ liệu là Double. |
| *Height* | Chiều cao của chữ có kiểu dữ liệu Double. |

Đoạn mã chương trình dưới đây khởi tạo một dòng chữ trong không gian vẽ với các thông số đầu vào gồm có vị trí đặt chữ, chiều cao của chữ và ký tự muốn thể hiện do người dùng sử dụng nhập vào từ bàn phím như hình 6.16.




***Hình 6.16.*** Một dòng chữ.

```
; Tên file VBA_AddText.dvb
  Sub VBA_AddText()
    'Khai báo đối tượng kiểu AcadText.
    Dim objText As AcadText
    Dim varInsPt As Variant
    Dim dblHeight As Double
    Dim strText As String
    ' Nhập vị trí của chữ trong bản vẽ.
    varInsPt = ThisDrawing.Utility.GetPoint(, vbCrLf & "Start point: ")
    ' Nhập chiều cao của chữ.
    dblHeight = ThisDrawing.Utility.GetReal(vbCrLf & "Height: ")
    ' Vào chữ cần thể hiện.
    strText = ThisDrawing.Utility.GetString(True, vbCrLf & "Text: ")
    ' Khởi tạo chữ trong bản vẽ.
    Set objText = ThisDrawing.ModelSpace.AddText(strText, varInsPt, dblHeight)
    objText.Update
  End Sub
; Kết thúc.
```

### 6.13.2. Kiểu chữ trên nhiều dòng

Lệnh **AddMText** là cách để tạo nhiều dòng chữ trong bản vẽ.

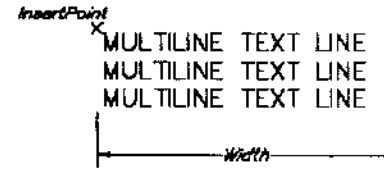
**Cú pháp:**

*Object* = *Space*.AddMText(*InsertPoint*, *Width*, *Text*)

**Trong đó:**

| | |
|---|---|
| *Object* | Đối tượng Mtext. |
| *Space* | Không gian vẽ hoặc không gian giấy. |
| *InsertPoint* | Vị trí để chèn chữ vào trong bản vẽ gồm có ba phần tử (x, y, z) có kiểu dữ liệu là Double. |
| *Width* | Bề rộng của khung mà chữ thể hiện ở bên trong có kiểu dữ liệu là Double. |
| *Text* | Dạng kí tự có kiểu dữ liệu là String. |

Đoạn mã chương trình dưới đây sẽ khởi tạo nhiều dòng chữ trong không gian vẽ với các thông số đầu vào gồm có vị trí đặt chữ, chiều rộng của đường bao chữ và chữ muốn thể hiện do người dùng sử dụng nhập vào hình 6.17.




***Hình 6.17.*** Nhiều dòng chữ.

```
; Tên file VBA_AddMtext.dvb
 Sub VBA_AddMtext()
 'Khai báo đối tượng có kiểu là AcadMText trong VBA.
 Dim objMtext As AcadMText
 Dim varInsPt As Variant
 Dim dblWidth As Double
 Dim strText As String
 ' Nhập vi trí của chữ trong bản vẽ.
 varInsPt = ThisDrawing.Utility.GetPoint(, vbCrLf & "Start point: ")
 ' Nhập chiều rộng của đường bao.
 dblWidth = ThisDrawing.Utility.GetReal(vbCrLf & "Width: ")
 ' Nhập chữ muốn thể hiện từ bàn phím.
 strText = ThisDrawing.Utility.GetString(True, vbCrLf & "Text: ")
 ' Khởi tạo  mtext.
 Set objMtext = ThisDrawing.ModelSpace.AddMText(varInsPt, dblWidth, strText)
 objMtext.Update
  End Sub
; Kết thúc.
```

## Chương 7

# HIỆU CHỈNH CÁC ĐỐI TƯỢNG

Trong quá trình lập trình tính toán, thiết kế cũng như vẽ tự động chi tiết cơ khí có nhiều chi tiết có tính chất đặc biệt, ví dụ như tính đối xứng, đối xứng trục v.v... Để tránh việc lặp lại một công việc nhiều lần cũng như hỗ trợ cho các lệnh vẽ các đối tượng cơ bản ở chương 6 một cách nhanh nhất, hiệu quả nhất, chương này trình bày các lệnh hiệu chỉnh nhanh các đối tượng như copy, copy theo mảng tròn, xoay nhóm đối tượng quanh một điểm, copy theo mảng chữ nhật v.v... Dưới đây trình bày các lệnh hiệu chỉnh.

## 7.1. ĐỔI TÊN ĐỐI TƯỢNG

Để đổi tên các đối tượng trong bản vẽ sử dụng thuộc tính **Name**.

**Cú pháp**:

object.Name

**Trong đó:**

object: Tất cả các đối tượng của bản vẽ.

Đoạn mã chương trình dưới đây tạo một lớp có tên là "NewLayer" sau đó đổi tên lớp đó sang tên mới là "MyLayer".

```
; Tên file VBA_RenamingLayer.dvb.
    Sub VBA_RenamingLayer()
        Dim layerObj As AcadLayer                    'Khai báo một lớp mới.
        Set layerObj =                               'Khởi tạo lớp mới.
        ThisDrawing.Layers.Add("NewLayer")
        layerObj.Name = "MyLayer"                    'Đổi tên lớp mới.
    End Sub
; Kết thúc.
```

## 7.2. COPY ĐỐI TƯỢNG

Trong quá trình thiết kế tự động việc tính toán, vẽ lặp lại với các đối tượng là điều tất yếu. Do vậy mà việc sử dụng lại các đối tượng đã tạo, tạo đối tượng mới làm cho quá trình tính toán, thiết kế trở lên nhanh chóng. Chức năng sao chép lại một đối tượng hay nhiều đối tượng được thực hiện trong không gian vẽ.

Có cú pháp như sau:

**Cú pháp:**

RetVal = Object.Copy

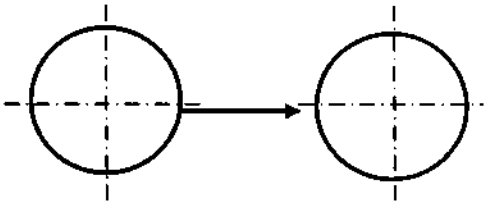
**Trong đó:**

| | |
|---|---|
| Object | Tất cả các đối tượng của bản vẽ bảng 7.1. |
| RetVal | Đối tượng mới được tạo thành sau khi copy. |

***Bảng 7.1.*** Các đối tượng

| | | |
|---|---|---|
| 3DFace | 3DPolyline | 3DSolid |
| Arc | Attribute | AttributeReference |
| BlockRef | Circle | Dim3pointAngular |
| DimAligned | DimAngular | DimDiametric |
| DimOrdinate | DimRadial | DimRotated |
| Ellipse | ExternalReference | Hatch |
| Leader | LightweightPolyline | Line |
| MInsertBlock | Mline | Mtext |
| Point | PolyfaceMesh | PolygonMesh |
| Polyline | PViewport | Raster |
| Ray | Region | Shape |
| Solid | Spline | Text |
| Tolerance | Trace | Xline |

Đoạn mã chương trình sau thực hiện vẽ một đường tròn trong không gian vẽ, sau đó có thông báo Copy đối tượng vừa vẽ. Khi đã lựa chọn Copy, chương trình thông báo di chuyển đối tượng được Copy đến một vị trí mới theo phương X như hình 7.1.



***Hình 7.1.*** Copy một đường tròn.

```
; Tên file VBA_Copy.dvb
 Sub VBA_Copy()
   'Khai báo đối tượng đường tròn có kiểu AcadCircle trong VBA.
   Dim circleObj As AcadCircle
   'Khai báo tâm đường tròn có kiểu dữ liệu Double.
  Dim center (0 To 2) As Double
```

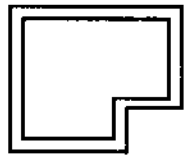
```
'Khai báo bán kính đường tròn có kiểu dữ liệu Double.
Dim radius As Double
' Định nghĩa các thông số đầu vào.
center(0) = 2.0: center(1) = 2.0: center(2) = 0.0
radius = 0.5
'Tạo đường tròn trong không gian vẽ.
Set circleObj = ThisDrawing.ModelSpace.AddCircle(center, radius)
'Hiện thông báo lựa chọn việc Copy đối tượng.
MsgBox "Copy the circle.", , "Copy Example"
'Khai báo một đối tượng đường tròn mới copyCircleObj
Dim copyCircleObj As AcadCircle
Set copyCircleObj = circleObj.Copy()
' Di chuyển đối tượng theo phương X một khoảng hai đơn vị .
Dim point1(0 To 2) As Double
Dim point2(0 To 2) As Double
point1(0) = 0: point1(1) = 0: point1(2) = 0
point2(0) = 2: point2(1) = 0: point2(2) = 0
'Hiện hộp thoại thông báo về việc di chuyển.
MsgBox "Move the copied circle 2 units in the X direction.", , "Copy Example"
'Di chuyển đường tròn đến vị trí mới.
copyCircleObj.Move point1, point2
MsgBox "Move completed.", , "Copy Example"   ' Thông báo việc Copy hoàn
thành
End Sub
; Kết thúc.
```

### 7.3. OFFSET CÁC ĐỐI TƯỢNG

Để Copy đối tượng sao cho đối tượng mới luôn cách đối tượng gốc một khoảng xác định cho trước ta dùng lệnh **Offset**. Đối tượng có thể là một hoặc có thể là một tập hợp đối tượng như đường cong hay các biên dạng phức tạp v.v...

**Cú pháp**:

OffsetObj = Object.Offset*(Distance)*

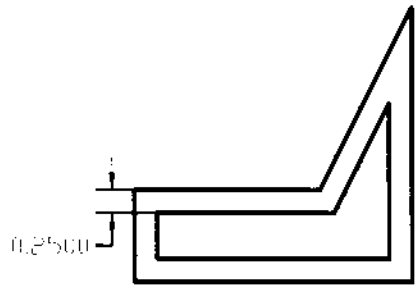


***Hình 7.2.*** Minh hoạ lệnh Offset.

**Trong đó:**

| | |
|---|---|
| Object | Các đối tượng Offset như (*Arc, Circle, Ellipse, Line, Polyline, Lightweight Polyline, Spline, XLine*...). |
| *Distance* | Khoảng cách Offset có kiểu dữ liệu là Double. |
| OffsetObj | Đối tượng mới được tạo ra sau khi đã Offset có kiểu Variant. |

Đoạn mã chương trình dưới đây vẽ một đường Polyline và Offset đường đó một khoảng cách là 0,25 mm như hình 7.3.

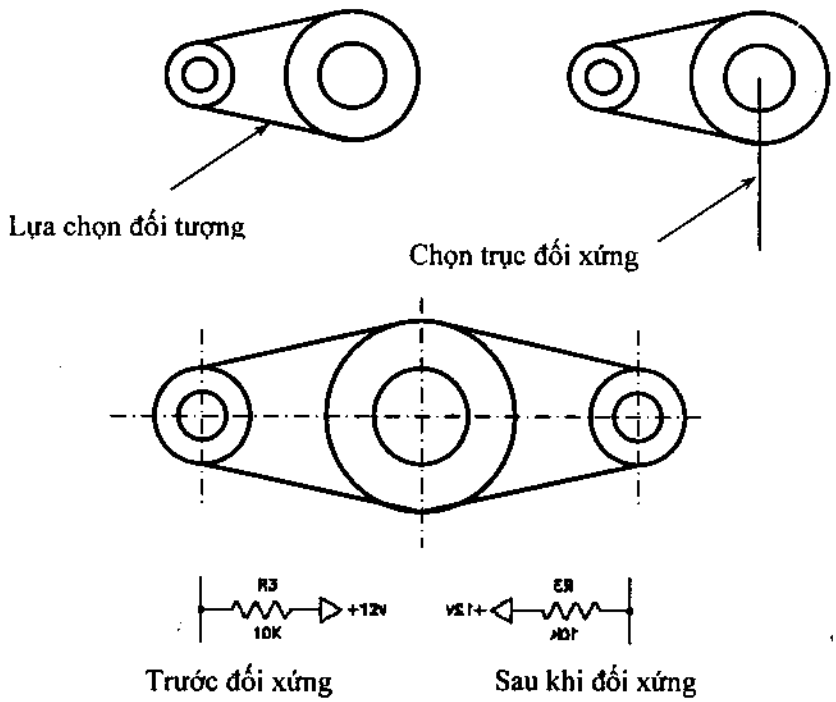



***Hình 7.3.*** Offset đường Polyline.

```
; Tên file VBA_Offset.dvb.
Sub VBA_Offset()
    ' Khai báo đối tượng đường Polyline.
    Dim plineObj As AcadLWPolyline
    'Khai báo các điểm.
    Dim points(0 To 11) As Double
    'Định nghĩa các điểm.
    points(0) = 1: points(1) = 1
    points(2) = 1: points(3) = 2
    points(4) = 2: points(5) = 2
    points(6) = 3: points(7) = 2
    points(8) = 4: points(9) = 4
    points(10) = 4: points(11) = 1
    'Khởi tạo đường Polyline trong không gian vẽ
    Set plineObj =  ThisDrawing.ModelSpace.AddLightWeightPolyline(points)
```

```
    plineObj.Closed = True
    MsgBox "Offset the polyline by 0.25.", , "Offset Example"
    Dim offsetObj As Variant
    'Khoảng cách cần offset
    offsetObj = plineObj.offset(0.25)
    ' Thông báo đã hoàn thành
    MsgBox "Offset completed.", , "Offset Example"
End Sub
; Kết thúc.
```

## 7.4. LẤY ĐỐI XỨNG CÁC ĐỐI TƯỢNG QUA MỘT TRỤC ĐỐI XỨNG CHO TRƯỚC




*Hình 7.4.* Đối xứng đối tượng.

Để lấy đối xứng một hay một tập hợp các đối tượng trong bản vẽ qua một trục đối xứng cho trước, ta sử dụng lệnh **Mirror**. Lệnh này cho phép vẽ nhanh các chi tiết có tính chất đối xứng.

**Cú pháp:**

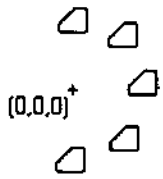
RetVal = Object.Mirror*(Point1, Point2)*

**Trong đó:**

| | |
|---|---|
| Object | Các đối tượng lấy đối xứng như (*Arc, Circle, Ellipse, Line, Polyline, LightweightPolyline, Spline, XLine...*). |
| *Point1, Point2* | Là hai điểm lấy đối xứng gồm ba phần tử (x, y, z) có kiểu dữ liệu Double. |

## 7.5. SAO CHÉP CÁC ĐỐI TƯỢNG

Để **Copy** đối tượng gốc thành các đối tượng khác nhau theo một mảng tròn, hay theo dạng ma trận ta dùng lệnh **Array**, hình 7.5 dưới đây là một ví dụ minh hoạ copy theo mảng tròn




*Số phần tử =5, góc =180°, tâm =0, 0, 0*

***Hình 7.5.*** Sao chép đối tượng theo mảng tròn.

### 7.5.1. Sao chép các đối tượng theo mảng tròn

**Cú pháp:**

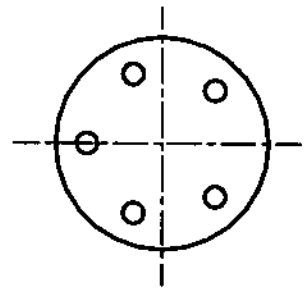
RetVal = Object.ArrayPolar *(NumberOfObjects, AngleToFill, CenterPoint)*

**Trong đó:**

| | |
|---|---|
| Object | Tất cả các đối tượng của bản vẽ. |
| *NumberOfObjects* | Số đối tượng cần sao chép có kiểu dữ liệu Double. |
| *AngleToFill* | Góc giới hạn phạm vi sao chép, có kiểu dữ liệu Double *(đơn vị Radians)*. |
| *CenterPoint* | Tâm sao chép gồm ba phần tử (x, y, z) có kiểu dữ liệu Double. |
| RetVal | Biến mảng dùng để chứa các đối tượng mới được tạo ra sau sao chép. |

Đoạn mã chương trình dưới đây vẽ một đường tròn trong không gian vẽ và sau đó sao chép nó thành bốn đường tròn khác nhau với giới hạn $360^0$ (hình 7.6).

```
; Tên file VBA_ArrayPolar.dvb
 Sub VBA_ArrayPolar()
   'Khai báo đối tượng đường tròn.
   Dim circleObj As AcadCircle
    'Khai báo tâm đường tròn có kiểu dữ liệu Double.
   Dim center(0 To 2) As Double
   'Khai báo bán kính  đường tròn có kiểu dữ liệu Double.
   Dim radius As Double
   ' Định nghĩa đường tròn.
   center(0) = 2#: center(1) = 2#: center(2) = 0#      ' Tâm đường tròn
   radius = 1#                                         ' Bán kính
   Set circleObj = ThisDrawing.ModelSpace.AddCircle(center, radius)
   center(0) = 0#: center(1) = 0#: center(2) = 0#
   radius = 10#
    Set circleObj = ThisDrawing.ModelSpace.AddCircle(center, radius)
   ZoomAll
   'Hiện thông báo sao chép đối tượng
   MsgBox "Perform the polar array on the circle.", , "ArrayPolar Example"
   ' Định nghĩa đối tượng sao chép
   Dim noOfObjects As Integer
   Dim angleToFill As Double
   'Điểm tâm dùng để sao chép đối tượng
   Dim basePnt(0 To 2) As Double
   noOfObjects = 5
   angleToFill = 1.5*3.14
   basePnt(0) = 0#: basePnt(1) = 0#: basePnt(2) = 0#
   'Khai báo đối tượng sao chép
   Dim retObj As Variant
   retObj = circleObj.ArrayPolar(noOfObjects, angleToFill, basePnt)
   ZoomAll
   MsgBox "Polar array completed.", , "ArrayPolar Example"
 End Sub
; Kết thúc.
```



*Hình 7.6.*

> ***Chú ý:***
>
> ***NumberOfObjects**: Số đối tượng phải là số nguyên dương và lớn hơn 1.*

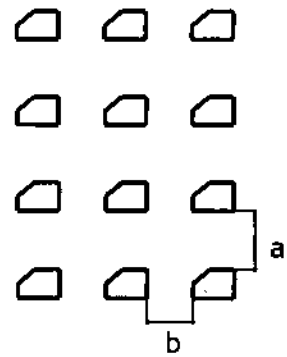
**7.5.2. Sao chép các đối tượng theo mảng hình chữ nhật**

**Cú pháp:**

RetVal = Object.ArrayRectangular (*NumberOfRows, NumberOfColumns, NumberOfLevels, DistBetweenRows, DistBetweenColumns, DistBetweenLevels*)

**Trong đó:**

| | |
|---|---|
| Object | Tất cả các đối tượng trong bản vẽ. |
| RetVal | Mảng chứa các đối tượng mới được tạo ra. |
| *NumberOfRows* | Số hàng cần sao chép có kiểu dữ liệu integer. |
| *NumberOfColumns* | Số cột cần sao chép có kiểu dữ liệu integer. |
| *NumberOfLevels* | Số phần tử cần sao chép có kiểu dữ liệu integer. |
| *DistBetweenRows* | Khoảng cách giữa các cột có kiểu dữ liệu Double. |
| *DistBetweenColumns* | Khoảng cách giữa các hang có kiểu dữ liệu Double. |
| *DistBetweenLevels* | Khoảng cách giữa các phần tử trong mảng có kiểu dữ liệu Double. |

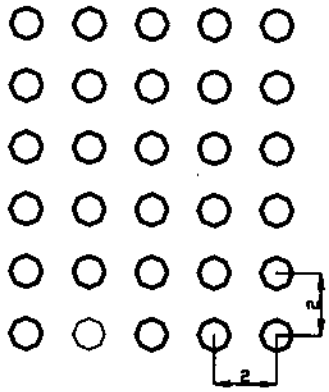



***Hình 7.7.*** Sao chép mảng theo hình chữ nhật.

*(Số hàng = 4, số cột = 3, khoảng cách giữa các hàng = a, khoảng cách giữa các cột = b)*

Đoạn mã chương trình sau đây sẽ vẽ một hình tròn trong không gian vẽ, sau đó sao chép theo mảng hình chữ nhật như hình 7.8.

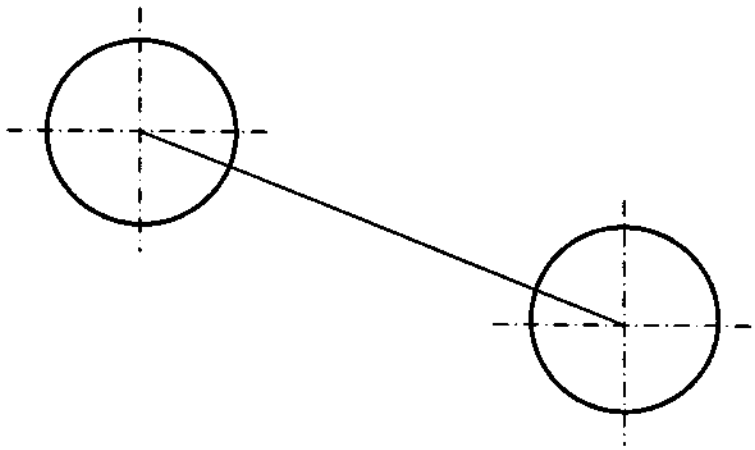
```
; Tên file VBA_ArrayRectangular.dvb.
  Sub VBA_ArrayRectangular()
    ' Tạo hình tròn.
    Dim circleObj As AcadCircle
    Dim center(0 To 2) As Double
    Dim radius As Double
    center(0) = 2#: center(1) = 2#: center(2) = 0#
    radius = 0.5
    Set circleObj = ThisDrawing.ModelSpace.AddCircle(center, radius)
    MsgBox "Perform the rectangular array on the circle.", , "ArrayRectangular
     Example"
    ' Định nghĩa mảng hình chữ nhật.
    Dim numberOfRows As Long
    Dim numberOfColumns As Long
    Dim numberOfLevels As Long
    Dim distanceBwtnRows As Double
    Dim distanceBwtnColumns As Double
    Dim distanceBwtnLevels As Double
    numberOfRows = 6
    numberOfColumns = 5
    numberOfLevels = 2
    distanceBwtnRows = 2
    distanceBwtnColumns = 2
    distanceBwtnLevels = 2
    ' Tạo đối tượng mảng.
    Dim retObj As Variant
    retObj = circleObj.ArrayRectangular(numberOfRows, numberOfColumns,
    numberOfLevels, distanceBwtnRows, distanceBwtnColumns, distanceBwtnLevels)
    MsgBox "Rectangular array completed.", , "ArrayRectangular Example"
  End Sub
; Kết thúc.
```




*Hình 7.8.*

## 7.6. DI CHUYỂN CÁC ĐỐI TƯỢNG

Để di chuyển một hay một nhóm các đối tượng từ vị trí này đến một vị trí mới trong bản vẽ, ta sử dụng lệnh **Move**.



***Hình 7.9.*** Di chuyển đối tượng.

**Cú pháp:**

Object.Move *Point1, Point2*

**Trong đó:**

| | |
|---|---|
| Object | Các đối tượng và các thuộc tính của các đối tượng trong bản vẽ. |
| *Point1* | Vị trí điểm đầu tiên gồm ba phần tử (x, y, z) có kiểu dữ liệu là Double. |
| *Point2* | Vị trí điểm mới di chuyển đến gồm ba phần tử (x, y, z) có kiểu dữ liệu là Double. |

Đoạn mã chương trình dưới đây vẽ một đường tròn sau đó di chuyển nó đến vị trí mới (hình 7.9).

```
; Tên file  VBA_Move.dvb
 Sub VBA_Move()
 Dim circleObj As AcadCircle
 Dim center(0 To 2) As Double
 Dim radius As Double
 ' Định nghĩa đường tròn.
 center(0) = 2#: center(1) = 2#: center(2) = 0#
 radius = 0.5
 Set circleObj = ThisDrawing.ModelSpace.AddCircle(center, radius)
 ZoomAll
 Dim point1(0 To 2) As Double
 Dim point2(0 To 2) As Double
 point1(0) = 0: point1(1) = 0: point1(2) = 0
 point2(0) = 2: point2(1) = 0: point2(2) = 0
 MsgBox "Move the circle 2 units in the X direction.", , "Move Example"
 circleObj.Move point1, point2
 ZoomAll
 MsgBox "Move completed.", , "Move Example
 End Sub
; Kết thúc.
```

## 7.7. QUAY CÁC ĐỐI TƯỢNG

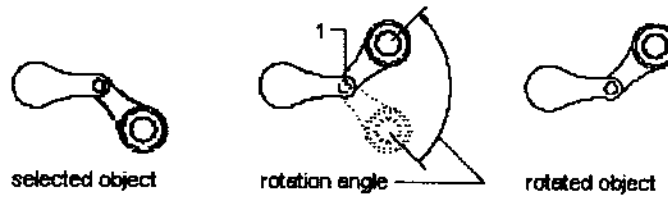
Để xoay một đối tượng hay một nhóm các đối tượng quanh một điểm trong bản vẽ với một góc xoay xác định ta dùng lệnh **Rotate**.

**Cú pháp**:

Object.Rotate (*BasePoint, RotationAngle)*

**Trong đó:**

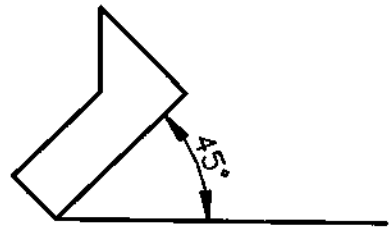
| | |
|---|---|
| *BasePoint* | Là điểm cơ sở được chọn ban đầu để làm tâm quay gồm ba phần tử (x, y, z) có kiểu dữ liệu Double. |
| *RotationAngle* | Trị số góc quay có kiểu dữ liệu Double, đơn vị Radians. |




***Hình 7.10.*** Quay các đối tượng.

Đoạn mã chương trình dưới đây tạo một đường Polyline và sau đó quay đường đó đi một góc là 45 ° như hình 7.11.

```
; Tên file VBA_Rotate.dvb
 Sub VBA_Rotate()
 Dim plineObj As AcadLWPolyline
 Dim points(0 To 11) As Double
 ' Định nghĩa đường Polyline.
 points(0) = 1: points(1) = 2
 points(2) = 1: points(3) = 3
 points(4) = 2: points(5) = 3
 points(6) = 3: points(7) = 3
 points(8) = 4: points(9) = 4
 points(10) = 4: points(11) = 2
 Set plineObj =   ThisDrawing.ModelSpace.AddLightWeightPolyline(points)
 plineObj.Closed = True
 ZoomAll
 MsgBox "Rotate the polyline by 45 degrees.", , "Rotate Example"
 ' Định nghĩa góc xoay
 Dim basePoint(0 To 2) As Double
 Dim rotationAngle As Double
 basePoint(0) = 4: basePoint(1) = 4.25: basePoint(2) = 0
 rotationAngle = 0.7853981   ' 45 degrees
 ' Quay đường Polyline
 plineObj.Rotate basePoint, rotationAngle
 ZoomAll
 MsgBox "Rotation completed.", , "Rotate Example"
 End Sub
; Kết thúc.
```




***Hình 7.11.*** Sau khi xoay đi $45^0$.

## 7.8. XÓA CÁC ĐỐI TƯỢNG

Để xóa một đối tượng hay một nhóm các đối tượng ta sử dụng lệnh **Delete**.

**Cú pháp:**

object.Delete

**Trong đó:**

object : Là tất cả các đối tượng của ACAD

Đoạn mã dưới đây sẽ xóa tất cả các đối tượng trong một Groups (*nhóm*).

```
for each obj in ThisDrawing.Groups
   obj.Delete
next obj
```

Xóa một phần tử trong một nhóm (*Groups*) sử dụng dòng lệnh

ThisDrawing.Groups.item("group1").Delete

## 7.9. PHÓNG TO, THU NHỎ ĐỐI TƯỢNG THEO MỘT TỶ LỆ

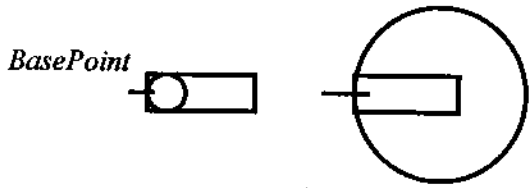
Để phóng to thu nhỏ đối tượng theo một tỷ lệ xác định ta sử dụng lệnh **ScaleEntity**.

**Cú pháp:**

Object.ScaleEntity *(BasePoint, ScaleFactor)*

**Trong đó:**

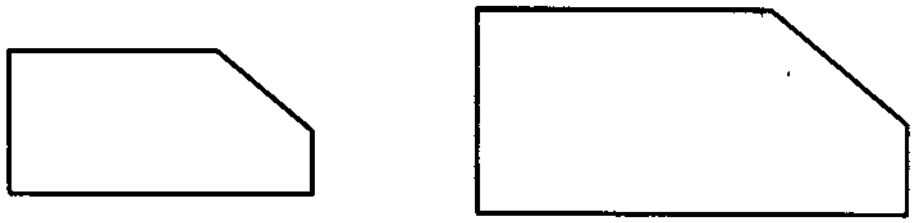
| | |
|---|---|
| Object | Tất cả các đối tượng của bản vẽ. |
| *BasePoint* | Điểm cơ sở ban đầu gồm ba phần tử (x, y, z) có kiểu dữ liệu là Double. |
| *ScaleFactor* | Giá trị tỉ lệ cần nhập vào có kiểu dữ liệu là Double. |




***Hình 7.12.*** Phóng to đối tượng.

Đoạn mã chương trình dưới đây sẽ tạo ra một đường Polyline sau đó phóng to gấp 2 lần như hình 7.13 và 7.14.

```
Sub VBA_ScalePolyline()
  ' Khai báo đối tượng đường Polyline.
  Dim plineObj As AcadLWPolyline
  Dim points(0 To 9) As Double
  ' Định nghĩa đường Polyline.
  points(0) = 5: points(1) = 5
  points(2) = 100: points(3) = 5
  points(4) = 100: points(5) = 25
  points(6) = 70: points(7) = 50
  points(8) = 5: points(9) = 50
  ' Tạo đường Polyline trong không gian vẽ.
  Set plineObj = ThisDrawing.ModelSpace.AddLightWeightPolyline(points)
  plineObj.Closed = True
  ZoomAll
  Dim basePoint(0 To 2) As Double
  Dim scalefactor As Double
  basePoint(0) = 5: basePoint(1) = 5: basePoint(2) = 0
  ' Gán hệ số lấy tỉ lệ.
  scalefactor = 2
  plineObj.ScaleEntity basePoint, scalefactor
  plineObj.Update
End Sub
; Kết thúc.
```

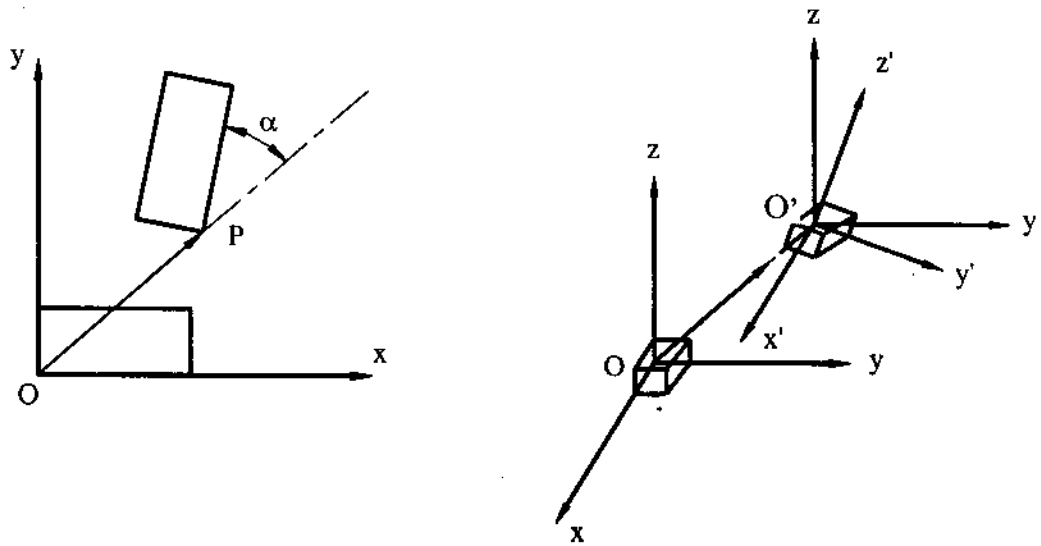


a) Trước khi lấy tỷ lệ  b) Sau khi lấy tỷ lệ

***Hình 7.13.***

## 7.10. BIẾN ĐỔI CÁC ĐỐI TƯỢNG

Để di chuyển tới một vị trí mới và xoay đối tượng theo một góc xác định ta dùng lệnh **TransformBy**. Lệnh này cho phép bạn xoay đối tượng qua một ma trận chuyển đổi gồm các cosin chỉ phương và các vectơ vị trí.




***Hình 7.14.*** Minh hoạ biến đổi vị trí và hướng.

**Cú pháp:**

Object.TransformBy *TransformationMatrix*

**Trong đó :**

| | |
|---|---|
| Object | Tất cả các đối tượng trong bản vẽ. |
| *TransformationMatrix* | Biến mảng có dạng ma trận 4 × 4 có kiểu dữ liệu Double. |

**Các phép chuyển đổi cơ bản như sau:**

***a. Phép quay quanh trục x***

Hệ toạ độ mới quay quanh trục x một góc α nào đó:

$$\underline{R}(x,\alpha) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha & 0 \\ 0 & \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (7.1)$$

***b. Phép quay quanh trục y***

Hệ toạ độ mới quay quanh trục y một góc φ nào đó:

$$\underline{R}(y,\varphi)=\begin{bmatrix}\cos\varphi & 0 & \sin\varphi & 0\\ 0 & 1 & 0 & 0\\ -\sin\varphi & 0 & \cos\varphi & 0\\ 0 & 0 & 0 & 1\end{bmatrix} \tag{7.2}$$

***c. Phép quay quanh trục z***

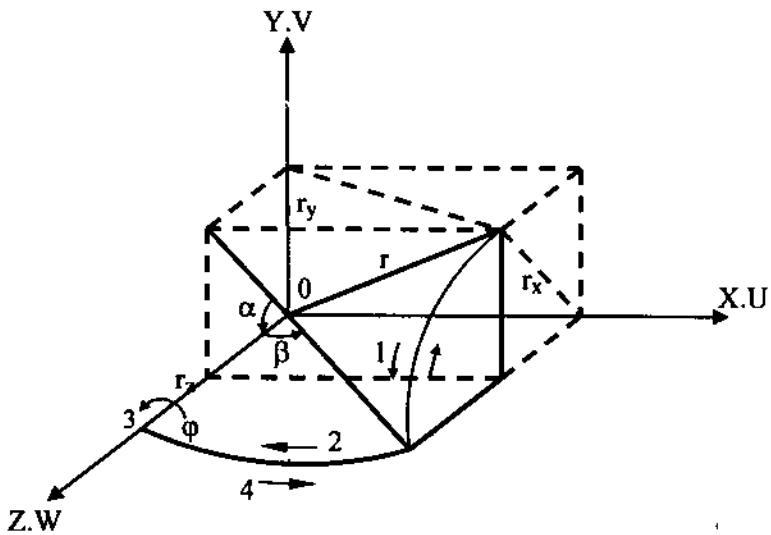
Hệ toạ độ mới quay quanh trục z một góc $\phi$ nào đó:

$$\underline{R}(z,\phi)=\begin{bmatrix}\cos\phi & -\sin\phi & 0 & 0\\ \sin\phi & \cos\phi & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{bmatrix} \tag{7.3}$$

***d. Phép quay quanh trục bất kỳ***

Trong nhiều trường hợp toạ độ động o'x'y'z' gắn liền với vật thể nào đó có thể quay góc φ quanh một trục bất kỳ nào đó cắt qua gốc O, đặc trưng bằng vectơ đơn vị chỉ phương:

$$r = (r_x, r_y, r_z)^T$$




***Hình 7.15.*** Phép quay quanh trục bất kỳ.

Ma trận R(r, φ ) biểu thị sự quay của vật thể quanh trục quay r như sau:

$$T=\begin{bmatrix}r_x^2V_\varphi+C_\varphi & r_xr_yV_\varphi-r_zS_\varphi & r_x\,r_zV_\varphi+r_yS_\varphi & 0\\ r_xr_yV_\varphi+r_zS_\varphi & r_y^2V_\varphi+C_\varphi & r_yr_zV_\varphi-r_xS_\varphi & 0\\ r_x\,r_zV_\varphi-r_yS_\varphi & r_yr_zV_\varphi+r_xS_\varphi & r_z^2V_\varphi+C_\varphi & 0\\ 0 & 0 & 0 & 1\end{bmatrix} \tag{7.4}$$

*e. Phép quay theo ba góc Euler*

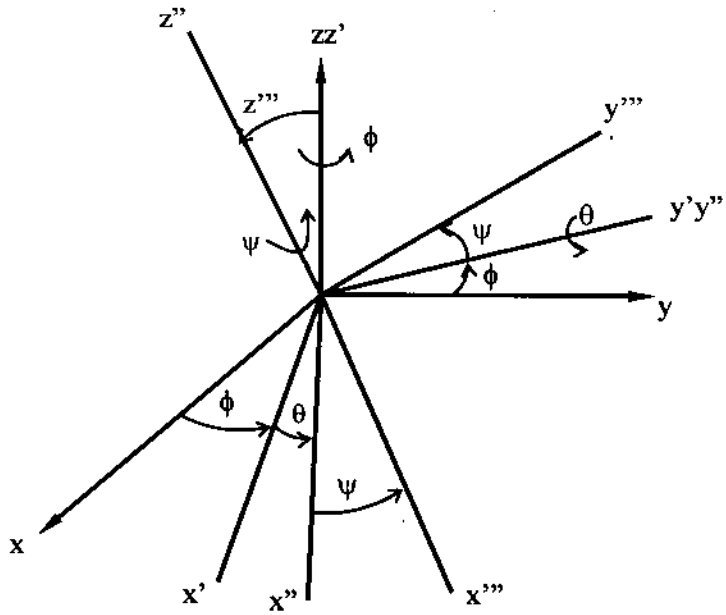
Biểu thị sự quay của vật thể thông qua các góc Euler. Có nhiều cách định hướng bằng 3 góc Euler. Dưới đây là cách thường dùng nhất.

1.Quay góc $\phi$ quanh trục z.

2.Quay tiếp góc $\theta$ quanh trục y, đó là y'.

3. Quay tiếp góc $\psi$ quanh trục z" và ma trận biểu diễn phép quay theo ba góc Euler, gọi tắt là phép quay Euler nhận được bằng cách nhân ba ma trận quay với nhau. Trong phép quay Euler nếu thực hiện phép quay ngược lại ($\psi \rightarrow \theta \rightarrow \phi$) cũng cho kết quả như nhau:

$$R(\phi,\theta,\psi) = R(z, \phi).R(y,\theta).R(z,\psi)$$




***Hình 7.16.*** Phép quay Euler.

**Các phép quay cụ thể như sau:**

| | |
|---|---|
| Quay quanh trục x một góc $\phi$ | $R(x,\phi) = \begin{bmatrix} C_\phi & -S_\phi & 0 & 0 \\ S_\phi & C_\phi & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$ |
| Quay quanh trục y một góc $\theta$ | $R(y,\theta) = \begin{bmatrix} C_\theta & -S_\theta & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -S_\theta & C_\theta & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$ |

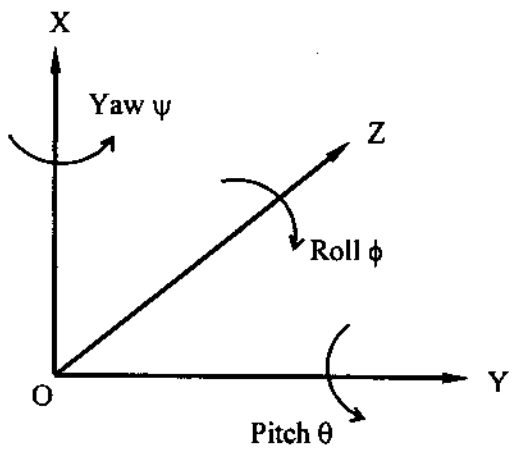
| | |
|---|---|
| Quay quanh trục z một góc $\psi$ | $R(z,\psi)=\begin{bmatrix} C_\psi & -S_\psi & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -S_\psi & C_\psi & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$ |

Ta có ma trận quay Euler như sau:

$$R(\phi,\theta,\psi)=\begin{bmatrix} C_\phi C_\theta C_\psi - S_\phi S_\psi & -C_\phi C_\theta C_\psi - S_\phi C_\psi & C_\phi S_\theta & 0 \\ S_\phi C_\theta C_\psi + C_\phi S_\psi & -S_\phi C_\theta S_\psi + C_\phi S_\psi & S_\phi S_\theta & 0 \\ S_\theta C_\psi & S_\theta S_\psi & C_\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (7.5)$$

***f. Phép quay Roll-Pitch-Yaw (RPY)***

***Thứ tự quay***: quay một góc $\psi$ quanh trục x, tiếp theo là quay một góc $\theta$ quanh trục y và sau đó là quay một góc $\phi$ quanh trục z.




***Hình 7.17.*** Phép quay Roll-Pitch-Yaw.

Theo thứ tự quay đó có thể biểu diễn phép quay RPY như sau:

$$RPY(\phi,\theta,\psi) = R(Z,\phi).R(Y,\theta).R(x,\psi) \quad (7.6)$$

Với:

| | |
|---|---|
| Quay quanh trục x một góc $\phi$: | $R(x,\phi)=\begin{bmatrix} C_\phi & -S_\phi & 0 & 0 \\ S_\phi & C_\phi & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$ |

| | |
|---|---|
| Quay quanh trục y một góc θ: | $R(y,\theta)=\begin{bmatrix} C_\theta & 0 & S_\theta & 0 \\ 0 & 1 & 0 & 0 \\ -S_\theta & 0 & C_\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$ |
| Quay quanh trục z một góc ψ: | $R(z,\psi)=\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & C_\psi & -S_\psi & 0 \\ 0 & S_\psi & C_\psi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$ |

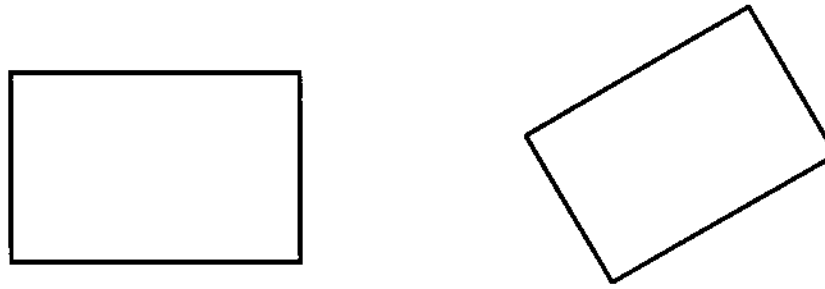
Thay vào phương trình 7.6 ta có:

$$RPY(\phi,\theta,\psi)=\begin{bmatrix} C_\phi C_\theta & C_\phi S_\theta S_\psi - S_\phi C_\psi & C_\phi S_\theta C_\psi + S_\phi S_\psi & 0 \\ S_\phi C_\theta & S_\phi S_\theta S_\psi + C_\theta C_\psi & S_\phi S_\theta C_\psi - C_\phi S_\psi & 0 \\ -S_\theta & C_\theta S_\psi & C_\theta C_\psi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (7.7)$$

Đoạn mã chương trình dưới đây vẽ một hình chữ nhật trong không gian vẽ, sau đó quay nó một góc $30^0$ sử dụng phép biến đổi ma trận như hình 7.18.

```
Public startPt(0 To 2) As Double
Public endPt(0 To 2) As Double
Sub VBA_TransformBy()
  ' Khai báo các biến.
  Dim startPt(0 To 2) As Double
  Dim endPt(0 To 2) As Double
  ' Định nghĩa hình chữ nhật.
  startPt(0) = 0: startPt(1) = 0: startPt(2) = 0
  endPt(0) = 50: endPt(1) = 25: endPt(2) = 0
  ' Gọi thủ tục vẽ hình chữ nhật.
  Call rectang(startPt(), endPt())
  ' Khai báo một ma trận.
  Dim transMat(0 To 3, 0 To 3) As Double
  ' Định nghĩa ma trận.
  transMat(0, 0) = Sqr(3) / 2#: transMat(0, 1) = -1 / 2#
  transMat(0, 2) = 0#: transMat(0, 3) = 0#
```

```
    transMat(1, 0) = 1 / 2#: transMat(1, 1) = -Sqr(3) / 2#
    transMat(1, 2) = 0#: transMat(1, 3) = 0#
    transMat(2, 0) = 0#: transMat(2, 1) = 0#
    transMat(2, 2) = 1#: transMat(2, 3) = 0#
    transMat(3, 0) = 0#: transMat(3, 1) = 0#
    transMat(3, 2) = 0#: transMat(3, 3) = 1#
    ' Phép biến đổi ma trận.
    rectang.TransformBy transMat
    ' Hiển thị đối tượng.
    rectang.Update
  End Sub
; Kết thúc.
```



***Hình 7.18.*** Di chuyển và xoay đối tượng.

## Chương 8
# HÌNH CẮT MẶT CẮT

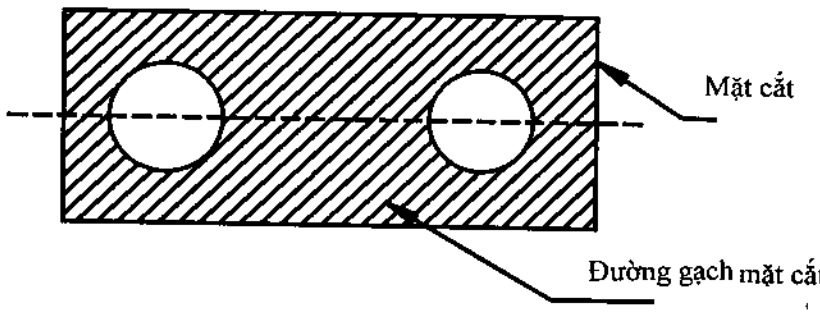
Chương này trình bày các lệnh gạch mặt cắt và đặt các thuộc tính của mặt cắt cũng như hiệu chỉnh các mặt cắt.

## 8.1. HÌNH CẮT MẶT CẮT

Khi biểu diễn trên bản vẽ gồm có hình chiếu, hình cắt và mặt cắt. Đối với các chi tiết có kết cấu phức tạp nếu chỉ sử dụng hình chiếu thôi thì chưa đủ. Vì như vậy trên các hình chiếu có nhiều đường khuất dẫn đến bản vẽ không được rõ ràng, sáng sủa. Do đó bản vẽ kỹ thuật dùng các hình biểu diễn khác nhau gọi là hình cắt và mặt cắt. Mặt cắt là một đối tượng của ACAD, do đó cho phép ta hiệu chỉnh (*di chuyển, tẩy, copy, đối xứng, tỷ lệ, đổi màu v.v...*). Bảng 8.1 là mẫu các mặt cắt theo tiêu chuẩn ISO và ANSI.

***Hình cắt:*** Là hình biểu diễn phần còn lại của vật thể sau khi tưởng tượng cắt bỏ phần vật thể nằm giữa mặt phẳng cắt và người quan sát.

***Mặt cắt:*** Là phần vật thể nằm trên mặt phẳng cắt và không vẽ vật thể nằm sau mặt phẳng cắt như hình 8.1.




***Hình 8.1.*** Ví dụ về mặt cắt.

## 8.2. KHỞI TẠO MẶT CẮT

Để gạch mặt cắt sử dụng lệnh: **AddHatch**

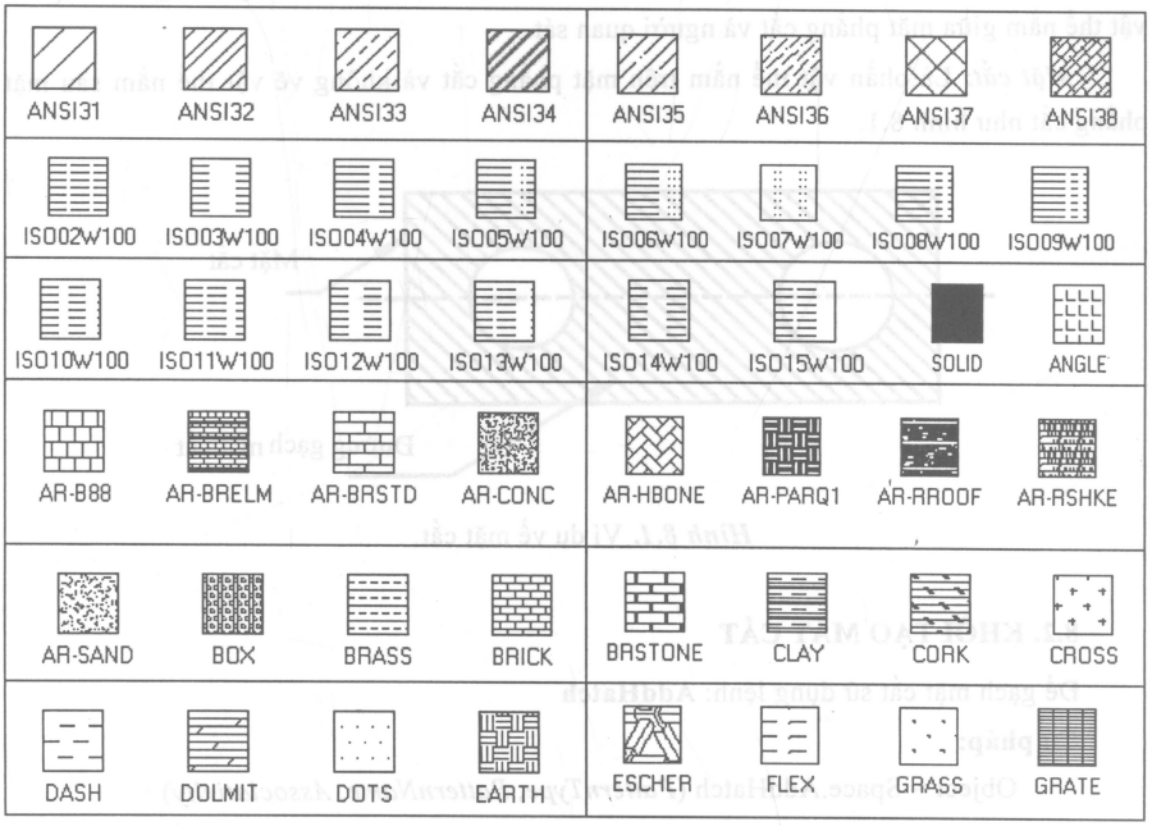
**Cú pháp:**

Object = Space.AddHatch (*PatternType*, *PatternName*, *Associativity*)

**Trong đó:**

| | |
|---|---|
| Object | Đối tượng mặt cắt. |
| Space | Không gian mô hình hoặc không gian giấy. |
| *PatternType* | Kiểu mặt cắt.<br>**acHatchPatternTypePredefined** *Cho phép ta chọn sẵn các mẫu có trong tập tin ACAD.PAT của AutoCAD bảng 8.1.*<br>**acHatchPatternTypeUserDefined** *Mẫu được chọn bằng file.PAT.*<br>**acHatchPatternTypeCustomDefined** Dùng để chọn mẫu có dạng các đoạn thẳng song song. |
| *PatternName* | Tên của mặt cắt cần sử dụng có kiểu dữ liệu String. |
| *Associativity* | Biến có kiểu Boolean<br>Trong đó có 2 giá trị trả về là **True** thì khi đó mặt cắt được khởi tạo, và khi nó có giá trị là **False** thì mặt cắt không được khởi tạo. |

***Bảng 8.1.*** Mẫu các mặt cắt



*Bảng 8.1.* (tiếp theo)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| HEX | HONEY | HOUND | INSUL | LINE | MUDST | NET | NET3 |
| PLAST | PLASTI | SACNCR | SQUARE | STARS | STEEL | SWAMP | TRANS |
| TRIANG | ZIGZAG | AR-B816 | AR-B816C | | | | |

Đoạn mã chương trình sau trình bày thủ tục tạo mặt cắt trong không gian vẽ.

```
' Thủ tục tạo mặt cắt trong VBA.
Public Sub Create_Hatch()
    ' Gọi layer hatch.
    Call Layer_change("hatch")
    ' Khai báo các biến.
    Dim patterName As String
    Dim patternType As Long
    Dim bAssciativity As Boolean
    ' Định nghĩa mặt cắt.
    patterName = "ANSI31"
    patternType = 0
    bAssciativity = True
    ' Tạo đối tượng mặt cắt trong không gian vẽ.
    Set HatchObj = acaddoc.ModelSpace.AddHatch(patternType, patterName,
    bAssciativity)
End Sub
```

***Để tìm hiểu rõ hơn về vấn đề này chúng ta đi tìm hiểu:***

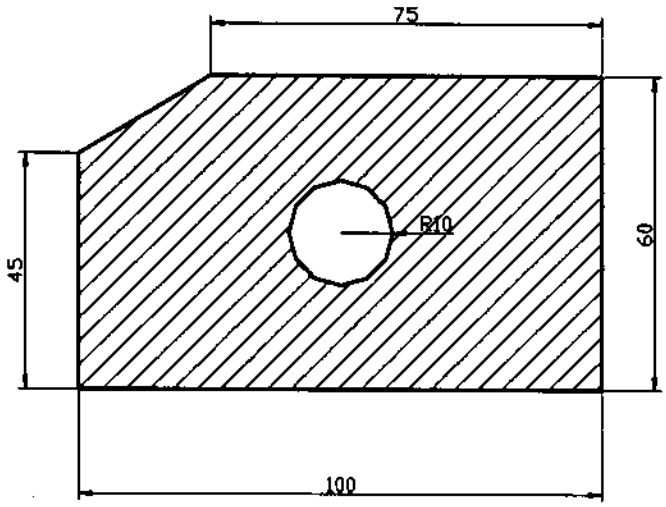
Đoạn mã chương trình sau vẽ một hình, ghi kích thước và gạch mặt cắt như hình 8.2.

```
; Tên file VBA_Hatch2.dvb.
Sub VBA_Hatch2()
    Dim hatchObj As AcadHatch
```

```
Dim patternName As String
Dim PatternType As Long
Dim bAssociativity As Boolean
' Tạo đường bao.
Dim plineObj As AcadLWPolyline
Dim CircleOj  As AcadCircle
Dim points(0 To 11) As Double
' Định nghĩa đường bao từ các điểm.
points(0) = 0 points(1) = 0
points(2) = 100 points(3) = 0
points(4) = 100 points(5) = 60
points(6) = 25 points(7) = 60
points(8) = 0 points(9) = 45
points(10) = 0 points(11) = 0
Set plineObj = ThisDrawing.ModelSpace.AddLightWeightPolyline(points)
' Chọn loại mặt cắt.
patternName = "ANSI31"
PatternType = 0
bAssociativity = True
' Tạo mặt cắt trong không gian vẽ.
Set hatchObj = ThisDrawing.ModelSpace.AddHatch(PatternType, patternName,
bAssociativity)
Dim outerLoop(0 To 0) As AcadEntity
Dim outerLoop1(0 To 0) As AcadEntity
Dim center(0 To 2) As Double
Dim radius As Double
center(0) = 50 center(1) = 30 center(2) = 0
radius = 10
Set circleObj = ThisDrawing.ModelSpace.AddCircle(center, radius)
Set outerLoop(0) = ThisDrawing.ModelSpace.AddLightWeightPolyline(points)
Set outerLoop1(0) = ThisDrawing.ModelSpace.AddCircle(center, radius)
hatchObj.AppendOuterLoop (outerLoop)
```

```
hatchObj.AppendOuterLoop (outerLoop1)
hatchObj.Evaluate
hatchObj.Color = acGreen
ThisDrawing.Regen True
' Ghi kích thước.
Dim dimObj As AcadDimAligned
Dim point1(0 To 2) As Double
Dim point2(0 To 2) As Double
Dim location(0 To 2) As Double
' Định nghĩa kích thước.
point1(0) = 0# point1(1) = 0# point1(2) = 0#
point2(0) = 100# point2(1) = 0# point2(2) = 0#
location(0) = 50# location(1) = -20# location(2) = 0#
' Tạo kích thước trong không gian vẽ.
Set dimObj = ThisDrawing.ModelSpace.AddDimAligned(point1, point2, location)
dimObj.Color = acRed
point1(0) = 100# point1(1) = 0# point1(2) = 0#
point2(0) = 100# point2(1) = 60# point2(2) = 0#
location(0) = 110# location(1) = 25# location(2) = 0#
Set dimObj = ThisDrawing.ModelSpace.AddDimAligned(point1, point2, location)
dimObj.Color = acRed
'Hàm ghi kích thước cạnh dọc bên trái
point1(0) = 0# point1(1) = 0# point1(2) = 0#
point2(0) = 0# point2(1) = 45# point2(2) = 0#
location(0) = -10# location(1) = 20# location(2) = 0#
Set dimObj = ThisDrawing.ModelSpace.AddDimAligned(point1, point2, location)
dimObj.Color = acRed
point1(0) = 100# point1(1) = 60# point1(2) = 0#
point2(0) = 25# point2(1) = 60# point2(2) = 0#
location(0) = 60# location(1) = 70# location(2) = 0#
Set dimObj = ThisDrawing.ModelSpace.AddDimAligned(point1, point2, location)
dimObj.Color = acRed
```

```
    ' Ghi kích thước đường kính.
    Dim dimObj_R As AcadDimRadial
    Dim chordPoint(0 To 2) As Double
    Dim leaderLen As Integer
    center(0) = 50# center(1) = 30# center(2) = 0#
    chordPoint(0) = 60# chordPoint(1) = 30# chordPoint(2) = 0#
    leaderLen = 5
    Set dimObj_R = ThisDrawing.ModelSpace.AddDimRadial(center, chordPoint,
    leaderLen)
    dimObj_R.Color = acRed
End Sub
; Kết thúc.
```




*Hình 8.2.*

## 8.3. CÁC THUỘC TÍNH CỦA MẶT CẮT

### 8.3.1. Chiều rộng bút vẽ

Để đặt chiều rộng bút vẽ khi gạch mặt cắt ta sử dụng thuộc tính **ISOPenWidth**.

**Cú pháp:**

Object.ISOPenWidth

**Trong đó:**

| | |
|---|---|
| Object | Đối tượng mặt cắt. |
| ISOPenWidth | Chiều rộng bút như bảng 8.3 |

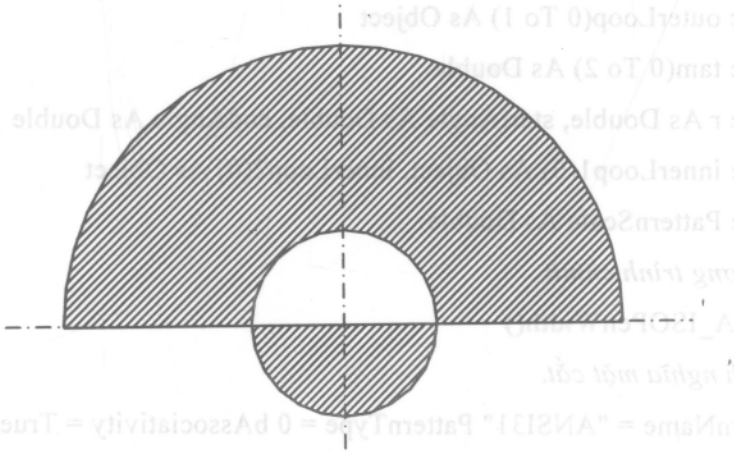
*Bảng 8.3.* Chiều rộng bút

| | |
|---|---|
| acPenWidth000 | 0.00 mm |
| acPenWidth013 | 0.13 mm |
| acPenWidth018 | 0.18 mm |
| acPenWidth025 | 0.25 mm |
| acPenWidth035 | 0.35 mm |
| acPenWidth050 | 0.50 mm |
| acPenWidth070 | 0.70 mm |
| acPenWidth100 | 1.00 mm |
| acPenWidth140 | 1.40 mm |
| acPenWidth200 | 2.00 mm |

Đoạn mã chương trình dưới đây vẽ một hình có gạch mặt cắt và sử dụng thuộc tính chiều rộng bút vẽ như hình 8.3.

```
' Khai báo các biến.
Public hatchObj As AcadHatch
Public patternName As String, PatternType As Long
Public bAssociativity As Boolean
Public outerLoop(0 To 1) As Object
Public tam(0 To 2) As Double
Public r As Double, startAngle As Double, endAngle As Double
Public innerLoop1(0) As Object, innerLoop2(0) As Object
Public PatternScale As Double
' Chương trình chính.
Sub VBA_ISOPenWidth()
' Định nghĩa mặt cắt.
patternName = "ANSI31" PatternType = 0 bAssociativity = True
' Tạo mặt cắt.
Set hatchObj = ThisDrawing.ModelSpace.AddHatch(PatternType, patternName,
bAssociativity)
tam(0) = 5 tam(1) = 3 tam(2) = 0
```

```
r = 300  startAngle = 0  endAngle = 3.141592
Set outerLoop(0) = ThisDrawing.ModelSpace.AddArc(tam, r, startAngle,
endAngle)
hatchObj.AppendOuterLoop (outerLoop)
tam(0) = 5 tam(1) = 4.5 tam(2) = 0
r = 100
Set innerLoop1(0) = ThisDrawing.ModelSpace.AddCircle(tam, r)
hatchObj.AppendInnerLoop (innerLoop1)
r = 0.5
Set innerLoop2(0) = ThisDrawing.ModelSpace.AddCircle(tam, r)
hatchObj.AppendInnerLoop (innerLoop2)
hatchObj.Evaluate
ThisDrawing.Regen True
'Sử dụng thuộc tính chiều rộng bút vẽ.
hatchObj.ISOPenWidth = acPenWidth200
hatchObj.Evaluate
ThisDrawing.Regen True
End Sub
; Kết thúc.
```



*Hình 8.3.*

### 8.3.2. Kiểu vẽ mặt cắt

Có ba kiểu **Normal, Outer** và **Ignore** như hình 8.4.


***Hình 8.4.*** Các kiểu mặt cắt.

Để chọn kiểu vẽ mặt cắt ta có cấu trúc như sau:

**Cú pháp:**

Object.HatchStyle

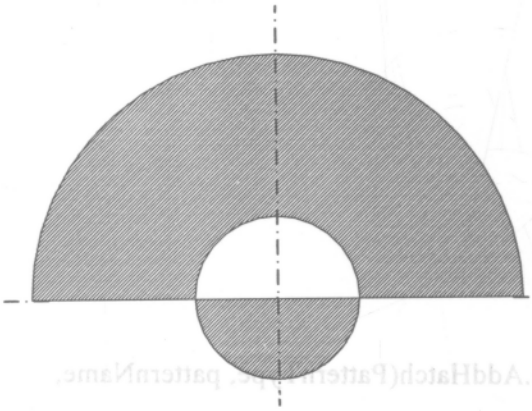
**Trong đó:**

| Object | Đối tượng mặt cắt. |
|---|---|
| HatchStyle | Kiểu mặt cắt, có 3 kiểu như hình 8.4:<br>*acHatchStyleNormal*<br>*acHatchStyleOuter*<br>*acHatchStyleIgnore* |

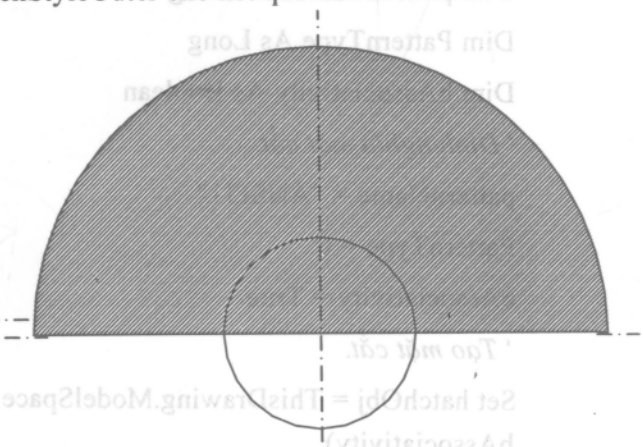
Từ ví dụ ở mục 8.3 nếu ta chọn kiểu vẽ mặt cắt **acHatchStyleNormal** cho kết quả như hình 8.5.

*hatchObj.HatchStyle = acHatchStyleNormal*

Nếu sử dụng kiểu vẽ mặt cắt acHatchStyleOuter cho kết quả như hình 8.6.



***Hình 8.5.*** Normal.



***Hình 8.6.*** Outer.

### 8.3.3. Độ nghiêng của các đường cắt so với mẫu chọn

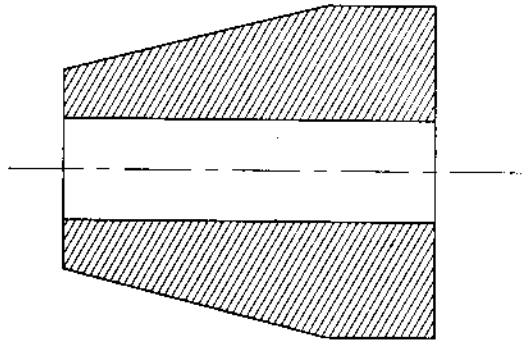
**Cú pháp:**

Object.PatternAngle

**Trong đó:**

| Object | Đối tượng mặt cắt. |
|---|---|
| PatternAngle | Độ nghiêng của các đường cắt so với mẫu chọn có kiểu dữ liệu là Double, đơn vị là radians và có miền giá trị trong khoảng từ 0 đến 2*Pi. Giá trị mặc định của góc là 0. |

Đoạn mã chương trình dưới đây vẽ chi tiết như hình 8.7 có độ nghiêng của gạch mặt cắt là 15 độ.



***Hình 8.7.***

```
; Tên file VBA_HatchAngule()
 Sub VBA_HatchAngule()
    Dim hatchObj As AcadHatch
    Dim patternName As String
    Dim PatternType As Long
    Dim bAssociativity As Boolean
    ' Định nghĩa mặt cắt.
    patternName = "ANSI31"
    PatternType = 0
    bAssociativity = True
    ' Tạo mặt cắt.
    Set hatchObj = ThisDrawing.ModelSpace.AddHatch(PatternType, patternName,
    bAssociativity)
     Dim outerLoop(0 To 0) As AcadEntity
    Dim plineObj As AcadLWPolyline
    Dim points(0 To 9) As Double
    points(0) = 100 points(1) = 100
```

```
    points(2) = 200 points(3) = 100
    points(4) = 200 points(5) = 60
    points(6) = 170 points(7) = 60
    points(8) = 100 points(9) = 30
    points(10) = 100 points(11) = 100
    Set outerLoop(0) = ThisDrawing.ModelSpace.AddLightWeightPolyline (points)
    hatchObj.AppendOuterLoop (outerLoop)
    Dim patternAngle As Double
    'Chọn góc nghiêng 15 độ.
    patternAngle = hatchObj.patternAngle
    hatchObj.patternAngle = 15 * 3.14 / 180
    hatchObj.Evaluate
    ThisDrawing.Regen True
  End Sub
; Kết thúc.
```

### 8.3.4. Đặt khoảng cách giữa các đường gạch chéo

**Cú pháp:**

Object.PatternSpace

**Trong đó:**

| Object | Đối tượng mặt cắt |
|---|---|
| PatternSpace | Khoảng cách giữa các đường gạch chéo của mặt cắt có kiểu dữ liệu là Double. |

Cách khai báo sử dụng thuộc tính **PatternSpace** để điều khiển khoảng cách giữa các đường gạch chéo.

```
Dim patternSpace As Double
hatchObj.patternSpace = 2
hatchObj.Evaluate
```

## Chương 9

# LỚP VÀ CÁC THUỘC TÍNH CỦA LỚP

Chương này trình bày các lệnh tạo lớp và đặt các thuộc tính của lớp như kiểu đường, màu v.v...

## 9.1. TẠI SAO PHẢI TẠO LỚP?

Trong bản vẽ có nhiều đối tượng có cùng thuộc tính giống nhau, những đối tượng này được nhóm lại thành từng nhóm gọi là lớp (*layer*). Khi ta thay đổi thuộc tính của lớp này thì tất cả các đối tượng trên lớp đó sẽ thay đổi theo. Ví dụ lớp đường tâm (*có kiểu đường là đường tâm, màu xanh*) những đối tượng thuộc lớp này sẽ có cùng thuộc tính trên, nếu ta thay đổi thuộc tính thì tất cả các đối tượng trên lớp đó sẽ thay đổi.

***Tạo lớp có lợi ích gì?***

Câu hỏi trên được trả lời như sau:

+ *Khi tạo các lớp sẽ tạo điều kiện thuận lợi hơn trong quá trình quản lý bản vẽ.*
+ *Bỏ bớt được các thao tác lặp lại trong quá trình thiết kế.*
+ *Thuận tiện cho việc hiệu chỉnh bản vẽ.*
+ *Giúp cho công việc thiết kế được nhanh hơn, bản vẽ được trình bày khoa học hơn v.v...*

## 9.2. TẠO MỘT LỚP MỚI

Để tạo một lớp và đặt thành lớp hiện hành ta sử dụng thuộc tính **ActiveLayer**.

**Cú pháp:**

Object.ActiveLayer

**Trong đó:**

| | |
|---|---|
| Object | Các đối tượng trong bản vẽ. |
| ActiveLayer | Lớp hiện hành. |

Trong ACAD thường mặc định lớp ban đầu là lớp "0". Để tạo lớp mới ta sử dụng dòng lệnh dưới đây:

```
Set newlayer = ThisDrawing.Layers.Add("LAYER1")
```

Đoạn mã chương trình dưới đây sẽ khai báo lớp, tạo một lớp mới ("LAYER1") và đặt thành lớp hiện hành.

```
Dim newlayer As AcadLayer                          ' Khai báo một lớp mới
Set newlayer = ThisDrawing.Layers.Add("LAYER1")    ' Tạo lớp ("LAYER1")
ThisDrawing.ActiveLayer = newlayer                 ' Đặt nó thành lớp hiện hành
```

Để tìm hiểu sâu hơn chúng ta tìm hiểu thêm một ví dụ nữa, đoạn mã chương trình sau tạo một lớp mới có tên ("TestLayer") trong bản vẽ hiện hành, khi chạy chương trình sẽ hiển thị lên thông báo về lớp đang hiện hành trong bản vẽ, sau đó thông báo lớp mới trong bản vẽ là ("TestLayer") và đặt thành lớp hiện hành, khi OK thì đặt lại lớp hiện hành ban đầu là lớp ("0").

```
; Tên file: ActiveLayer.dvb
Sub ActiveLayer()                                  ' Tên chương trình
  Dim currLayer As AcadLayer                       ' Khai báo lớp đang hiện hành
  Dim newLayer As AcadLayer                        ' Khai báo lớp mới
  Set currLayer = ThisDrawing.ActiveLayer          ' Nhận lớp hiện hành
  MsgBox "The current layer is " &                 ' Thông báo các lớp đã có trong
  currLayer.Name, vbInformation, "ActiveLayer      bản vẽ.
  Example"
  Set newLayer =                                   ' Tạo một lớp mới có tên là
  ThisDrawing.Layers.Add("TestLayer")              ("TestLayer"), sử dụng phương
                                                   thức Add.
  ThisDrawing.ActiveLayer = newLayer               'Đưa lớp mới thành lớp hiện hành
  MsgBox "The new layer is " & newLayer.Name,      'Thông báo lớp mới đó đã được
  vbInformation, "ActiveLayer Example"             tạo, và hiện hành.
  ThisDrawing.ActiveLayer = currLayer              'Hiện hành lớp mặc định ban đầu
                                                   ("0") của bản vẽ
  MsgBox "The active layer is reset to " &         'Thông báo lớp đang hiện hành là
  currLayer.Name, vbInformation, "ActiveLayer      lớp mặc định ban đầu.
  Example"
End Sub
; Kết thúc.
```

### 9.3. SẮP XẾP CÁC LỚP

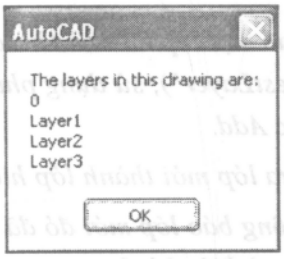
VBA hỗ trợ việc sắp xếp các lớp trong bản vẽ để dễ quản lý các đối tượng.

Đoạn mã chương trình sắp xếp tất cả các lớp trong một bản vẽ, và hiện thị tất cả các lớp đó trên một thông báo.

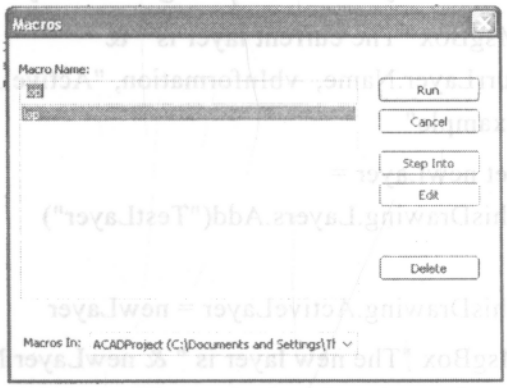
```
; Tên file: lop.dvb.
  Sub lop()                                        ' Tên chương trình
      Dim layerNames As String                     ' Khai báo tên lớp
      Dim entry As AcadLayer
      layerNames = ""
      For Each entry In ThisDrawing.Layers
      layerNames = layerNames + entry.Name + vbCrLf
      Next
      MsgBox "The layers in this drawing are: " + _ vbCrLf + layerNames
  End Sub
; Kết thúc chương trình: lop.dvb
```

Thực hiện việc chạy chương trình:

Khi chạy chương trình hộp thoại thông báo trên hình 9.1 và tất cả các lớp được thông báo như hình 9.2




*Hình 9.1.*




*Hình 9.2.*

## 9.4. ĐẶT TÊN CHO LỚP

Khi tạo một lớp mới ACAD thường mặc định tên lớp layer1, layer2, layer3…tuỳ thứ tự layer được tạo mà có chữ số đằng sau 1, 2, 3, 4, 5 v.v..., để lớp có tên tường minh, có ý nghĩa trong quản lý bản vẽ ta đặt lại tên, để làm điều đó sử dụng thuộc tính **name** có cú pháp như sau:

Object.Name

**Trong đó:**

| | |
|---|---|
| Object | Đối tượng lớp. |
| Name | Tên lớp có kiểu dữ liệu String. |

*Chú ý: Tên của một layer dài tối đa 31 ký tự bao gồm các chữ, số, ký tự đặc biệt ($, - , _) nhưng không được có dấu cách.*

Đoạn mã chương trình sau tạo một lớp mới có tên là ("NewLayer") và sau đó đổi tên lớp đó thành lớp ("duongtam").

; Tên file: Name_layer.dvb.

```
Sub Name_layer()                                    'Tên chương trình
  Dim layerObj As AcadLayer                         'Khai báo tên lớp
  Set layerObj = ThisDrawing.Layers.Add("NewLayer") 'Tạo lớp mới
  Dim layerName As String                           'Khai báo tên lớp có kiểu
                                                    String.
  layerName = layerObj.Name                         'Nhận tên lớp
  MsgBox "A new layer was created with the name:
   " & layerObj.Name, , "Name Example"
  layerObj.Name = "duongtam"                        'Đổi tên lớp
  layerName = layerObj.Name
  MsgBox "The new name of the layer is:             'Thông báo tên lớp mới được
  " & layerObj.Name, , "Name Example"               tạo.
End Sub
```

; Kết thúc.

### 9.5. ẨN, HIỆN LỚP

Đôi khi trong một bản vẽ có rất nhiều lớp tùy thuộc vào độ phức tạp của bản vẽ và số lớp được tạo nhiều khi người thiết kế muốn ẩn hay hiện một hoặc một số lớp để thuận tiện cho quá trình thiết kế, để làm điều này ta sử dụng thuộc tính **LayerOn**.

**Cú pháp:**

Object.LayerOn

**Trong đó:**

| | |
|---|---|
| Object | Đối tượng lớp. |
| LayerOn | Có kiểu Boolean, nó có hai giá trị:<br>▪ True: Bật lớp.<br>▪ False: Tắt lớp. |

Đoạn mã chương trình sau sẽ thực hiện việc mở hoặc tắt một lớp mới tạo trong bản vẽ.

```
; Tên file: Mo_Tat_lop.dvb.
Sub Mo_Tat_lop()
   Dim layerObj As AcadLayer
   Set layerObj = ThisDrawing.Layers.Add("LayerOn")
   'Tạo một lớp mới là ("LayerOn")
   GoSub DISPLAYSTATUS
   layerObj.LayerOn = Not (layerObj.LayerOn)
   'Tắt lớp: Not (layerObj.LayerOn)
   GoSub DISPLAYSTATUS
   Exit Sub
   DISPLAYSTATUS:
   If layerObj.LayerOn Then
   MsgBox "Layer " & layerObj.Name & " is turned on.", , "LayerOn Example"
   Else
   MsgBox "Layer " & layerObj.Name & " is turned off.", , "LayerOn Example"
   End If
   Return
End Sub
; Kết thúc.
```

## 9.6. ĐÓNG VÀ LÀM TAN BĂNG LỚP TRÊN KHUNG NHÌN

Để làm đóng băng, tan băng lớp sử dụng thuộc tính **Freeze**.

**Cú pháp:**

*<tên lớp>*.Freeze

**Trong đó:**

**Freeze**: Có kiểu dữ liệu Boolean với hai giá trị như sau:

| | |
|---|---|
| True | Đóng băng lớp. |
| False | Tan băng lớp. |

## 9.7. KHOÁ VÀ MỞ KHOÁ CHO LỚP

Để khoá và mở khoá cho lớp ta sử dụng thuộc tính **Lock**.

**Cú pháp:**

*<tên lớp>*.Lock

**Trong đó:**

**Lock**: Có kiểu dữ liệu Boolean với hai giá trị như sau:

| | |
|---|---|
| True | Khoá lớp. |
| False | Mở khoá lớp. |

## 9.8. GÁN MÀU CỦA LỚP

Trong bản vẽ để dễ dàng quản lý các đối tượng theo thuộc tính người thiết kế đặt màu cho các lớp. Để đặt màu cho lớp chúng ta sử dụng thuộc tính **Color**.

**Cú pháp:**

Object.Color

**Trong đó:**

| | |
|---|---|
| Object | Đối tượng lớp. |
| Color | Giá trị màu muốn sử dụng như bảng 9.1. |

***Bảng 9.1.*** Bảng màu cơ bản

| Màu | Chỉ số |
|---|---|
| acRed | 1 |
| acYellow | 2 |
| acGreen | 3 |
| acCyan | 4 |
| acBlue | 5 |
| acMagenta | 6 |
| acWhite | 7 |

***Chú ý:*** *Có thể gán màu cho lớp ngoài các màu cơ bản ra, nếu muốn gán các màu đặc biệt có thể thêm tên của màu hoặc các chỉ số ACI.*

***Ví dụ:***

*'đặt màu đỏ cho lớp ("layerObj") là màu đỏ*

layerObj.Color = acRed (*hoặc 1*)

## 9.9. XOÁ LỚP

Để xoá một lớp sử dụng lệnh **Delete**, có thể xóa bất kỳ một lớp nào trong bản vẽ ngoại trừ lớp mặc định là lớp ("0") hoặc các lớp từ bên ngoài.

**Cú pháp:**

<*Tên_lớp*>.Delete

Đoạn mã chương trình này tạo một lớp mới là ("TEST") sau đó xóa lớp đó.

```
Sub VBALayer_Delete()
   Dim layerObj As AcadLayer
   Set layerObj = ThisDrawing.Layers.Add("TEST")
   'Xoá lớp.
   layerObj.Delete
Exit Sub
```

## 9.10. LÀM VIỆC VỚI KIỂU MÀU

Khi làm việc với các kiểu màu trong bản vẽ sử dụng đối tượng **AcCmColor**. Sử dụng các giá trị trong đối tượng **AcCmColor**, có thể lựa chọn hàng triệu màu khác nhau khi muốn đặt màu cho đường thẳng, đường tròn và tất cả các đối tượng khác trong bản vẽ. Đối tượng **AcCmColor** có các phương thức, thuộc tính đặc biệt như: (*tên màu, khối màu, phương thức thể hiện, các giá trị màu* ). Ngoài ra còn có thể đặt màu cho các lớp, mỗi màu được xác định bằng một tên riêng hoặc các chỉ số màu, là các số nguyên có giá trị từ (1 ÷ 255).

**Cú pháp:**

Object.Color

**Trong đó:**

| | |
|---|---|
| Object | Các đối tượng trong bản vẽ. |
| Color | Màu muốn thiết lập. |

Đối tượng **AcCmColor**:

Trong VBA đối tượng **AcCmColor**: thuộc lớp **AcadAcCmColor**.

Khởi tạo đối tượng màu trong VBA.

GetInterfaceObject ("AutoCAD.AcCmColor.16"), hoặc Dim col As New AcadAcCmColor

Đối tượng này bao gồm tất cả các thuộc tính của các màu.

Đoạn mã chương trình sau vẽ một đường tròn trong không gian vẽ với màu là xanh da trời.

```
; Tên file : VBA_Color.dvb.
Sub VBA_Color()
```

```
    Dim color As AcadAcCmColor
    'Khai báo đối tượng màu color có kiểu AcadAcCmColor trong VBA.
    Set color = _
    AcadApplication.GetInterfaceObject("AutoCAD.AcCmColor.16")
    Call color.SetRGB(80, 100, 244)
    Dim circleObj As AcadCircle
    Dim centerPoint(0 To 2) As Double
    Dim radius As Double
    centerPoint(0) = 0#: centerPoint(1) = 0#: centerPoint(2) = 0#
    radius = 5#
    Set circleObj = _
    ThisDrawing.ModelSpace.AddCircle(centerPoint, radius)
    circleObj.TrueColor = color
    ZoomAll
End Sub
; Kết thúc.
```

> ***Chú ý:*** *Màu mặc định là 7 có thể là màu trắng hoặc đen tùy thuộc vào màu nền của bản vẽ.*

## 9.11. LÀM VIỆC VỚI KIỂU ĐƯỜNG

Tùy thuộc vào bản vẽ kỹ thuật mà ta sử dụng các kiểu đường khác nhau. Các dạng đường có trong file ***acad.lin*** và mã VBA được trình bày như trong bảng 9.2.

***Bảng 9.2.*** Các kiểu đường

| Tên đường | Hình dáng |
|---|---|
| ACAD_ISO02W100 | |
| ACAD_ISO03W100 | |
| ACAD_ISO04W100 | |
| ACAD_ISO05W100 | |
| ACAD_ISO06W100 | |
| ACAD_ISO07W100 | |
| ACAD_ISO08W100 | |

*Bảng 9.2.* (tiếp theo)

| | |
|---|---|
| ACAD_ISO09W100 | |
| ACAD_ISO10W100 | |
| ACAD_ISO11W100 | |
| ACAD_ISO12W100 | |
| ACAD_ISO13W100 | |
| ACAD_ISO14W100 | |
| ACAD_ISO15W100 | |
| BATTING | SSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSSS |
| BORDER | |
| BORDER2 | |
| BORDERX2 | |
| CENTER | |
| CENTER2 | |
| CENTERX2 | |
| DASHDOT | |
| DASHDOT2 | |
| DASHDOTX2 | |
| DASHED | |
| DASHED2 | |
| DASHEDX2 | |
| DIVIDE | |
| DIVIDE2 | |
| DIVIDEX2 | |
| DOT | |
| DOT2 | |
| DOTX2 | |
| FENCELINE1 | ----0-----0----0-----0----0--- |
| FENCELINE2 | ----[]-----[]----[]----[]---- |
| GAS_LINE | -GAS----GAS----GAS----GAS----GAS--- |
| HIDDEN | |

*Bảng 9.2.* (tiếp theo)

| | |
|---|---|
| HIDDEN2 | - - - - - - - - - - - - - - - - |
| HIDDENX2 | — — — — — — — — — — |
| HOT_WATER_SUPPLY | --- HW ---- HW ---- HW ---- |
| PHANTOM | — — — — — — — — — — — — |
| PHANTOM2 | — - - — - - — - - — - - |
| PHANTOMX2 | — — — — — — — — — — |
| TRACKS | ++++++++++++++++++ |
| ZIGZAG | \/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/\/ |

### 9.11.1. Khởi tạo kiểu đường

Để sử dụng các kiểu đường trong **acad.lin** ta phải sử dụng thuộc tính **Linetype**.

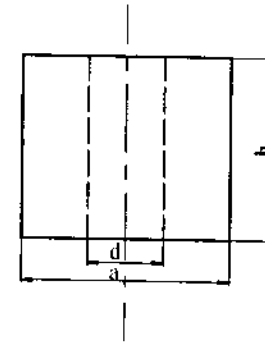
**Cú pháp:**

Object.Linetype

**Trong đó:**

| | |
|---|---|
| Object | Các đối tượng trong bản vẽ. |
| Linetype | Kiểu đường trong file acad.lin. |

Đoạn mã chương trình dưới đây vẽ hình trụ đường kính a, chiều cao h, đường kính lỗ trong là d như hình 9.3, đường tâm là center, đường khuất là ACAD_ISOO2W100.




*Hình 9.3.*

```
; Tên file VBA_Linetype.dvb.
  Public FirstPoint(2) As Double
  Public EndPoint(2) As Double
  Public tam(2) As Double
  ' Thủ tục gọi kiểu đường trong file acad.lin.
  Public Sub Linetype(name As String)
    Dim entry As AcadLineType
    Dim found As Boolean
```

```
    found = False
    For Each entry In acaddoc.Linetypes
        If StrComp(entry.name, name, 1) = 0 Then
            found = True
            Exit For
        End If
    Next
    If Not (found) Then ThisDrawing.Linetypes.Load name, "acad.lin"
End Sub
' Chương trình chính.
Sub main_Linetype()
  tam(0) = 100: tam(1) = 100: tam(2) = 0
  FirstPoint(0) = tam(0): FirstPoint(1) = tam(1): FirstPoint(2) = tam(2)
  EndPoint(0) = tam(0) + a: EndPoint(1) = tam(1) + h: EndPoint(2) = tam(2)
  ' Gọi thủ tục chọn kiểu đường.
  Call Linetype(CONTINUOUS)
  ' Thủ tục vẽ hình chữ nhật.
  Call Rectang(FirstPoint(), EndPoint())
  Call Linetype(Center)
  FirstPoint(0) = tam(0) + a / 2: FirstPoint(1) = tam(1): FirstPoint(2) = tam(2)
  EndPoint(0) = tam(0) + a / 2: EndPoint(1) = tam(1) + h: EndPoint(2) = tam(2)
  Call AddLine(FirstPoint(), EndPoint())
  Call Linetype(ACAD_ISO02W100)
  FirstPoint(0) = tam(0) + a / 4: FirstPoint(1) = tam(1): FirstPoint(2) = tam(2)
  EndPoint(0) = tam(0) + a / 4: EndPoint(1) = tam(1) + h: EndPoint(2) = tam(2)
  ' Gọi thủ tục vẽ đường thẳng.
  Call AddLine(FirstPoint(), EndPoint())
  FirstPoint(0) = tam(0) + a / 2: FirstPoint(1) = tam(1): FirstPoint(2) = tam(2)
  EndPoint(0) = tam(0) + a / 2: EndPoint(1) = tam(1) + h: EndPoint(2) = tam(2)
  ' Gọi thủ tục lấy đối xứng đối tượng.
  Call Mirror(FirstPoint(), EndPoint())
  ZoomAll
End Sub
; Kết thúc.
```

### 9.11.2. Độ rộng của đường

Muốn thay đổi độ rộng đường của các đối tượng trong bản vẽ sử dụng thuộc tính **Lineweight**.

**Cú pháp:**

Object.Lineweight

**Trong đó:**

| | |
|---|---|
| Object | Các đối tượng trong bản vẽ. |
| Lineweight | Độ rộng của đường như bảng 9.3. |

Đoạn mã chương trình sau vẽ đường tròn và đặt độ rộng của đường có kiểu là: acLnWt013 (0.13 mm).

```
; Tên file VAB_LineWeight.dvb.
  Sub VAB_LineWeight()
    Dim circleObj As AcadCircle
    Dim centerPoint(0 To 2) As Double
    Dim radius As Double
    centerPoint(0) = 0#: centerPoint(1) = 0#: centerPoint(2) = 0#
    radius = 5#
    Set circleObj = ThisDrawing.ModelSpace.AddCircle(centerPoint, radius)
    circleObj.Lineweight = acLnWt013
    circleObj.Update
    ZoomAll
  End Sub
; Kết thúc.
```

***Bảng 9.3.*** Bảng độ rộng của đường

| Độ rộng của đường trong VBA | |
|---|---|
| acLnWtByLayer | ——— ByLayer |
| acLnWtByBlock | ——— ByBlock |
| acLnWtByLwDefault | ——— Default |
| acLnWt000 | ——— 0.00 mm |
| acLnWt005 | ——— 0.05 mm |
| acLnWt009 | ——— 0.09 mm |

*Bảng 9.3.* (tiếp theo)

| | |
|---|---|
| acLnWt013 | 0.13 mm |
| acLnWt015 | 0.15 mm |
| acLnWt018 | 0.18 mm |
| acLnWt020 | 0.20 mm |
| acLnWt025 | 0.25 mm |
| acLnWt030 | 0.30 mm |
| acLnWt035 | 0.35 mm |
| acLnWt040 | 0.40 mm |
| acLnWt050 | 0.50 mm |
| acLnWt053 | 0.53 mm |
| acLnWt060 | 0.60 mm |
| acLnWt070 | 0.70 mm |
| acLnWt080 | 0.80 mm |
| acLnWt090 | 0.90 mm |
| acLnWt100 | 1.00 mm |
| acLnWt106 | 1.06 mm |
| acLnWt120 | 1.20 mm |
| acLnWt140 | 1.40 mm |
| acLnWt158 | 1.58 mm |

## 9.12. GÁN DẠNG ĐƯỜNG CHO LỚP

Để đặt kiểu đường cho một lớp đối tượng tức là cho tất cả các đối tượng trong lớp có cùng một kiểu đường.

**Cú pháp:**

<*Tên_lớp*>.Linetype = "kiểu đường"

**Trong đó:**

"kiểu đường": trình bày trong bảng 9.2.

*Ví dụ:* Đoạn mã chương trình dưới đây trình bày thủ tục tạo một đường tâm trong không gian vẽ.

```
Public Sub line_center(x1 As Double, y1 As Double, x2 As Double, y2 As Double)
  ' Gọi lớp đường tâm
  Call Layer_change("center")
```

```
    Dim points(0 To 3) As Double
    Dim plineObj As AcadLWPolyline
    ' Định nghĩa đường tâm
    points(0) = x1: points(1) = y1
    points(2) = x2: points(3) = y2
    Set plineObj = acaddoc.ModelSpace.AddLightWeightPolyline(points)
End Sub
```

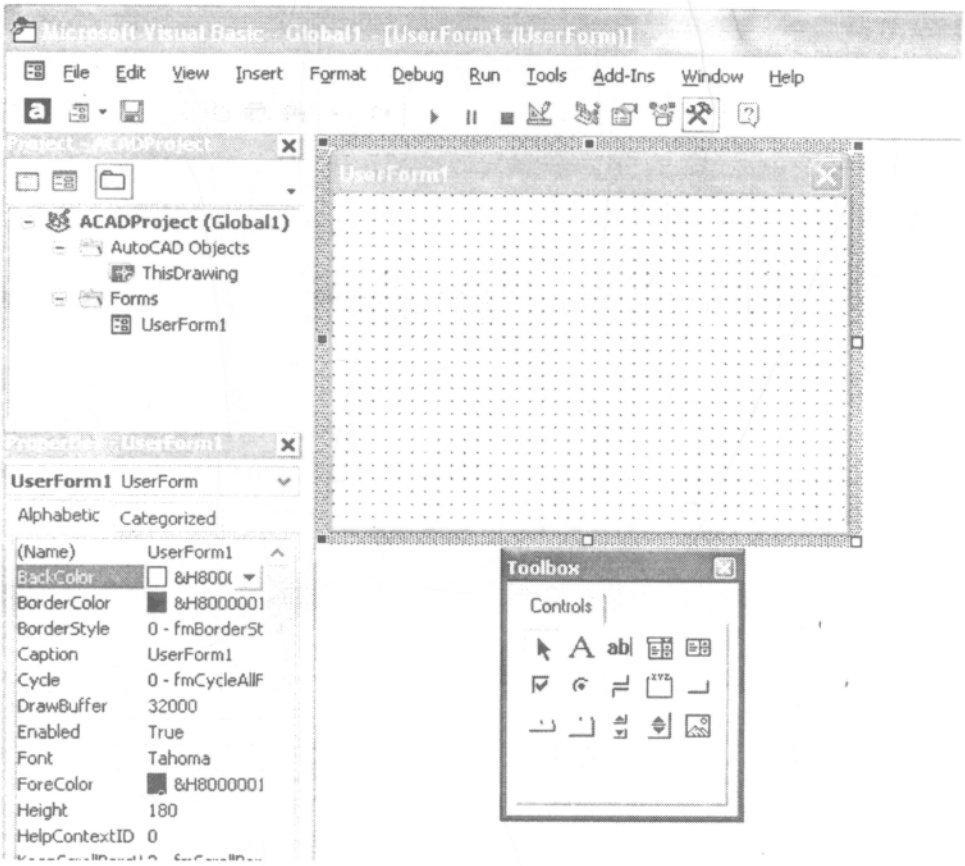
## Chương 10

# PHÁT TRIỂN ỨNG DỤNG VỚI VBA

## 10.1. LÀM VIỆC VỚI FORM

### 10.1.1. Thêm Form vào dự án

Các Form là các đối tượng riêng biệt để thêm vào các dự án (*Project*) của VB&VBA. Để làm điều này theo trình tự như sau: mở môi trường soạn thảo *(Visual Basic Editor)*, trên thanh công cụ **ToolBar** chọn Insert ở hộp thoại **UserForm** hoặc có thể sổ xuống nút công cụ **Insert** *(thứ hai tính từ bên trái)* và chọn **UserForm** VBA sẽ thể hiện các tác vụ sau như hình 10.1.




*Hình 10.1.* Thêm Form vào dự án.

- *Thêm một nhánh **Form** vào cây **Project Explorer**.*
- *Tạo qua một đối tượng **UserForm** mới và thêm vào nhánh **Form**.*
- *Hiển thị **Form** trong vùng làm việc.*
- *Hiển thị hộp công cụ.*

### 10.1.2. Thay đổi các thuộc tính của Form khi thiết kế

Các Form cùng các đối tượng điều khiển trên **Form**, đều có các chung thuộc tính của sổ thuộc tính. Để hiển thị thuộc tính (*Properties*) kích chọn vào đối tượng hoặc bằng cách nhấn phím F4.

> *Chú ý: Bên cạnh việc hiệu chỉnh Form và các đối tượng điều khiển trên Form, ngoài việc thay đổi các thuộc tính của Form (Properties), còn bằng cách đưa vào các câu lệnh thích hợp trong các modul của VBA.*

### 10.1.3. Hiển thị Form

Mỗi đối tượng **UserForm** có một phương thức hiển thị *(Show)* dùng để hiển thị Form.

*Ví dụ:* Để hiển thị một Form có tên là ACadForm1, sử dụng câu lệnh như sau:

```
ACadForm1.Show.
```

Ngoài ra khi muốn tải Form vào bộ nhớ dùng lệnh như sau:

```
Load Form
```

Trong đó Form là tên form mà bạn muốn tải.

*Ví dụ:*

```
Private Sub UserForm_Initialize()
  Load UserForm2
  UserForm2.Show
End Sub
```

### 10.1.4. Hủy tải Form

Để tắt hiển thị Form sử dụng câu lệnh như sau: **Unload Me**. Trong đó từ khoá **Me** tham chiếu đến **Form** mà sự kiện là của **Form** đó.

*Ví dụ:*

```
Private Sub UserForm_Click()
  Unload UserForm1
End Sub
```

## 10.2. LÀM VIỆC VỚI CÁC ĐIỀU KHIỂN

### 10.2.1. Thêm các điều khiển lên Form

Hộp thoại **ToolBox** chứa tất cả các điều khiển để thêm vào **Form**. Thực hiện công việc đưa các điều khiển lên **Form** ta thực hiện các trình tự như sau:

+ Nhấp nút muốn sử dụng vào chương trình.

+ Di chuyển con chuột lên Form và đặt điều khiển đó vào vị trí bất kỳ trên Form tùy theo thiết kế của người sử dụng.

### 10.2.2. Các điều khiển của Form

#### *10.2.2.1. Nhãn (Label)*

Sử dụng điều khiển nhãn *(label )* dùng để thêm kiểu chữ lên Form. Sử dụng nút label trong hộp công cụ ToolBox để vẽ các đối tượng nhãn, sau đó sửa các thuộc tính Caption để thay đổi hoặc thêm các nhãn vào chương trình. Nhãn thường dùng để hiển thị các chữ, ngoài ra còn sử dụng chúng để đặt tên cho các điều khiển khác mà không có phần Caption riêng của chúng, chẳng hạn như các **TextBox**, các **ListBox**, các thanh cuộn **Scoll bars**, các thanh trượt **Spinner**.

#### *10.2.2.2. Các TextBox*

Các **TextBox** là các điều khiển mà cho phép người dùng nhập chữ hoặc số. Để tạo các **TextBox** vào hộp công cụ **ToolBox** và vẽ nó vào biểu mẫu.

+ Vào **Properties** để thay đổi các thuộc tính của **Text Box**. Tìm thuộc tính **Font**, có thể thay đổi kiểu **Font** (*đậm, nghiêng, gạch dưới*), kích cỡ, màu sắc...

+ Thuộc tính ***BackColor*** của văn bản để sửa màu nền.

+ Sửa thuộc tính ***Multiline*** của hộp văn bản hiện hành thành ***True***, nghĩa là khi văn bản quá dài, điều khiển này sẽ tự động xuống hàng.

+ Không cho gõ vào hộp văn bản bằng cách sửa thuộc tính ***Locked*** thành ***True***.

> *<u>Chú ý:</u> Hộp văn bản không tự kiểm tra giá trị nhập vào, người lập trình phải làm việc đó. Ví dụ nếu thay đổi thuộc tính MaxLength thành một con số, ví dụ số 5, khi đó ta chưa nhập được 5 ký tự (thuộc tính Locked) phải đổi thành False). Nếu đổi MaxLength về 0, khi đó có thể gõ vào bao nhiêu tuỳ người sử dụng.*
>
> *Sự kiện* **KeyPress** *được phát ra khi người sử dụng gõ vào hộp văn bản.*

#### *10.2.2.3. Khung (Frame)*

Sử dụng các khung để tạo các nhóm hay hai nhiều điều khiển. Các thuận lợi khi bạn dùng Frame.

#### *10.2.2.4. Các nút chọn lựa (Option)*

Các nút **Option** là các điều khiển hay xuất hiện trong các nhóm hai hay nhiều; người dùng có thể chỉ chọn một trong các nút tuỳ chọn. Để tạo một nút **Option**, sử dụng công cụ **OptionButton**. Bạn xác định nút bắt đầu được kích hoạt, hay huỷ nếu thuộc tính **Value**, nếu là **True** thì sẽ tùy chọn được kích hoạt, còn nếu là **False** tùy chọn sẽ hủy kích hoạt.

Để sử dụng điều khiển lựa chọn một cách hiệu quả, cần nhóm các lựa chọn đó để người dùng có thể chỉ chọn một nút tuỳ chọn một lần. Để thực hiện điều đó VBA có ba cách như sau:

+ Tạo một khung và sau đó vẽ các nút tuỳ chọn bên trong khung.

+ Sử dụng các thuộc tính **GroupName** cho các tuỳ chọn khi muốn nhóm lại.

+ Nếu không vẽ các nút **Options** bên trong một khung hoặc sử dụng thuộc tính **GroupName** thì VBA sẽ xem tất cả các nút tùy chọn trong một Form dưới dạng một nhóm.

***10.2.2.5. Hộp kiểm***

Các hộp kiểm cho phép đưa vào các tuỳ chọn mà người dùng có thể chuyển qua lại giữa hai trạng thái **On** và **Off**. Để tạo một hộp kiểm trong hộp công cụ ToolBox, bằng cách nhấn nút **CheckBox**.

Cũng như với các nút tuỳ chọn có thể điều khiển một hộp kiểm có được kích hoạt từ ban đầu hay không, xác lập giá trị Value là True để kích hoạt hộp kiểm, và False để hủy hộp kiểm.

***10.2.2.6. Nút chuyển đổi (Toggle)***

Một nút chuyển đổi *(nút bật lên xuống)* là một sự cộng tác giữa một hộp kiểm và một nút lệnh: Khi nhấp một lần, nút sẽ ở trạng thái được giữ; nhấp một lần nữa, nút sẽ trở về trạng thái bình thường. Bạn tạo các nút chuyển đổi (*toggle*) bằng cách sử dụng công cụ **ToggleButton** trong hộp công cụ **ToolBox.**

Điều khiển một nút chuyển đổi *(toggle)* có được kích hoạt từ ban đầu hay không *(tức là được nhấn hay không)* bằng cách xác lập giá trị Value của là True để chọn nút hoặc False để hủy chọn nút.

***10.2.2.7. Các ListBox***

VBA đưa ra hai đối tượng danh sách khác nhau để có thể sử dụng để cho người dùng một danh sách các lựa chọn như: một **ListBox** và một **ComboBox.**

***10.2.2.8. Hộp ComboBox***

Đối tượng **ListBox** là một danh sách các hạng mục đơn mà người dùng có thể chọn một mục hoặc nhiều mục từ danh sách. Đây là một số các thuộc tính của đối tượng danh sách cần lưu ý:

**ColumnCount**- Số cột trong **ListBox.**

**ColumnHeads**- Nếu thuộc tính này là True, số cột của danh sách được hiển thị với các heading (phần tiêu đề).

**MultiSelect** - Nếu thuộc tính là True, người dùng có thể chọn nhiều mục trong danh sách.

**RowSource** - Xác định các mục xuất hiện trong danh sách. Trong Excel, nhập một miền hoặc một tên miền.

**Text** -Xác lập hoặc trả về mục được chọn.

*10.2.2.9. Thanh cuộn (Scrollbar)*

Thanh cuộn **Scrollbar** thường được sử dụng để định hướng các cửa sổ, cũng có thể sử dụng chúng để nhập các giá trị giữa một giá trị tối đa và một giá trị tối thiểu được định nghĩa trước. Sử dụng các thanh cuộn **Scrollbar** để tạo ra các thanh cuộn ngang hoặc dọc. Đây là một bảng tóm tắt của các thuộc tính đối tượng **Scrollbar** sử dụng thường xuyên nhất trong mã VBA:

**LargeChange** - Trả về hoặc xác lập một khoảng mà giá trị **Scrollbar** sẽ thay đổi khi người dùng nhấp giữa hộp cuộn *(scrollbox)* và một trong các mũi tên cuộn.

**Max** - Trả về hoặc xác lập giá trị tối đa của **Scrollbar**.

**Min** - Trả về hoặc xác lập giá trị tối thiểu của **Scrollbar**.

**SmallChange** - Trả về hoặc xác lập một khoảng mà giá trị **Scrollbar** sẽ thay đổi khi người dùng nhấp một trong các mũi tên cuộn.

**Value** - Trả về hoặc xác lập giá trị hiện thời của **Scrollbar**.

*10.2.2.10. Nút Spin*

Một nút **Spin** tương tự như một **Scrollbar**, trong đó người dùng có thể nhấp các mũi tên của nút để tăng hoặc giảm một giá trị. Để tạo một thuộc tính Spin, hãy sử dụng công cụ SpinButton tương tự như một **Scrollbar** *(ngoại trừ không có thuộc tính LargeChange)*.

Hầu hết các nút Spin có một **Control TextBox** bên cạnh chúng để cho phép người dùng chọn nhập số trực tiếp hoặc chọn một số bằng cách sử dụng các nút Spin. Bạn phải sử dụng code VBA để bảo đảm rằng các giá trị trong **TextBox** và **Spinner** đồng bộ. Nói cách khác, nếu muốn tăng Spinner, giá trị hiển thị trong TextBox cũng phải tăng, và ngược lại. Để thực hiện điều này phải thêm mã sự kiện cho cả hai điều khiển.

***Ví dụ:***

Giả sử khi có một TextBox có tên là TextBox1 và một nút Spin tên là SpinButton1. Listing 13.1 minh hoạ mã sẽ giữ cho các giá trị của hai control được đồng bộ.

```
Private Sub TextBox1_Change()
    SpinButton1.Value=TextBox1.Value
End Sub
Private Sub SpinButton1_Change()
    TextBox1.Value=SpinButton1.Value
End Sub
```

### *10.2.2.11. Điều khiển TapShip*

Điều khiển **TabStrip** là một cách lý tưởng để cho người dùng một phương thức hiển thị nhiều tập hợp dữ liệu. Ý tưởng cơ bản đằng sau điều khiển **TabStrip** là khi người dùng định hướng từ tab này sang tab khác, các điều khiển nhìn thấy được vẫn còn, và chỉ dữ liệu được hiển thị bên trong. Thuận lợi ở đây mà bạn cần chỉ tạo một tập hợp control đơn cho Form, bạn sử dụng mã lệnh để điều chỉnh nội dung của các điều khiển này. Để tạo một **TabStrip** bằng cách nhấp nút **TabStrip** trong thanh công cụ ToolBox và sau đó nhấp và rê chuột cho đến khi Strip có kích thước và hình dạng mong muốn.

***Dưới đây là một số điểm cần lưu ý:***

- Cách tốt nhất để thiết lập một **TabStrip** là thêm nó dưới dạng một control đầu tiên cho form và sau đó thêm các control khác bên trong **TabStrip**.

- Khi đã sẵn sàng cho các control định nghĩa cho Form, vẽ **TabStrip** và các control, sau đó sử dụng lệnh Send to Back (*mô tả trước đó*) để đưa **TabStrip** xuống đáy của thứ tự Z.

- Ngoài ra cũng có thể hiển thị một số nút thay cho Tab. Để sử dụng dạng này, chọn **TabStrip** và thay đổi thuộc tính **Style** là **fmTabStyleButtons** (*hoặc 1*).

*Ví dụ:*

```
Private Sub UserForm_Initialize()
    CommandButton1.Caption = "Size Image to Tab Area"
    CommandButton1.WordWrap = True
    CommandButton1.AutoSize = True
End Sub
Private Sub CommandButton1_Click()
    Image1.ZOrder (fmFront)
    Image1.Left = TabStrip1.Left + TabStrip1.ClientLeft
    Image1.Top = TabStrip1.Top + TabStrip1.ClientTop
    Image1.Width = TabStrip1.ClientWidth
    Image1.Height = TabStrip1.ClientHeight
End Sub
```

*Bảng 10.1.* Liệt kê tất cả các control có giá trị Value và cung cấp một mô tả về loại dữ liệu trả về

| **Đối tượng** | **Giá trị trả về** |
|---|---|
| CheckBox | True nếu hộp kiểm được kích hoạt; Falsse nếu không được kích hoạt |
| ComboBox | Vị trí của mục được chọn trong danh sách (*1 là mục tiêu đầu tiên*) |
| ListBox | Vị trí của mục được chọn trong danh sách (*1 là mục tiêu đầu tiên*) |
| OptionButton | True, nếu tuỳ chọn được kích hoạt; form nếu nó được huỷ kích hoạt |
| ScrollBar | Một con số giữa các giá trị tối đa và tối thiểu của Scrollbar |
| SpinButton | Một con số giữa giá trị tối đa và tối thiểu của Spinner |
| TabStrip | Một số nguyên đại diện cho Tab hoạt động (*trong đó tab đầu tiên*) |
| TextBox | Giá trị được nhập trong hộp |
| ToggleButton | True nếu nút được nhấn; False khi ngược lại |

## Chương 11

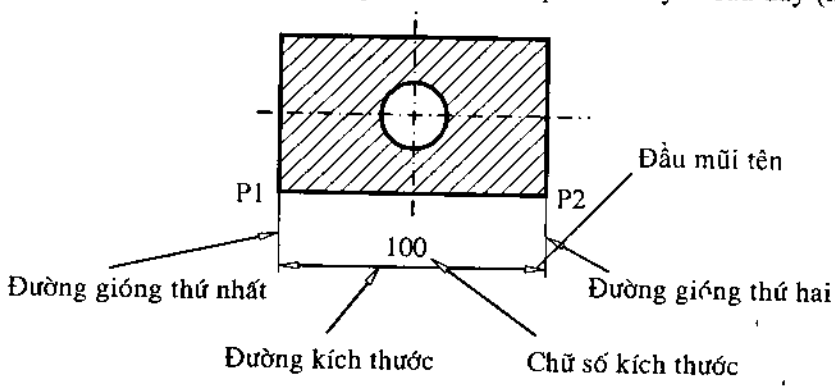
# GHI KÍCH THƯỚC VÀ DUNG SAI

**Chương này trình bày:**

- ***Ghi kích thước***
  - *Kích thước theo phương của hệ trục toạ độ.*
  - *Kích thước theo phương bất kỳ.*
  - *Ghi theo chuỗi kích thước.*
  - *Ghi kích thước đường kính.*
  - *Ghi kích thước bán kính.*
  - *Ghi kích thước góc.*
  - *Đặt đường tâm.*
- ***Ghi dung sai.***
- ***Ghi kích thước theo đường dẫn.***

## 11.1. CÁC THÀNH PHẦN CỦA KÍCH THƯỚC

Một kích thước được ghi bất kỳ bao gồm các thành phần chủ yếu sau đây (hình 11.1).




***Hình 11.1.*** Các thành phần cơ bản của kích thước.

**Đường kích thước**

Đường kích thước được giới hạn hai đầu bởi hai mũi tên *(gạch chéo hoặc một ký hiệu tiêu chuẩn bất kỳ)*. Nếu là kích thước thẳng thì nó cùng phương với đoạn thẳng ghi kích thước, nếu là kích thước góc thì nó là một cung tròn có tâm là đỉnh góc.

**Đường gióng**

Đường gióng là các đoạn thẳng vuông góc với đối tượng được ghi kích thước, thường có hai đường gióng. Tuy nhiên có thể hiệu chỉnh nó thành xiên góc với đường kích thước. Đường gióng được kéo dài quá đường kích thước một đoạn bằng hai đến ba lần chiều rộng kích thước đường cơ bản. Hai đường gióng của cùng một kích thước phải song song với nhau.

**Chữ số kích thước**

Chữ số kích thước là giá trị của đối tượng được ghi kích thước có thể là đơn vị đo khoảng cách hoặc là góc. Trong chữ số kích thước có thể ghi thêm dung sai, ghi tiền tố, hậu tố của kích thước. Chiều cao chữ số kích thước trong các bản vẽ kỹ thuật là các giá trị tiêu chuẩn.

**Mũi tên**

Mũi tên là kí hiệu hai đầu của đường kích thước, thông thường là mũi tên, dấu nghiêng, dấu chấm... hay là một khối bất kỳ do ta tạo nên.

Bảng 11.1 dưới đây là ký hiệu các kiểu mũi tên.

***Bảng 11.1.*** Các kiểu mũi tên theo tiêu chuẩn ISO

| Các kiểu mũi tên thể hiện bằng VBA | Các kiểu mũi tên |
|---|---|
| acArrowDefault | Kiểu mặc định sẵn. |
| AcArrowDot | |
| acArrowDotSmall | |
| acArrowDotBlank | |
| acArrowOrigin | |
| acArrowOrigin2 | |
| acArrowOpen | |
| acArrowOpen90 | |
| acArrowOpen30 | |
| acArrowClosed | |
| acArrowSmall | |
| acArrowNone | |
| acArrowOblique | |
| acArrowBoxFilled | |
| acArrowBoxBlank | |

*Bảng 11.1.* (tiếp theo)

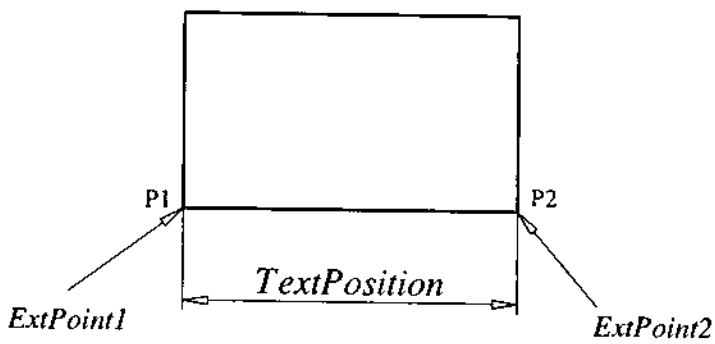
| Các kiểu mũi tên thể hiện bằng VBA | Các kiểu mũi tên |
|---|---|
| acArrowClosedBlank | |
| acArrowDatumFilled | |
| acArrowDatumBlank | |
| acArrowIntegral | |
| acArrowArchTick | |

## 11.2. GHI KÍCH THƯỚC THẲNG

Khi ta muốn ghi kích thước có đường kích thước song song với các trục toạ độ x, y ta dùng lệnh **AddDimAligned**.

**Cú pháp:**

Object = Space.AddDimAligned *(ExtPoint1, ExtPoint2, TextPosition)*




*Hình 11.2.* Minh hoạ cho lệnh AddDimAligned

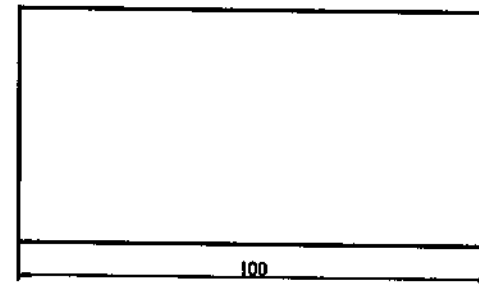
**Trong đó:**

| | |
|---|---|
| *Object* | Là đối tượng đường ghi kích thước thẳng. |
| *Space* | Không gian vẽ hoặc không gian giấy. |
| *ExtPoint1* | Điểm cuối của đường gióng thứ nhất gồm ba phần tử (x, y, z) có kiểu dữ liệu là Double. |
| *ExtPoint2* | Điểm cuối của đường gióng thứ hai gồm ba phần tử (x, y, z) có kiểu dữ liệu là Double. |
| *TextPosition* | Vị trí của đường kích thước và giá trị ghi kích thước gồm ba phần tử (x, y, z) có kiểu dữ liệu Double. |

Đoạn mã chương trình dưới đây vẽ một hình chữ nhật có gốc có toạ độ (0, 0, 0) và đỉnh có toạ độ (100, 50, 0) trong không gian vẽ. Sau đó ghi kích thước của chiều dài hình chữ nhật đó.

```
; Tên file Alined_Demension.dvb.
Sub Alined_Demension()
  Dim rectang As AcadLWPolyline
  Dim points(0 To 9) As Double
  'Định nghĩa hình chữ nhật.
  points(0) = 0: points(1) = 0
  points(2) = 100: points(3) = 0
  points(4) = 100: points(5) = 50
  points(6) = 0: points(7) = 50
  points(8) = 0: points(9) = 0
  'Tạo hình chữ nhật trong không gian vẽ.
  Set rectang = ThisDrawing.ModelSpace.AddLightWeightPolyline(points)
  'Khai báo đối tượng đường ghi kích thước trong VBA.
  Dim Demension As AcadDimAligned
  'Khai báo toạ độ điểm cuối của đường gióng thứ nhất có kiểu dữ liệu Double.
  Dim point1(0 To 2) As Double
  'Khai báo toạ độ điểm cuối của đường gióng thứ hai có kiểu dữ liệu Double.
  Dim point2(0 To 2) As Double
  'Khai báo toạ độ vị trí ghi giá trị kích thước có kiểu dữ liệu  Double.
  Dim location(0 To 2) As Double
  'Định nghĩa các toạ độ để ghi kích thước.
  point1(0) = 0#: point1(1) = 0#: point1(2) = 0#
  point2(0) = 100#: point2(1) = 0#: point2(2) = 0#
  location(0) = 50#: location(1) = -7#: location(2) = 0#
  'Tạo đường ghi kích thước trong không gian vẽ.
  Set Demension = ThisDrawing.ModelSpace.AddDimAligned(point1, point2, location)
  'Gán màu đỏ cho đường ghi kích thước.
  Demension.Color = acRed
  'Thu nhỏ màn hình quan sát.
  ZoomAll
End Sub
; Kết thúc.
```

Sau khi chạy chương trình ta có kết quả như hình 11.3.



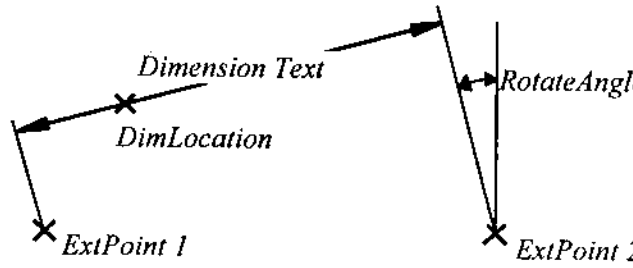

***Hình 11.3.*** Ghi kích thước thẳng.

## 11.3. GHI KÍCH THƯỚC CÓ ĐƯỜNG KÍCH THƯỚC NGHIÊNG VỚI ĐƯỜNG CHUẨN MỘT GÓC XÁC ĐỊNH BẤT KỲ

Lệnh **AddDimRotated** được dùng để ghi kích thước có đường kích thước nghiêng với đường chuẩn một góc xác định bất kỳ.

**Cú pháp:**

Object = Space.AddDimRotated(*ExtPoint1*, *ExtPoint2*, *DimLocation*, *RotateAngle*)

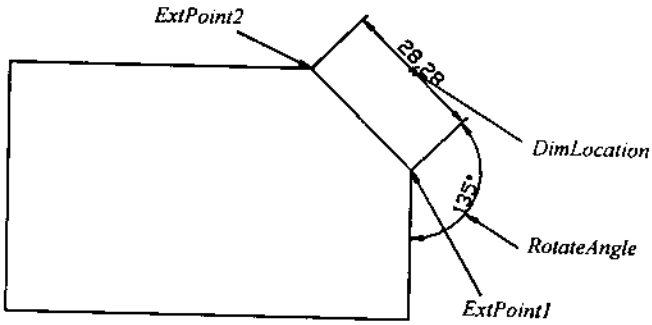



***Hình 11.4.*** Ghi kích thước theo phương nghiêng.

**Trong đó:**

| | |
|---|---|
| *Object* | Đối tượng kích thước nghiêng. |
| *Space* | Không gian vẽ hoặc không gian giấy. |
| *ExtPoint1* | Điểm cuối của đường gióng thứ nhất gồm ba phần tử (x, y, z) có kiểu dữ liệu là Double. |
| *ExtPoint2* | Điểm cuối của đường gióng thứ hai gồm ba phần tử (x, y, z) có kiểu dữ liệu là Double. |
| *DimLocation* | Vị trí của đường kích thước và của giá trị ghi kích thước gồm ba phần tử (x, y, z) có kiểu dữ liệu là Double. |
| *RotateAngle* | Góc quay của đường kích thước *(đơn vị là Radians)* có kiểu dữ liệu là Double. |

Đoạn mã chương trình dưới đây vẽ một hình như hình 11.5 có đường gióng thứ nhất của đường kích thước hợp với phương thẳng đứng một góc 135°.




***Hình 11.5.*** Kích thước không theo phương x, y.

```
; Tên file VBA_AddDimRotated.dvb.
Sub VBA_AddDimRotated()
  Dim plineObj As AcadPolyline
  Dim points(14) As Double
  points(0) = 0: points(1) = 0: points(2) = 0
  points(3) = 80: points(4) = 0: points(5) = 0
  points(6) = 80: points(7) = 30: points(8) = 0
  points(9) = 60: points(10) = 50: points(11) = 0
  points(12) = 0: points(13) = 50: points(14) = 0
  Set plineObj = ThisDrawing.ModelSpace.AddPolyline(points)
  ' Sử dụng thuộc tính đóng kín đối tượng.
  plineObj.Closed = True
  ThisDrawing.Regen (True)
  ' Khai báo đường ghi kích thước nghiêng.
  Dim dimObj As AcadDimRotated
  Dim point1(0 To 2) As Double
  Dim point2(0 To 2) As Double
  Dim location(0 To 2) As Double
  Dim rotAngle As Double
  ' Định nghĩa đường ghi kích thước.
  point1(0) = 80#: point1(1) = 30#: point1(2) = 0#
  point2(0) = 60#: point2(1) = 50#: point2(2) = 0#
  location(0) = 80#: location(1) = 50#: location(2) = 0#
```

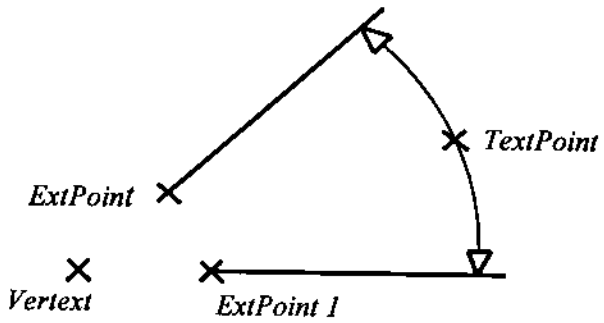
```
' Định nghĩa góc nghiêng so với phương thẳng đứng đơn vị độ.
rotAngle = 135
' Chuyển đổi đơn vị góc nghiêng từ độ sang radians.
rotAngle = rotAngle * 3.141592 / 180#
' Tạo đường kích thước nghiêng trong không gian vẽ.
Set dimObj = ThisDrawing.ModelSpace.AddDimRotated(point1, point2, location, rotAngle)
' Khôi phục lại đối tượng.
DimObj.Update
ZoomAll
End Sub
; Kết thúc.
```

## 11.4. GHI KÍCH THƯỚC GÓC QUA BA ĐIỂM

Lệnh **AddDim3PointAngular** dùng để ghi kích thước góc.

**Cú pháp:**

Object = Space.AddDim3PointAngular(*Vertex*, *EndPoint1*, *EndPoint2*, *TextPoint*)

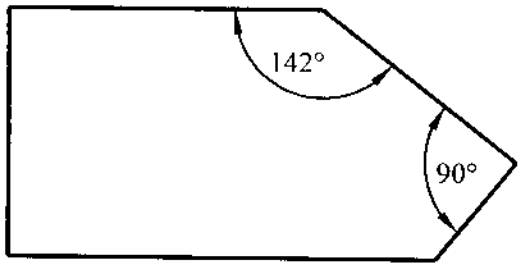



***Hình 11.6.*** Kích thước góc qua ba điểm

**Trong đó:**

| | |
|---|---|
| Object | Đối tượng kích thước góc qua ba điểm. |
| Space | Không gian vẽ hoặc không gian giấy. |
| *Vertex* | Điểm đỉnh của góc gồm ba phần tử (x, y, z) có kiểu dữ liệu Double. |
| *ExtPoint1* | Điểm cuối của cạnh thứ nhất gồm ba phần tử (x, y, z), kiểu dữ liệu Double. |
| *ExtPoint2* | Điểm cuối của cạnh thứ hai gồm ba phần tử (x, y, z), kiểu dữ liệu Double. |
| *TextPoint* | Vị trí của chữ số kích thước gồm ba phần tử (x, y, z), kiểu dữ liệu Double. |

Đoạn mã chương trình dưới đây vẽ một hình và ghi kích thước góc như hình 11.7.




***Hình 11.7.*** Kích thước góc.

```
; Tên file AddDim3PointAngular.dvb.
Sub AddDim3PointAngualr()
 Dim line As AcadLWPolyline
 Dim points(0 To 9) As Double
 ' Định nghĩa đối tượng.
 points(0) = 0: points(1) = 0
 points(2) = 84: points(3) = 0
 points(4) = 100: points(5) = 20
 points(6) = 62: points(7) = 50
 points(8) = 0: points(9) = 50
 ' Tạo một đường light weight Polyline trong không gian vẽ.
 Set line = ThisDrawing.ModelSpace.AddLightWeightPolyline(points)
 ' Sử dụng thuộc tính đóng kín đường.
 line.Closed = True
 ThisDrawing.Regen (True)
 ' Khai báo đối tường đường ghi kích thước góc qua ba điểm.
 Dim DimPointAngular As AcadDim3PointAngular
 Dim Vertex(0 To 2) As Double
 Dim FirstPoint(0 To 2) As Double, SecondPoint(0 To 2) As Double
 Dim TextPoint(0 To 2) As Double
 ' Định nghĩa đối tượng Dim3PointAngular mới.
 Vertex(0) = 100: Vertex(1) = 20: Vertex(2) = 0
 FirstPoint(0) = 84: FirstPoint(1) = 0: FirstPoint(2) = 0
 SecondPoint(0) = 62: SecondPoint(1) = 50: SecondPoint(2) = 0
 TextPoint(0) = 80: TextPoint(1) = 20: TextPoint(2) = 0
 ' Tạo đường ghi kích thước góc Dim3PointAngular trong bản vẽ.
```

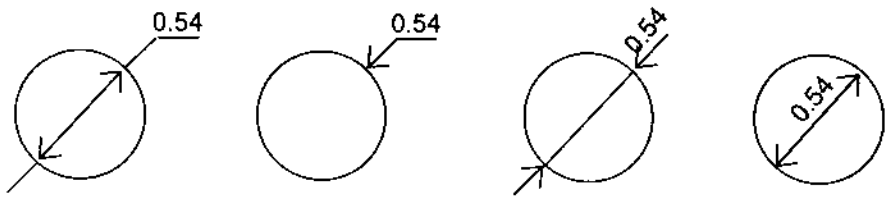
```
    Set DimPointAngular = ThisDrawing.ModelSpace.AddDim3PointAngular(Vertex,
    FirstPoint, SecondPoint, TextPoint)
    DimPointAngular.Color = acGreen
    Vertex(0) = 62: Vertex(1) = 50: Vertex(2) = 0
    FirstPoint(0) = 100: FirstPoint(1) = 20: FirstPoint(2) = 0
    SecondPoint(0) = 0: SecondPoint(1) = 50: SecondPoint(2) = 0
    TextPoint(0) = 60: TextPoint(1) = 35: TextPoint(2) = 0
    Set DimPointAngular = ThisDrawing.ModelSpace.AddDim3PointAngular(Vertex,
    FirstPoint, SecondPoint, TextPoint)
    'Gán màu xanh lá cây cho đường kích thước.
    DimPointAngular.Color = acGreen
    ZoomAll
End Sub
; Kết thúc.
```

## 11.5. GHI KÍCH THƯỚC ĐƯỜNG KÍNH

Lệnh **AddDimDiametric** dùng để ghi kích thước đường kính cho các chi tiết hình trụ hay các lỗ tròn.

**Cú pháp:**

Object = Space.AddDimDiametric(*ChordPoint*, *FarChordPoint*, *LdrLength*)

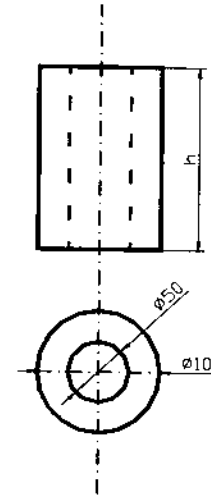



***Hình 10.8.*** Kích thước đường kính.

**Trong đó:**

| | |
|---|---|
| Object | Đối tượng đường ghi kích thước đường kính. |
| Space | Không gian vẽ hoặc không gian giấy. |
| *ChordPoint* | Điểm đầu tiên trên đường tròn và là điểm đầu tiên của đường dẫn kích thước gồm ba phần tử (x, y, z) có kiểu dữ liệu Double. |
| *FarChordPoint* | Điểm thứ hai trên đường tròn có kiểu dữ liệu Double. |
| *LdrLength* | Chiều dài đường dẫn kích thước có kiểu dữ liệu Double. |

Đoạn mã chương trình dưới đây vẽ hình chiếu đứng và hình chiếu bằng của ống trụ và ghi kích thước đường kính trong và đường kính ngoài của ống trụ như hình 11.9.




***Hình 11.9.*** Ghi kích thước đường kính.

```
; Tên file AddDimDiameter.dvb.
'Khai báo biến tâm và đường kính hình trụ.
Public tam_tru(0 To 2) As Double
Public Diametric As Double
'Thủ tục vẽ các hình chiếu đứng, chiếu cạnh hình trụ và ghi kích thước đường kính.
Sub AddDimDiametric()
  tam_tru(0) = 0: tam_tru(1) = 0: tam_tru(2) = 0
   ' Định nghĩa đường kính hình trụ.
  Diametric = 100
  ' Gọi thủ tục vẽ hình chiếu bằng.
  Call chieu_bang
  'Gọi thủ tục vẽ hình chiếu đứng.
  Call chieu_dung
  'Khai báo hàm ghi kích thước đường kính.
  Dim dimDiametric As AcadDimDiametric
  Dim chordPoint(0 To 2) As Double
  Dim farChordPoint(0 To 2) As Double
  Dim leaderLength As Double
  ' Định nghĩa đường ghi kích thước.
  chordPoint(0) = Diametric / 2: chordPoint(1) = 0#: chordPoint(2) = 0#
  farChordPoint(0) = -Diametric / 2: farChordPoint(1) = 0#: farChordPoint(2) = 0#
  leaderLength = 20#
  ' Tạo đường ghi kích thước đường kính trong bản vẽ.
```

```
  Set dimDiametric = ThisDrawing.ModelSpace.AddDimDiametric(chordPoint,
  farChordPoint, leaderLength)
  ZoomAll
End Sub
'Đây là thủ tục lựa chọn kiểu đường trong VBA.
Public Sub Linetype(name As String)
  Dim entry As AcadLineType
  Dim found As Boolean
  found = False
  For Each entry In ThisDrawing.Linetypes
    If StrComp(entry.name, name, 1) = 0 Then
      found = True
      Exit For
    End If
  Next
  If Not (found) Then ThisDrawing.Linetypes.Load name, "acad.lin"
End Sub
'Thủ tục vẽ đường tròn.
Public Sub duong_tron(center() As Double, dd As Double)
  Dim Cir As AcadCircle
  Dim Center3D(2) As Double
  Center3D(0) = center(0)
  Center3D(1) = center(1)
  Center3D(2) = 0
  Dim R As Double
  R = dd / 2
  Set Cir = ThisDrawing.ModelSpace.AddCircle(Center3D, R)
End Sub
Public Sub chieu_bang()
 Call duong_tron(tam_tru, Diametric)
 Call duong_tron(tam_tru, Diametric / 2)
End Sub
Public Sub chieu_dung()
  Dim R As Double, a As Double, h As Double
  R = Diametric / 2
```

```
  a = R
  h = 150
  Dim plineObj As AcadLWPolyline
  Dim points(0 To 7) As Double
  ' Định nghĩa các điểm.
  points(0) = R: points(1) = R + a
  points(2) = R: points(3) = R + a + h
  points(4) = -R: points(5) = R + a + h
  points(6) = -R: points(7) = R + a
  Set plineObj = ThisDrawing.ModelSpace.AddLightWeightPolyline(points)
  plineObj.Closed = True
  ThisDrawing.Regen (True)
  Dim line As AcadLine
  Dim Point1(0 To 2) As Double
  Dim Point2(0 To 2) As Double
  ' Định nghĩa điểm đầu và điểm cuối của đường thẳng.
  Point1(0) = R / 2: Point1(1) = R + a: Point1(2) = 0
  Point2(0) = R / 2: Point2(1) = R + a + h: Point2(2) = 0
  Set line = ThisDrawing.ModelSpace.AddLine(Point1, Point2)
  Call Linetype("ACAD_ISO03W100")
  line.Linetype = "ACAD_ISO03W100"
  Point1(0) = -R / 2: Point1(1) = R + a: Point1(2) = 0
  Point2(0) = -R / 2: Point2(1) = R + a + h: Point2(2) = 0
  Set line = ThisDrawing.ModelSpace.AddLine(Point1, Point2)
  Call Linetype("ACAD_ISO03W100")
  line.Linetype = "ACAD_ISO03W100"
  Point1(0) = 0: Point1(1) = -2 * R: Point1(2) = 0
  Point2(0) = 0: Point2(1) = 2 * R + a + h: Point2(2) = 0
  Set line = ThisDrawing.ModelSpace.AddLine(Point1, Point2)
  'Gọi hàm lựa chọn kiểu đường.
  Call Linetype("center")
  line.Linetype = "center"
  line.Color = acBlue
End Sub
; Kết thúc.
```

## 11.6. GHI KÍCH THƯỚC BÁN KÍNH

Lệnh **AddDimRadial** dùng để ghi kích thước bán kính các cung tròn, góc lượn.

**Cú pháp:**

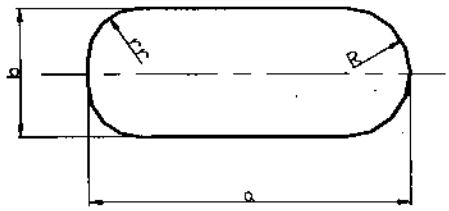
Object = Space.AddDimRadial(*Center*, *ChordPoint*, *LdrLength*)

***Hình 11.10.*** Các kiểu ghi kích thước bán kính.

**Trong đó:**

| | |
|---|---|
| Object | Đối tượng đường ghi kích thước bán kính. |
| Space | Không gian vẽ hoặc không gian giấy. |
| *Center* | Tâm của cung tròn, đường tròn cần ghi kích thước gồm ba phần tử (x, y, z), kiểu dữ liệu Double. |
| *ChordPoint* | Điểm đầu tiên của đường dẫn kích thước gồm ba phần tử (x, y, z), kiểu dữ liệu Double. |
| *LdrLengt* | Chiều dài của đường dẫn kích thước, kiểu dữ liệu Double. |

Đoạn mã chương trình dưới đây viết một thủ tục ghi kích thước bán kính, sau đó áp dụng ghi kích thước bán kính cho một hình vẽ cụ thể trong không gian vẽ hình 11.11.




***Hình 11.11.*** Ghi kích thước bán kính.

```
; Tên file VBA_AddDimRadial.dvb.
Public center(0 To 2) As Double
Public chordPoint(0 To 2) As Double
Public ldrLengt As Integer
Public rr As Double, R As Double
Public a As Double, b As Double
Public tam(2) As Double
Public dimObj As AcadDimRadial
Public line As AcadLine
```

```
Public Sub AddDimRadial(center() As Double, chordPoint() As Double, leaderLen
As Integer)
Set dimObj = ThisDrawing.ModelSpace.AddDimRadial(center, chordPoint,
leaderLen)
End Sub
Sub Main_AddDimRadial()
  Dim Point1(2) As Double, Point2(2) As Double
  Point1(0) = tam(0): Point1(1) = tam(1): Point1(2) = tam(2)
  Point2(0) = tam(0): Point2(1) = tam(1) + (b - rr) / 2: Point1(2) = tam(2)
  Call AddLine(Point1(), Point2())
  Call AddArc(centerArc(), rr, StartAngle, EndPoint)
  Point1(0) = tam(0) + rr: Point1(1) = tam(1) + (b - rr) / 2: Point1(2) = tam(2)
  Point2(0) = tam(0) + (a - R): Point2(1) = tam(1) + (b - rr) / 2: Point1(2) = tam(2)
  Call AddLine(Point1(), Point2())
  Call AddArc(centerArc(), rr, StartAngle, EndPoint)
  Call Linetype("center")
  line.Linetype = "center"
  Point1(0) = tam(0): Point1(1) = tam(1): Point1(2) = tam(2)
  Point2(0) = tam(0) + a: Point2(1) = tam(1): Point1(2) = tam(2)
  Call AddLine(Point1(), Point2())
  Call Mirror(Point1(), Point2())
  center(0) = tam(0) + a - R: center(1) = tam(1): center(2) = tam(2)
  chordPoint(0) = tam(0) + a: chordPoint(1) = tam(1): chordPoint(2) = tam(2):
  Call AddDimRadial(center, chordPoint, leaderLen)
End Sub
; Kết thúc.
```

## 11.7. GHI TOẠ ĐỘ ĐIỂM

Lệnh **AddDimOrdinate** là cách ghi toạ độ của điểm.

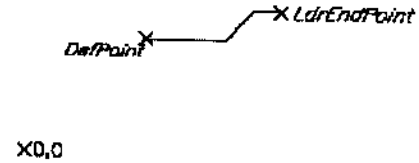
**Cú pháp:**

Object = Space.AddDimOrdinate(*DefPoint*, *LdrEndPoint*, *X_Axis*)

**Trong đó:**

| | |
|---|---|
| Object | Đối tượng toạ độ điểm. |
| Space | Không gian vẽ hoặc không gian giấy. |
| *DefPoint* | Điểm cần ghi kích thước gồm ba phần tử (x, y, z), kiểu dữ liệu Double. |
| *LdrEndPoin* | Điểm cần ghi kích thước gồm ba phần tử (x, y, z), kiểu dữ liệu Double. |
| *X_Axis* | Là một cờ có kiểu dữ liệu Boolean, nếu nó có giá trị True thì hiện giá trị kích thước là hoành độ, còn False thì hiện giá trị kích thước là tung độ. |

Đoạn mã chương trình dưới đây ghi toạ độ điểm trong không gian vẽ sử dụng một điểm cho trước để ghi kích thước, một điểm trên đường dẫn và giá trị hiển thị theo X hoặc Y do người sử dụng như hình 11.12.




***Hình 11.12.*** Ghi toạ độ điểm.

```
; Tên file VBA_AddDimOrdinate.dvb.
Sub VBA_AddDimOrdinate()
  'Khai báo đối tượng có kiểu dữ liệu AcadDimOrdinate trong VBA
  Dim objDim As AcadDimOrdinate
  'Khai báo các biến đầu vào có kiểu dữ liệu Variant
  Dim varDefPt As Variant
  Dim varLdrPt As Variant
  Dim strKeyWord As String
  Dim blnXaxis As Boolean
  'Nhập điểm cần ghi kích thước từ bàn phím.
  varDefPt = ThisDrawing.Utility.GetPoint(, vbCrLf & "Feature location: ")
   'Nhập điểm đầu tiên của đường dẫn.
   varLdrPt = ThisDrawing.Utility.GetPoint(varDefPt, vbCrLf & "Leader endpoint: ")
ThisDrawing.Utility.InitializeUserInput 1, "X Y"
strKeyWord = ThisDrawing.Utility.GetKeyword _ ("X axis or Y axis? [X/Y]: ")
   'Kiểm tra điều kiện sử dụng cấu trúc if ...then
   If strKeyWord = "X" Then
     blnXaxis = True
     Else
   blnXaxis = False
  End If
  'Ghi toạ độ một điểm.
  Set objDim = ThisDrawing.ModelSpace.AddDimOrdinate(varDefPt, varLdrPt, _
  objDim.Update
End Sub
; Kết thúc.
```

## 11.8. ĐẶT MÀU CHO CÁC ĐƯỜNG KÍCH THƯỚC

### 11.8.1. Đặt màu cho đường gióng

Để đặt màu cho các đường gióng kích thước bạn sử dụng thuộc tính **ExtensionLineColor**.

**Cú pháp:**

Object.ExtensionLineColor

**Trong đó:**

| | |
|---|---|
| Object | Kích thước thẳng, đường kính, bán kính, kích thước nghiêng v.v... |
| ExtensionLineColor | Giá trị màu. |

*Ví dụ:*

```
Dim dimObj As AcadDimAligned
dimObj.ExtensionLineColor = acRed
```

### 11.8.2. Đặt màu cho đường kích thước

**Cú pháp:**

Object.Color

**Trong đó:**

| | |
|---|---|
| Object | Kích thước thẳng, đường kính, bán kính, kích thước nghiêng v.v... |
| Color | Cho kích thước, giá trị màu đã được đặt trước trong chương trình. |

## 11.9. ĐẶT ĐƠN VỊ

### 11.9.1. Định dạng đơn vị

**Cú pháp:**

Object.UnitsFormat

**Trong đó:**

| | |
|---|---|
| Object | Kích thước thẳng, đường kính, bán kính, góc nghiêng v.v... |
| UnitsFormat | Các kiểu đơn vị ghi kích thước.<br>Giả sử cần ghi kích thước 1 đoạn thẳng có độ dài là ***10.12345678*** thì tương ứng với các dạng đơn vị như bảng 11.2. |

*Bảng 11.2.* Các kiểu ghi đơn vị kích thước

| Kiểu đơn vị kích thước được ghi | Hình minh hoạ |
|---|---|
| acDimLScientific | 1.0123E+01 |
| acDimLDecimal | 10.1235 |
| acDimLEngineering | 10.1235″ |
| acDimLArchitectural | $10\frac{1}{8}''$ |
| acDimLFractional | $10\frac{1}{8}$ |
| acDimLWindowsDesktop | 10.1235 |

*Ví dụ:*

```
Dim dimObj As AcadDimAligned
dimObj.UnitsFormat = acDimLEngineering
```

### 11.9.2. Độ chính xác của kích thước

Để đặt độ chính xác của kích thước ta sử dụng thuộc tính **PrimaryUnitsPrecision**.

**Cú pháp**:

Object.PrimaryUnitsPrecision

**Trong đó**:

| Object | Kích thước thẳng, đường kính, bán kính, góc nghiêng v.v. |
|---|---|
| PrimaryUnitsPrecision | Độ chính xác của kích thước tương ứng.<br>acDimPrecisionZero 0.<br>acDimPrecisionOne 0.0<br>acDimPrecisionTwo 0.00<br>acDimPrecisionThree 0.000<br>acDimPrecisionFour 0.0000<br>acDimPrecisionFive 0.00000<br>acDimPrecisionSix 0.000000<br>acDimPrecisionSeven 0.0000000<br>acDimPrecisionEight 0.00000000 |

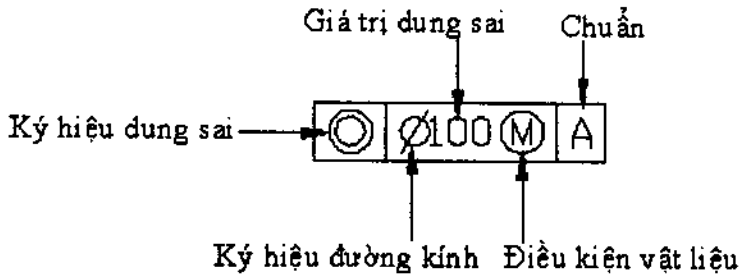
*Ví dụ:*

Dim dimObj As AcadDimAligned *'Khai báo đối tượng kích thước thẳng.*

dimObj.PrimaryUnitsPrecision = acDimPrecisionZero

*'Thiết lập độ chính xác của kích thước có dạng acDimPrecisionZero (không có giá trị nào sau dấu phẩy).*

## 11.10. GHI DUNG SAI




***Hình 11.13.*** Các thành phần dung sai.

**Ghi dung sai**

Lệnh **AddTolerance** dùng để ghi dung sai hình dạng và vị trí các bề mặt trên bản vẽ.

**Cú pháp:**

Object = space.AddTolerance(*Text, InsertionPoint, Direction*)

**Trong đó:**

| | |
|---|---|
| Object | Đối tượng dung sai. |
| space | Không gian vẽ hoặc không gian giấy. |
| Text | Ghi giá trị dung sai có kiểu dữ liệu String. |
| *InsertionPoint* | Điểm chèn các kí tự dung sai gồm ba phần tử (x, y, z) có kiểu dữ liệu là Double. |
| *Direction* | Phương hướng của các ký tự dung sai gồm ba phần tử (x, y, z) có kiểu dữ liệu là Double. |

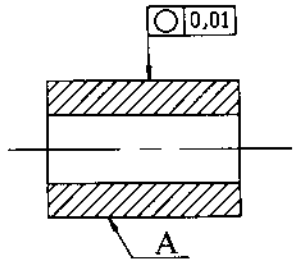
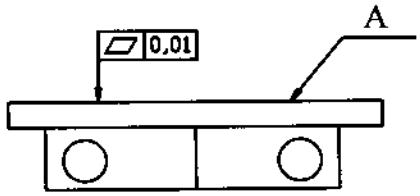
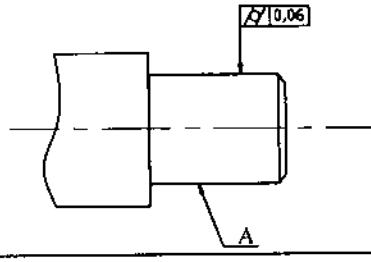
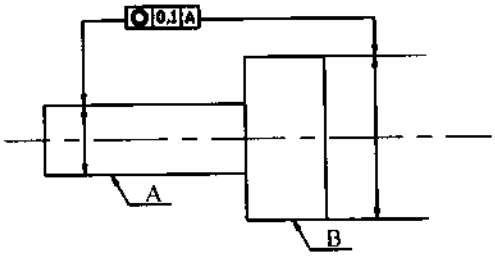
Đoạn mã chương trình dưới đây tạo một thủ tục dùng để ghi dung sai của các chi tiết, thủ tục này được khai báo ở dạng **Public**. Do vậy có thể sử dụng thủ tục này để đưa các thông số đầu vào và gọi thủ tục này, làm cho chương trình của bạn trở lên rõ ràng, sáng sủa và dễ quản lý. Sau đây áp dụng cho một ví dụ cụ thể hình 11.14.

```
; Tên file VBA_tolerance.dvb.
'Khai báo đối tượng ghi dung sai trong VBA.
Public toleranceObj As AcadTolerance
'Khai báo các thông số đầu vào của thủ tục.
Public textString As String
Public insertionPoint(0 To 2) As Double
Public direction(0 To 2) As Double
Public Sub vba_tolerance(textString As String, insertionPoint() As Double,
 direction() As Double)
    Dim text As String
    Dim insertPoint(0 To 2) As Double
    Dim direc(0 To 2) As Double
    'Định nghĩa các đối tượng.
    text = textString
    insertPoint(0) = insertionPoint(0):
    insertPoint(1) = insertionPoint(1):
    insertPoint(2) = insertionPoint(2):
    direc(0) = direction(0)
    direc(1) = direction(1)
    direc(2) = direction(2)
    'Tạo dung sai trong không gian vẽ.
    Set toleranceObj = ThisDrawing.ModelSpace.AddTolerance(text, insertPoint, direc)
End Sub
'Chương trình chính ghi dung sai, gọi thủ tục ghi dung sai ở trên.
Sub Main_AddTolerance()
    ' Định nghĩa các thông số đầu vào.
    textString = "{\Fgdt;r}%%vasdf{\Fgdt;l}%%vdf%%vxc%%v12{\Fgdt;m}%%vsd"
    insertionPoint(0) = 5#: insertionPoint(1) = 5#: insertionPoint(2) = 0#
    direction(0) = 100#: direction(1) = 0#: direction(2) = 0#
    ' Gọi thủ tục ghi dung sai áp dụng với các thông số đầu vào.
    Call vba_tolerance(textString, insertionPoint, direction)
    ZoomAll
End Sub
; Kết thúc.
```

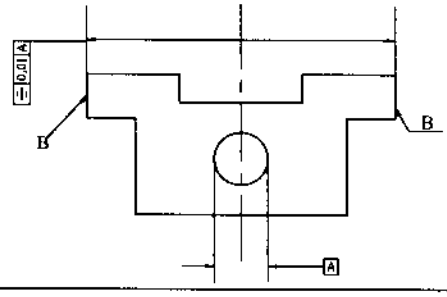
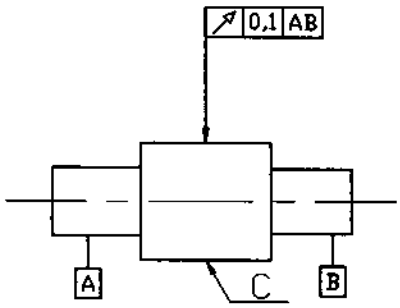
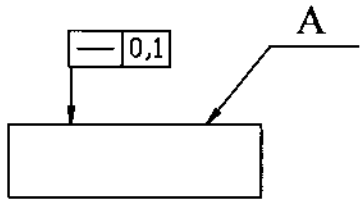
| | | | | | |
|---|---|---|---|---|---|
| ◎ | asdf Ⓛ | df | xc | 12 Ⓜ | sd |

*Hình 11.14.* Ghi dung sai.

*Bảng 11.3.* Một số ví dụ kí hiệu dung sai hình dạng và vị trí bề mặt thường dùng trong bản vẽ kỹ thuật

| Kí hiệu | Yêu cầu kĩ thuật |
|---|---|
|  | Dung sai độ tròn của bề mặt A là 0,01 mm<br>Mã VBA<br>textString = "{\Fgdt;e}%%v0,1%%vA" |
|  | Yêu cầu kĩ thuật<br>Dung sai độ phẳng của bề mặt A là 0,01 mm<br>Mã VBA<br>textString = "{\Fgdt;c}%%v0,1%%vA" |
|  | Dung sai độ trụ của bề mặt A là 0,06 mm<br>Mã VBA<br>textString = "{\Fgdt;g}%%v0,06" |
|  | Yêu cầu kĩ thuật<br>Dung sai độ đồng tâm của các bề mặt A và B là 0,1 mm<br>Mã VBA<br>textString = "{\Fgdt;g}%%v0,1%%vA" |

*Bảng 11.3.* (tiếp theo)

| Kí hiệu | Yêu cầu kĩ thuật |
|---|---|
|  | Yêu cầu kĩ thuật<br>Dung sai độ đối xứng của bề mặt B so với đường tâm lỗ A là 0,01 mm<br>Mã VBA<br>textString = "{\Fgdt;i}%%v0,01%%vA" |
|  | Yêu cầu kĩ thuật<br>Dung sai độ đảo hướng của bề mặt C so với đường tâm chung của 2 mặt A, B là 0,1 mm<br>Mã VBA<br>textString = "{\Fgdt;h}%%v0,1%%vAB" |
|  | Yêu cầu kĩ thuật<br>Dung sai độ thẳng của bề mặt A là 0,1 mm trên toàn bộ bề mặt<br>Mã VBA<br>textString = "{\Fgdt;c}%%v0,1%%vA" |

***Chú ý:*** *Với thủ tục ghi dung sai ở trên có thể đặt ở bất kì chỗ nào trong chương trình (đóng gói), khi cần dùng tới chỉ cần gọi thủ tục đó mà không cần phải mất công xây dựng lại. Các ký hiệu của mã ghi dung sai tham khảo bảng 11.3.*

## 11.11. ĐẶT MÀU CHO KIỂU CHỮ CỦA ĐỐI TƯỢNG KÍCH THƯỚC VÀ DUNG SAI

Để đặt màu cho giá trị kích thước và dung sai sử dụng thuộc tính **TextColor**.

**Cú pháp:**

Object.TextColor

**Trong đó:**

| | |
|---|---|
| Object | Các kiểu ghi kích thước và dung sai. |
| TextColor | Thuộc tính màu đã trình bày ở chương trước. |

*Ví dụ:*

```
' Khai báo đối tượng kích thước thẳng.
Dim dimObj As AcadDimAligned
  ' Đặt kiểu màu cho kiểu chữ kích thước.
dimObj.TextColor = acRed
```

## 11.12. KHOẢNG CÁCH GIỮA GIÁ TRỊ KÍCH THƯỚC VÀ ĐƯỜNG KÍCH THƯỚC

Trong nhiều trường hợp muốn giá trị ghi kích thước ra xa hoặc gần đường kích thước sử dụng thuộc tính **TextGap**.

**Cú pháp:**

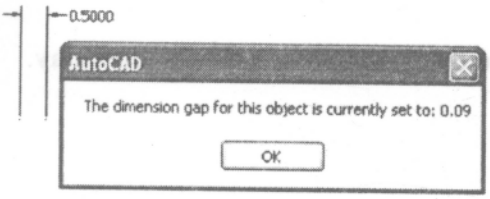
Object.TextGap

**Trong đó:**

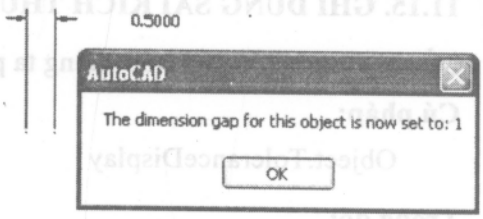
| | |
|---|---|
| Object | Các kiểu ghi kích thước, đường dẫn và dung sai. |
| TextGap | Giá trị khoảng cách có kiểu dữ liệu là Double. |

Đoạn mã chương trình này sẽ ghi kích thước khoảng cách hai điểm trong không gian vẽ, sử dụng thuộc tính TextGap.

```
; Tên file vba_TextGap.dvb.
Sub vba_TextGap()
  Dim dimObj As AcadDimAligned
  Dim point1(0 To 2) As Double, point2(0 To 2) As Double
  Dim location(0 To 2) As Double
  'Định nghĩa các điểm ghi kích thước.
  point1(0) = 5: point1(1) = 5: point1(2) = 0
  point2(0) = 5.5: point2(1) = 5: point2(2) = 0
  location(0) = 5: location(1) = 7: location(2) = 0
  Set dimObj = ThisDrawing.ModelSpace.AddDimAligned(point1, point2, location)
  ThisDrawing.Application.ZoomAll
  ' Thông báo giá trị của khe hở kích thước hiện thời.
  MsgBox "The dimension gap for this object is currently set to: "& dimObj.TextGap
  ' Tăng khe hở kích thước lên 1.
  dimObj.TextGap = 1
  ThisDrawing.Regen acAllViewports
  ' Thông báo giá trị của khe hở kích thước sau khi tăng.
  MsgBox "The dimension gap for this object is now set to: " & dimObj.TextGap
End Sub
; Kết thúc.
```




*Hình 11.15.* Khoảng cách kích thước ban đầu.




*Hình 11.16.* Khoảng cách kích thước sau khi thiết lập.

## 11.13. CHIỀU CAO CHỮ SỐ KÍCH THƯỚC VÀ DUNG SAI

Để thay đổi chiều cao của chữ số ghi kích thước và dung sai sử dụng thuộc tính **TextHeight**.

**Cú pháp:**

Object.TextHeight

**Trong đó:**

| | |
|---|---|
| Object | Các kiểu ghi kích thước, đường dẫn và dung sai. |
| TextHeight | Chiều cao chữ có kiểu dữ liệu Double. |

*Ví dụ:*

```
Dim dimObj As AcadDimAligned        ' Khai báo đối tượng kích thước thẳng.
dimObj.TextHeight = 10              ' Đặt chiều cao cho kiểu chữ kích thước bằng 10.
```

## 11.14. GIÁ TRỊ KÍCH THƯỚC Ở BÊN TRONG ĐƯỜNG GIÓNG

Để giá trị ghi kích thước ở bên trong đường gióng sử dụng thuộc tính **TextInside**.

**Cú pháp:**

Object.TextInside

**Trong đó:**

| | |
|---|---|
| Object | Các kiểu ghi kích thước. |
| TextInside | Có kiểu dữ liệu Boolean<br>+ **True**: giá trị ghi kích thước ở bên trong đường gióng.<br>+ **False**: giá trị ghi kích thước ở bên trong đường gióng trong trường hợp còn không gian, nếu không giá trị đó sẽ ở bên ngoài đường gióng. |

*Ví dụ:*

```
Dim dimObj As AcadDimAligned        ' Khai báo đối tượng kích thước thẳng.
dimObj.TextInside = True            ' Giá trị kích thước sẽ ở bên trong đường gióng.
```

## 11.15. GHI DUNG SAI KÍCH THƯỚC

Để ghi dung sai kích thước chúng ta phải sử dụng thuộc tính **ToleranceDisplay**.

**Cú pháp:**

Object.ToleranceDisplay

**Trong đó:**

| | |
|---|---|
| Object | Các kiểu ghi kích thước. |
| ToleranceDisplay | Các cách thể hiện dung sai như bảng 11.4. |

***Bảng 11.4.*** Kiểu dung sai kích thước

| **Kiểu dung sai kích thước** | **Hình minh hoạ** |
|---|---|
| acTolNone | 0,5 |
| acTolSymmetrical | 0,5±0 |
| acTolDeviation | 0<br>0,5-0 |
| acTolLimits | 0,5<br>0,5 |
| acTolBasic | 0,5 |

## 11.16. VỊ TRÍ GIÁ TRỊ DUNG SAI CỦA KÍCH THƯỚC

Để thể hiện vị trí của giá trị dung sai sử dụng thuộc tính **ToleranceJustification**.

**Cú pháp:**

Object.ToleranceJustification

**Trong đó:**

| | |
|---|---|
| Object | Các kiểu ghi kích thước. |
| ToleranceJustification | Vị trí ghi giá trị dung sai bảng 11.5. |

*Bảng 11.5.* Vị trí dung sai

| Kiểu xác định vị trí | Hình minh hoạ |
|---|---|
| acTolTop (đỉnh) | 0,5 |
| acTolMiddle (giữa) | 0,5 |
| acTolBottom (dưới) | 0,5 |

## 11.17. GHI SAI LỆCH TRÊN DƯỚI DUNG SAI KÍCH THƯỚC

Để thể hiện sai lệch của giá trị dung sai sử dụng thuộc tính **ToleranceLowerLimit, ToleranceUpperLimit**.

**Cú pháp:**

Object.ToleranceLowerLimit

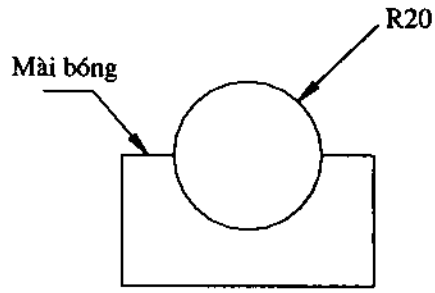
Object.ToleranceUpperLimit

**Trong đó:**

| Object | Các kiểu ghi kích thước. |
|---|---|
| ToleranceLowerLimit | Sai lệnh dưới. |
| ToleranceUpperLimit. | Sai lệnh trên. |

## 11.18. ĐÁNH SỐ CHI TIẾT VÀ GHI CHỈ DẪN

Lệnh **AddLeader** dùng để ghi kích thước liên kết có đường dẫn hình 11.17.




*Hình 11.17.* Ghi kích thước theo đường dẫn.

**Cú pháp:**

Object = Space.AddLeader(PointsArray, Annotation, Type)

**Trong đó:**

| | |
|---|---|
| Object | Đối tượng đường dẫn. |
| Space | Không gian vẽ hoặc không gian giấy. |
| PointsArray | Mảng điểm ghi đường dẫn. |
| Annotation | Giá trị dung sai (*kiểu chữ hoặc để chống*). |
| Type | **Các kiểu đường dẫn**<br>AcLineNoArrow (*không có mũi tên*)<br>AcLineWithArrow (*có mũi tên*)<br>AcSplineNoArrow (*kiểu đường cong không có mũi tên*)<br>AcSplineWithArrow (*kiểu đường cong có có mũi tên*) |

Đoạn mã chương trình dưới đây tạo một đường dẫn trong không gian vẽ.

```
; Tên file VBA_AddLeader.dvb.
Sub AddLeader()
   Dim leaderObj As AcadLeader
   Dim points(0 To 8) As Double
   Dim leaderType As Integer
   Dim annotationObject As AcadObject
   ' Định nghĩa đường dẫn
   points(0) = 0: points(1) = 0: points(2) = 0
   points(3) = 4: points(4) = 4: points(5) = 0
   points(6) = 4: points(7) = 5: points(8) = 0
   leaderType = acSplineWithArrow
   Set annotationObject = Nothing
   'Tạo đường dẫn trong không gian vẽ
   Set leaderObj = ThisDrawing.ModelSpace.AddLeader(points, annotationObject,
   leaderType)
End Sub
; Kết thúc.
```

## Chương 12
# LÀM VIỆC VỚI ĐỐI TƯỢNG BLOCK

Chương này trình bày các nội dung sau:

- *Định nghĩa một Block*
- *Khởi tạo một Block*
- *Đổi tên một Block*
- *Xóa một Block*
- *Chèn Block*
- *Phá bỏ Block*
- *Xoá một Block chèn*
- *Ghi một Block*
- *Tạo các thuộc tính của Block*
- *Chèn các thuộc tính của Block*
- *Gắn bản vẽ tham khảo ngoài*
- *Loại bỏ việc tham khảo bản vẽ ngoài*
- *Chuyển bản vẽ tham khảo ngoài thành một Block của bản vẽ hiện thời*

## 12.1. ĐỊNH NGHĨA BLOCK

**Block** là một đối tượng trong ACAD và được tạo từ một hoặc nhiều đối tượng cơ bản trong bản vẽ ACAD. Các đối tượng trong một **Block** được coi như các đối tượng đơn. Các khối **Block** cho phép bạn có thể sử dụng trong cùng một bản vẽ hoặc các bản vẽ khác nhau, **Block** được chèn vào trong bản vẽ tại một vị trí bất kỳ với hệ số tỷ lệ theo phương x, y và quay quanh điểm chèn một góc xác định.

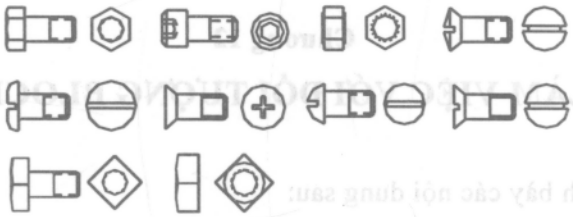
Sử dụng **Block** giúp ta tạo thư viện bản vẽ, các ký hiệu và thông thường được sử dụng trong các bản vẽ lắp.

***Thuận lợi khi sử dụng Block:***

- Dùng lại nội dung của bản vẽ.
- Khởi tạo một thư viện biểu tượng.
- Sử dụng các chú thích và các thông tin của thuộc tính.
- Thu nhỏ kích cỡ của bản vẽ.

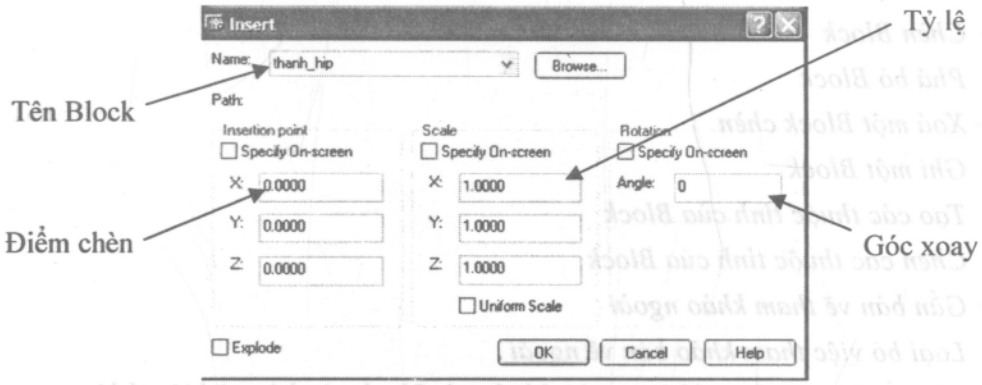
Trong ACAD khi cài đặt sẽ có thư viện biểu tượng trong thư mục: \Sample\DesignCenter.

**DesignCenter** là một công cụ vạn năng của ACAD. Sử dụng **DesignCenter**, bạn có thể quan sát các khối trước khi muốn sử dụng chúng.



***Hình 12.1.*** Một số hình tạo Block khi vẽ các chi tiết máy.

Trong bản vẽ ACAD hộp thoại chèn Block được sử dụng để chèn một Block như hình 12.2 dưới đây.




***Hình 12.2.*** Hộp thoại chèn Block.

***Chú ý:** Tên Block không được dài quá 31 ký tự và giữa chúng không có khoảng trống.*

## 12.2. LIÊN KẾT CÁC KHỐI

Mỗi một khối trong bản vẽ có thể gồm một hoặc nhiều chi tiết hợp lại.

### 12.2.1. Gọi một Block từ thư viện

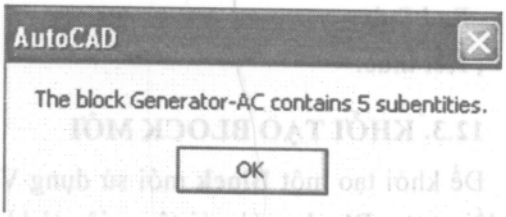
Để gọi một **Block** trong thư viện ta có thể sử dụng các chỉ số của **Block** hoặc gọi tên của **Block** đó.

Đoạn mã chương trình dưới đây sẽ mở thư viện Block Generator-AC và hiện thông báo về đối tượng **Block** đó.

```
; Tên file VBA_GetBlockDef.dvb.
Sub VBA_GetBlockDef()
Dim objBlock As AcadBlock
' Nhảy đến  NOTFOUND nếu như có lỗi xảy ra.
```

```
On Error GoTo NOTFOUND:
' Nhận tên Block "Generator-AC" từ thư viện.
Set objBlock = ThisDrawing.Blocks.Item("Generator - AC")
' Hiển thị các thành phần cấu tạo nên Block đó.
MsgBox "The Block Generator-AC contains " & objBlock.Count & _
" subentities."
Exit Sub
NOTFOUND:
' Thông báo Block "Generator-AC" không tồn tại.
MsgBox "The Block Generator-AC was not found."
End Sub
; Kết thúc.
```

Khi chạy chương trình VBA_GetBlockDef.dvb hiện lên thông báo về các đối tượng trong **Block** như hình 12.3.




***Hình 12.3.***

### 12.2.2. Liệt kê các khối trong bản vẽ

Để liệt kê tất cả các khối trong một bản vẽ khi sử dụng vòng lặp: **For Each... loop to**.

Đoạn chương trình sau sẽ hiển thị tất cả các Blocks được định nghĩa trong bản vẽ hiện hành hình 12.4 minh hoạ.


***Hình 12.4***

```
; Tên file VBA_IterateBlockDefs.dvb
 Sub VBA_IterateBlockDefs()
 Dim strBlocks As String
 Dim objBlockRef As AcadBlock
 ' Mặc định khối ban đầu là rỗng.
 strBlocks = ""
 ' Liệt kê các khối trong bản vẽ.
 For Each objBlockRef In ThisDrawing.Blocks
 strBlocks = strBlocks & objBlockRef.Name & vbCrLf
 Next
 ' Hiển thị kết quả với người sử dụng.
 MsgBox "Blocks defined in drawing: " & vbCrLf & vbCrLf & strBlocks
 End Sub
; Kết thúc.
```

## 12.3. KHỞI TẠO BLOCK MỚI

Để khởi tạo một **Block** mới sử dụng VBA và các đối tượng của ACAD chỉ cần thêm một đối tượng **Block** mới với tên miêu tả khối Block đó để liên kết các **Block**. Một **Block** mới được tạo ra khi đó có thể chèn **Block** đó vào trong bản vẽ ở **MS** hoặc **PS**.

Phương thức **Add** được sử dụng để tạo **Block** với tên gọi cụ thể dùng để liên kết các **Block**.

**Cú pháp:**

ObjBlock = Blocks.Add(*BasePt*, *Name*)

**Trong đó:**

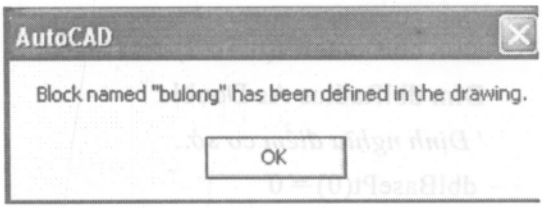
| | |
|---|---|
| ObjBlock | Là đối tượng Block. |
| *BasePt* | Là điểm chèn của khối Block trong bản vẽ gồm ba phần tử (x, y, z) có kiểu dữ liệu Double, |
| *Name* | Tên của khối Block có kiểu dữ liệu là String. |

Đoạn mã chương trình dưới đây chèn một Block có tên là ("bulong") đến bản vẽ hiện hành.

```
;Tên file VBA_AddBlockDef.dvb
 Sub VBA_AddBlockDef()
    ' Khai báo các đối tượng.
   Dim objBlock As AcadBlock
   Dim dblBasePt(0 To 2) As Double
```

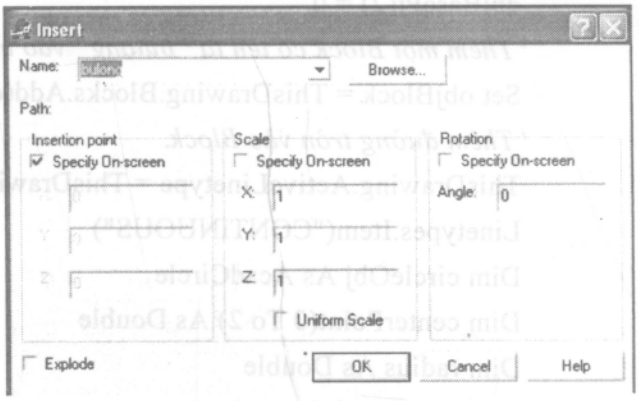
```
    Dim dblCenterPt(0 To 2) As Double
    Dim dblRadius As Double
    ' Định nghĩa điểm cơ sở..
    dblBasePt(0) = 0
    dblBasePt(1) = 0
    dblBasePt(2) = 0
    ' Thêm một Block có tên la "bulong" vào bản vẽ.
    Set objBlock = ThisDrawing.Blocks.Add(dblBasePt, "bulong")
    ' Thêm đường tròn vào Block.
    ThisDrawing.ActiveLinetype = ThisDrawing. _
    Linetypes.Item("CONTINUOUS")
    Dim circleObj As AcadCircle
    Dim centerPoint(0 To 2) As Double
    Dim radius As Double
    ' Định nghĩa đường tròn.
    centerPoint(0) = 0#: centerPoint(1) = 0#: centerPoint(2) = 0#
    radius = 50#
    ' Thêm đường tròn vào Block.
    objBlock.AddCircle centerPoint, radius
    Dim plineObj As AcadLWPolyline
    Dim points(0 To 13) As Double
    points(0) = 96: points(1) = 0
    points(2) = 48: points(3) = 84
    points(4) = -48: points(5) = 84
    points(6) = -96: points(7) = 0
    points(8) = -48: points(9) = -84
    points(10) = 48: points(11) = -84
    points(12) = 96: points(13) = 0
    objBlock.AddLightWeightPolyline (points)
    ThisDrawing.Regen (True)
    ZoomAll
    'Thông báo đã có một Block "bulong trong bản vẽ hiện hành".
    MsgBox "Block named ""bulong"" has been defined in the drawing."
  End Sub
; Kết thúc.
```

+ Khi chạy chương trình VBA_AddBlockDef.dvb, hộp thoại thông báo là khối Block "Bulong" đã được tạo trong bản vẽ.



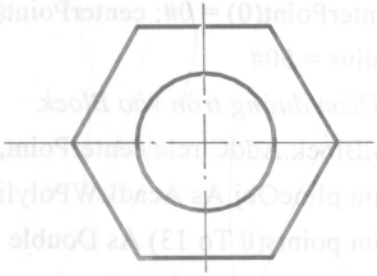

*a.* Hộp thoại thông báo Block đã được khởi tạo

+ Từ thanh công cụ ToolBar/ Insert/Block...xuất hiện hộp thoại chèn Block như hình 12.5. Trong mục Name ta thấy có tên Block "bulong" ta vừa được tạo.




***Hình 12.5.*** Hộp thoại chèn Block.

+ Từ hộp thoại Block ta chọn OK và chọn điểm chèn trên bản vẽ để chèn khối **Block** trên vào. Kết quả khối Block như hình 12.6.



***Hình 12.6.***

***Chú ý:*** *Nhớ rằng MS và PS dùng để thể hiện các* ***Block****. Do vậy mà không thể tạo các Blocks có tên trùng với* ***Modelspace*** *hoặc* ***PaperSpace****.*

## 12.4. ĐỔI TÊN BLOCK

Dựa vào thuộc tính **Name** của **Block** để đặt lại tên của một **Block**. Cách đơn giản nhất để thay đổi tên của **Block** là sử dụng thuộc tính **Name** của nó để đổi sang tên mới.

Đoạn mã chương trình dưới đây sẽ thay đổi tên khối Block ("bulong") thành ("bulong1").

```
; Tên file: BlockRename.dvb
Sub renameBlock()
Dim objBlock As AcadBlock
' Nhảy đến NOTFOUND nếu như lỗi xảy ra.
```

```
On Error GoTo NOTFOUND:
Set objBlock = ThisDrawing.Blocks.Item("bulong")
' Đổi tên khối
objBlock.Name = "bulong1"
' Thông báo với người sử dụng.
MsgBox """bulong"" Block renamed to ""bulong1""."
Exit Sub
NOTFOUND:
MsgBox """bulong"" not found."
End Sub
; Kết thúc.
```

## 12.5. XOÁ MỘT BLOCK

Để xoá một **Block** từ liên kết các **Block** ta sử dụng lệnh **Delete**.

**Cú pháp:**

ObjBlock.Delete

**Trong đó:**

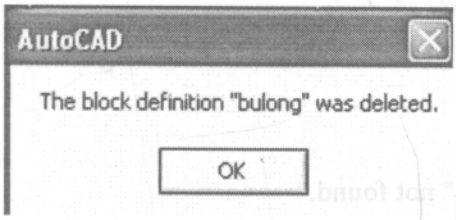
| | |
|---|---|
| ObjBlock | Là đối tượng Block cần xoá. |

Đoạn mã chương trình dưới đây sẽ xoá **Block** có tên là ("bulong") từ bản vẽ hiện hành.

```
; Tên file  DeleteBlock.dvb
Sub VBA_DeleteBlock()
Dim objBlock As AcadBlock
' Tắt lỗi.
On Error Resume Next
' Nhận giá trị của Block trong liên kết.
Set objBlock = ThisDrawing.Blocks.Item("bulong")
' Xoá Block.
objBlock.Delete
' Thông báo cho người sử dụng.
If Err.Number = 0 Then
MsgBox "The Block definition ""bulong"" was deleted."
Else
MsgBox "Cannot delete the Block definition ""bulong""."
```

```
Err.Clear
End If
End Sub
```

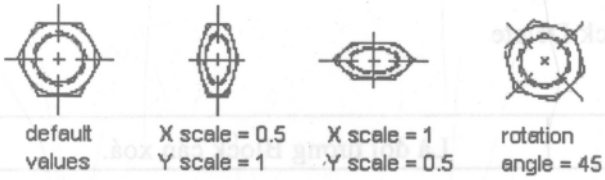
; Kết thúc chương trình hộp thoại thông báo như hình 12.7.




*Hình 12.7.* Xoá Block.

## 12.6. CHÈN BLOCK

Block có thể chèn ở vị trí bất kì, với tỉ lệ các phương X, Y khác nhau và quay chung quanh điểm chèn một góc tùy ý hình 12.8 minh hoạ.




*Hình 12.8.* Block được chèn với tỷ lệ khác nhau.

Phương thức **InsertBlock** dùng để chèn một Block vào trong bản vẽ.

**Cú pháp:**

ObjBlockRef = Space.InsertBlock(*InsertPt*, *Name*, *Xscale*, *Yscale*, *Zscale*, *_Rotation*)

**Trong đó:**

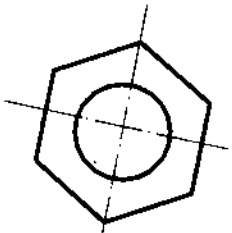
| | |
|---|---|
| ObjBlockRef | Khối chuyển đến |
| Space | Không gian vẽ hoặc không gian giấy |
| *InsertPt* | Là điểm chèn khối gồm có ba phần tử (x, y, z) có kiểu dữ liệu Double |
| *Name* | Tên Block có kiểu dữ liệu String |
| *Xscale* | Hệ số tỉ lệ theo phương X có kiểu dữ liệu Double |
| *Yscale* | Hệ số tỉ lệ theo phương Y có kiểu dữ liệu Double |
| *Zscale* | Hệ số tỉ lệ theo phương Z có kiểu dữ liệu Double |
| *Rotation* | Góc xoay của Bolocks có kiểu dữ liệu Double (*đơn vị: radians*) |

Đoạn mã chương trình dưới đây tạo một Block có tên là ("bulong") sau đó chèn nó ở vị trí có toạ độ (0, 0, 0) với tỉ lệ 1 và có góc xoay là $60^0$.

```
; Tên file: VBA_InsertBlock.dvb.
  Sub Ch11_AddBlockDef()
    Dim objBlock As AcadBlock
    Dim dblBasePt(0 To 2) As Double
    Dim dblCenterPt(0 To 2) As Double
    Dim dblRadius As Double
    Dim dblInsertPt(0 To 2) As Double
    ' Định nghĩa điểm cơ sở của Block.
    dblBasePt(0) = 0
    dblBasePt(1) = 0
    dblBasePt(2) = 0
    ' Thêm một Block "bulong" vào trong đề án.
    Set objBlock = ThisDrawing.Blocks.Add(dblBasePt, "bulong")
    ThisDrawing.ActiveLinetype = ThisDrawing. _
    Linetypes.Item("CONTINUOUS")
    Dim circleObj As AcadCircle
    Dim centerPoint(0 To 2) As Double
    Dim radius As Double
    ' Định nghĩa các đối tượng.
    centerPoint(0) = 0#: centerPoint(1) = 0#: centerPoint(2) = 0#
    radius = 50#
    objBlock.AddCircle centerPoint, radius
    Dim plineObj As AcadLWPolyline
    Dim points(0 To 13) As Double
    points(0) = 96: points(1) = 0
    points(2) = 48: points(3) = 84
    points(4) = -48: points(5) = 84
    points(6) = -96: points(7) = 0
    points(8) = -48: points(9) = -84
    points(10) = 48: points(11) = -84
    points(12) = 96: points(13) = 0
    objBlock.AddLightWeightPolyline (points)
    ' Định nghĩa điểm chèn Block.
```

```
    dblInsertPt(0) = 0
    dblInsertPt(1) = 0
    dblInsertPt(2) = 0
    ThisDrawing.ModelSpace.InsertBlock dblInsertPt, "bulong", 1#, 1#, 1#, 60
    ThisDrawing.Regen (True)
    ZoomAll
  End Sub
; Kết thúc.
```

+ Sau khi chạy chương trình kết quả như hình 12.9. Đã thực hiện việc chèn Block ("bulong") vào trong bản vẽ.



*Hình 12.9.*

## 12.7. GHI BLOCK LÊN ĐĨA

Một Block có thể được chèn vào nhiều bản vẽ khác nhau. Chính vì vậy mà phải ghi bản vẽ lên đĩa. VBA thực hiện việc ghi Block lên đĩa nhờ lệnh **WBlock**.

**Cú pháp:**

ThisDrawing.WBlock *FileName*, *SelectionSet*

**Trong đó:**

| | |
|---|---|
| *FileName* | Tên file ghi lên đĩa, có kiểu dữ liệu String. |
| *SelectionSet* | Thư mục chứa các đối tượng ghi lên đĩa. |

Đoạn mã chương trình dưới đây thực hiện việc ghi **Block** lên đĩa vào thư mục: ***C:\VBA for ACAD\Drawings\MyBlock.dwg.***

```
; Tên file ghlendia.dvb.
Sub ghlendia()
Dim objSSet1 As AcadSelectionSet
Dim intDxfCode(0 To 1) As Integer
Dim varDxfValue(0 To 1) As Variant
' Tạo một nhóm lựa chọn mới.
Set objSSet1 = ThisDrawing.SelectionSets.Add("SS1")
' Tạo đối tượng mã DXF để chèn vào đúng kiểu.
```

```
intDxfCode(0) = 0
varDxfValue(0) = "INSERT"
intDxfCode(1) = 2
varDxfValue(1) = "MyBlock"
' Nhận tất cả các đối tượng trong bản vẽ.
objSSet1.Select acSelectionSetAll, , , intDxfCode, varDxfValue
' Tiếp tục thực hiện nếu như Block không có.
If objSSet1.Count > 0 Then
Dim varExplodedObjs As Variant
Dim objSSet2 As AcadSelectionSet
' Phá vỡ Block và tạo một thiết lập mới.
varExplodedObjs = objSSet1.Item(0).Explode
' Tạo một thiết lập mới.
Set objSSet2 = ThisDrawing.SelectionSets.Add("SS2")
' Thêm các đối tượng mảng vào lựa chọn.
objSSet2.AddItems varExplodedObjs
' Ghi đối tượng.
ThisDrawing.WBlock "C:\VBA for ACAD\Drawings\MyBlock.dwg", objSSet2
' Thông báo với người sử dụng.
MsgBox """MyBlock"" wBlocked to file " & _
"""C:\VBA for ACAD\Drawings\MyBlock.dwg"""
' Xoá đối tượng.
objSSet2.Erase
' Không thiết lập đối tượng.
objSSet2.Delete
Else
MsgBox """MyBlock"" not found in drawing."
End If
objSSet1.Delete
End Sub
; Kết thúc chương trình.
```

## 12.8. PHÁ VỠ BLOCK

Khi một **Block** được chèn vào trong bản vẽ khi đó nó sẽ là một đối tượng của ACAD. Do vậy, để làm việc với từng đối tượng trong **Block** phải sử dụng lệnh **Explode** để phá vỡ thành nhiều đối tượng.

**Cú pháp:**

*ReturnVal* = *ObjSelected*.Explode

**Trong đó:**

| | |
|---|---|
| *ObjSelected* | Các đối tượng của bản vẽ được chọn. |
| *ReturnVal* | Biến kiểu Variant là mảng các đối tượng Block phá vỡ. |

Đoạn mã chương trình dưới đây lựa chọn Block có tên là "VBA_Block" sau đó phá vỡ Block đó thành các đối tượng.

```
; Tên file VBA_ExplodeInserts.dvb.
Sub VBA_ExplodeInserts()
Dim objSSet As AcadSelectionSet
Dim objBlockRef As AcadBlockReference
Dim intDxfCode(0 To 1) As Integer
Dim varDxfValue(0 To 1) As Variant
Dim intCounter As Integer
' Tạo một sự lựa chọn mới.
Set objSSet = ThisDrawing.SelectionSets.Add("SS1")
intDxfCode(0) = 0
varDxfValue(0) = "INSERT"
intDxfCode(1) = 2
varDxfValue(1) = "VBA_Block"
' Nhận tất cả các trường hợp của "VBA_Block" trong bản vẽ.
objSSet.Select acSelectionSetAll, , , intDxfCode, varDxfValue
' Phá vỡ đối tượng, nếu đối tượng tồn tại.
If objSSet.Count > 0 Then
For intCounter = 0 To objSSet.Count - 1
' Phá vỡ.
objSSet.Item(intCounter).Explode
' Xóa.
objSSet.Item(intCounter).Delete
Next
Else
```

```
MsgBox "No copies of ""VBA_Block"" found in this drawing."
End If
' Thôi không lựa chọn.
objSSet.Delete
  End Sub
; Kết thúc.
```

## 12.9. THUỘC TÍNH CỦA BLOCK

Thuộc tính của Block là các dòng text *(chữ hoặc số)* đi kèm với Block để miêu tả Block. Một Block có thể có nhiều thuộc tính. Lệnh **AddAttribute** dùng để tạo các thuộc tính của Block.

**Cú pháp:**

ObjAttribute = Space.AddAttribute(*Height, Mode, Prompt, InsertionPoint, Tag, Value*)

| | |
|---|---|
| Space | Không gian vẽ hoặc không gian giấy. |
| ObjAttribute | Các đối tượng thuộc tính. |
| *Height* | Chiều cao của Text có kiểu dữ liệu Double. |
| *Mode* | Các phương thức thuộc tính bảng 12.1. |
| *Prompt* | Dòng nhắc khi chèn Block có kiểu dữ liệu String. |
| *InsertionPoint* | Điểm chèn của Block là biến mảng gồm ba phần tử (x, y, z). |
| *Tag* | Tên biến thuộc tính có kiểu dữ liệu String. |
| *Value* | Giá trị mặc định cho biến thuộc tính có kiểu dữ liệu String. |

***Bảng 12.1.*** Phương thức thuộc tính

| | |
|---|---|
| acAttributeModeInvisible | Thuộc tính không hiện lên khi **Block** được chèn vào. |
| acAttributeModeConstant | Giá trị thuộc tính không thay đổi. |
| AcAttributeModeVerify | Khi Block được chọn thì nhập các giá trị thuộc tính vào dòng nhắc. |
| acAttributeModePreset | Khi Block được chọn, thuộc tính là các giá trị mặc định. |

Có thể kết hợp các thuộc tính ở bảng 12.1, bằng cách kết hợp cả hai giá trị. acAttributeModeInvisible + acAttributeModeConstant.

+ Đoạn mã chương trình dưới đây định nghĩa một Block của Valve trong bản vẽ hiện hành. Tên Block là Valve, và thuộc tính SIZE sẽ có giá trị không đổi để chỉ định kích cỡ của Valve.

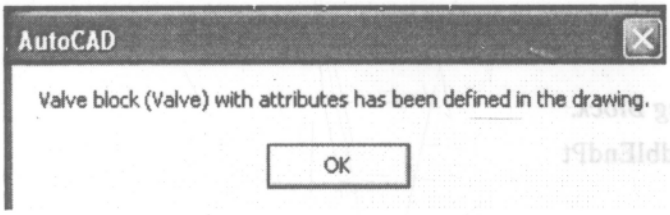
; Tên file VBA_CreateAttributes.dvb

```
Sub VBA_CreateAttributes()
    'Khai báo các đối tượng.
Dim objBlock As AcadBlock
Dim dblBasePt(0 To 2) As Double
Dim dblStartPt(0 To 2) As Double
Dim dblEndPt(0 To 2) As Double
Dim objText As AcadText
Dim strTextValue As String
Dim dblTextInsPt(0 To 2) As Double
Dim dblTextHeight As Double
Dim objAttribute As AcadAttribute
Dim dblAttHeight As Double
Dim intAttMode As Long
Dim strAttPrompt As String
Dim dblAttInsPt(0 To 2) As Double
Dim strAttTag As String
Dim strAttValue As String
' Định nghĩa điểm cơ sở.
dblBasePt(0) = 0
dblBasePt(1) = 0
dblBasePt(2) = 0
' Tạo một Block "Valve".
Set objBlock = ThisDrawing.Blocks.Add(dblBasePt, "Valve")
' Tạo các đường cho Block.
' Định nghĩa đường thẳng đầu tiên.
dblStartPt(0) = -0.25
dblStartPt(1) = -0.125
dblStartPt(2) = 0
dblEndPt(0) = 0.25
dblEndPt(1) = 0.125
dblEndPt(2) = 0
' Thể hiện đường thẳng đó trong Block.
objBlock.AddLine dblStartPt, dblEndPt
' Đường thẳng thứ 2.
dblStartPt(0) = -0.25
```

```
dblStartPt(1) = 0.125
dblStartPt(2) = 0
dblEndPt(0) = 0.25
dblEndPt(1) = -0.125
dblEndPt(2) = 0
' Thể hiện đường thẳng đó trong Block.
objBlock.AddLine dblStartPt, dblEndPt
' Đường thẳng thứ 3.
dblStartPt(0) = 0.25
dblStartPt(1) = -0.125
dblStartPt(2) = 0
dblEndPt(0) = 0.25
dblEndPt(1) = 0.125
dblEndPt(2) = 0
' Thể hiện đường thẳng đó trong Block.
objBlock.AddLine dblStartPt, dblEndPt
' Đường thẳng thứ 4.
dblStartPt(0) = -0.25
dblStartPt(1) = -0.125
dblStartPt(2) = 0
dblEndPt(0) = -0.25
dblEndPt(1) = 0.125
dblEndPt(2) = 0
' Thể hiện đường thẳng đó trong Block.
objBlock.AddLine dblStartPt, dblEndPt
' Chữ.
' Chiều cao chữ.
dblTextHeight = 0.125
' Vị trí chèn chữ.
dblTextInsPt(0) = 0
dblTextInsPt(1) = -0.25
dblTextInsPt(2) = 0
' Định nghĩa chữ.
strTextValue = "VALVE"
Set objText = objBlock.AddText(strTextValue, dblTextInsPt, dblTextHeight)
```

```
objText.Alignment = acAlignmentMiddleCenter
objText.TextAlignmentPoint = dblTextInsPt
dblAttHeight = 0.125
intAttMode = acAttributeModeNormal
strAttPrompt = "Size"
dblAttInsPt(0) = 0
dblAttInsPt(1) = -0.5
dblAttInsPt(2) = 0
strAttTag = "SIZE"
strAttValue = "6"""
Set objAttribute = objBlock.AddAttribute(dblAttHeight, intAttMode, strAttPrompt, _
dblAttInsPt, strAttTag, strAttValue)
objAttribute.Alignment = acAlignmentMiddleCenter
objAttribute.TextAlignmentPoint = dblAttInsPt
' thông báo với người sử dụng
MsgBox "Valve Block (Valve) with attributes has been defined in the drawing."
End Sub
; Kết thúc.
```

Hiện hộp thoại thông báo lên màn hình, khởi tạo Block Valve và các thuộc tính hình 12.10 &12.11.




*Hình 12.10.*


*Hình 12.11.*

## 12.10. CHÈN KHỐI VỚI CÁC THUỘC TÍNH

Khi tạo một **Block** với các thuộc tính, vì vậy mà khi **Block** được chèn vào bản vẽ thì các thuộc tính cũng được chèn vào bản vẽ. Phương thức **GetAttributes** dùng để chèn các giá trị thuộc tính của **Block**.

**Cú pháp:**

*varAttributes* = *BlockRef*.GetAttributes()

**Trong đó:**

| | |
|---|---|
| *varAttributes* | Mảng chứa các thuộc tính có kiểu dữ liệu là Variant. |
| *BlockRef* | Đối tượng BlockRef. |

Đoạn mã chương trình dưới đây tạo một Block là valve trong bản vẽ hiện thời. Tên của Block là Valve và thuộc tính SIZE, chèn Block tại điểm có toạ độ (0, 0, 0) và cập nhật thuộc tính SIZE lên 8''.

```
; Tên file VBA_InsertAttributes.dvb.
  Sub VBA_InsertAttributes()
  Dim objBlock As AcadBlock
  Dim dblBasePt(0 To 2) As Double
  Dim dblStartPt(0 To 2) As Double
  Dim dblEndPt(0 To 2) As Double
  Dim objText As AcadText
  Dim strTextValue As String
  Dim dblTextInsPt(0 To 2) As Double
  Dim dblTextHeight As Double
  Dim objAttribute As AcadAttribute
  Dim dblAttHeight As Double
  Dim intAttMode As Long
  Dim strAttPrompt As String
  Dim dblAttInsPt(0 To 2) As Double
  Dim strAttTag As String
  Dim strAttValue As String
  Dim intCounter As Integer
  Dim dblBlkInsPt(0 To 2) As Double
  Dim objBlockRef As AcadBlockReference
  Dim varAttributes As Variant
  ' Định nghĩa điểm cở sở.
  dblBasePt(0) = 0
  dblBasePt(1) = 0
  dblBasePt(2) = 0
  Set objBlock = ThisDrawing.Blocks.Add(dblBasePt, "Valve")
  ' Tạo các đường thẳng.
  ' Đường thẳng đầu tiên.
```

```
dblStartPt(0) = -0.25
dblStartPt(1) = -0.125
dblStartPt(2) = 0
dblEndPt(0) = 0.25
dblEndPt(1) = 0.125
dblEndPt(2) = 0
objBlock.AddLine dblStartPt, dblEndPt
' Đường thẳng thứ hai.
dblStartPt(0) = -0.25
dblStartPt(1) = 0.125
dblStartPt(2) = 0
dblEndPt(0) = 0.25
dblEndPt(1) = -0.125
dblEndPt(2) = 0
objBlock.AddLine dblStartPt, dblEndPt
' Đường thẳng thứ ba.
dblStartPt(0) = 0.25
dblStartPt(1) = -0.125
dblStartPt(2) = 0
dblEndPt(0) = 0.25
dblEndPt(1) = 0.125
dblEndPt(2) = 0
objBlock.AddLine dblStartPt, dblEndPt
' đường thẳng thứ tư.
dblStartPt(0) = -0.25
dblStartPt(1) = -0.125
dblStartPt(2) = 0
dblEndPt(0) = -0.25
dblEndPt(1) = 0.125
dblEndPt(2) = 0
objBlock.AddLine dblStartPt, dblEndPt
' Chữ.
' Chiều cao chữ.
dblTextHeight = 0.125
' Vị trí chèn chữ
```

```
dblTextInsPt(0) = 0
dblTextInsPt(1) = -0.25
dblTextInsPt(2) = 0
strTextValue = "VALVE"
Set objText = objBlock.AddText(strTextValue, dblTextInsPt, dblTextHeight)
objText.Alignment = acAlignmentMiddleCenter
objText.TextAlignmentPoint = dblTextInsPt
dblAttHeight = 0.125
intAttMode = acAttributeModeNormal
strAttPrompt = "Size"
dblAttInsPt(0) = 0
dblAttInsPt(1) = -0.5
dblAttInsPt(2) = 0
strAttTag = "SIZE"
strAttValue = "6"""
Set objAttribute = objBlock.AddAttribute(dblAttHeight, intAttMode, strAttPrompt, _
dblAttInsPt, strAttTag, strAttValue)
objAttribute.Alignment = acAlignmentMiddleCenter
objAttribute.TextAlignmentPoint = dblAttInsPt
' Chèn Block.
dblBlkInsPt(0) = 0
dblBlkInsPt(1) = 0
dblBlkInsPt(2) = 0
Set objBlockRef = ThisDrawing.ModelSpace.InsertBlock(dblBlkInsPt, _
"Valve", 1, 1, 1, 0)
varAttributes = objBlockRef.GetAttributes
For intCounter = LBound(varAttributes) To UBound(varAttributes)
If varAttributes(intCounter).TagString = "SIZE" Then
varAttributes(intCounter).TextString = "8"""
End If
Next
End Sub
; Kết thúc chương trình.
```

| | |
|---|---|
| + Khi chạy chương trình cho ta kết quả như hình 12.12. |  <br>*Hình 12.12.* |

## 12.11. HIỂN THỊ CÁC THUỘC TÍNH BLOCK

Để hiển thị các thuộc tính của **Block** sử dụng phương thức **GetAttributes**. Ngoài ra còn có thể xác định qua thuộc tính **HasAttributes**.

Thuộc tính **HasAttributes** của đối tượng **BlockRef** cho phép xác định rõ **Block** có các thuộc tính kết hợp với nhau. Thuộc tính **HasAttributes** có kiểu dữ liệu Boolean trả về hai giá trị:

**True**: Nếu **Block** có thuộc tính.

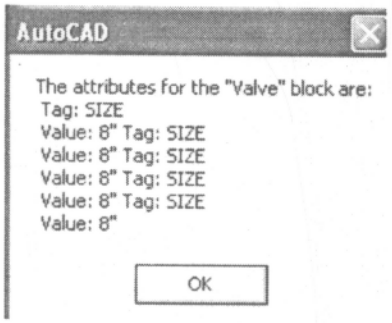
**False**: Nếu **Block** không có thuộc tính.

Đoạn mã chương trình dưới đây tạo một **Block** Valve, sau đó kiểm tra xem **Block** đó có thuộc tính tạo bởi thuộc tính **HasAttributes**. Nếu phải thì tên của thuộc tính và giá trị của thuộc tính được hiển thị trên hộp thoại.

```
; Tên file VBA_GetAttributes.dvb.
 Sub VBA_GetAttributes()
 Dim objSSet As AcadSelectionSet
 Dim objBlockRef As AcadBlockReference
 Dim intDxfCode(0 To 1) As Integer
 Dim varDxfValue(0 To 1) As Variant
 Dim intCounter As Integer
 Set objSSet = ThisDrawing.SelectionSets.Add("SS1")
 intDxfCode(0) = 0
 varDxfValue(0) = "INSERT"
 intDxfCode(1) = 2
 varDxfValue(1) = "Valve"
 objSSet.Select acSelectionSetAll, , , intDxfCode, varDxfValue
 If objSSet.Count > 0 Then
 Dim varAttributes As Variant
 varAttributes = objSSet.Item(0).GetAttributes
 If objSSet.Item(0).HasAttributes Then
 Dim strAttributes As String
 strAttributes = ""
```

```
For intCounter = LBound(varAttributes) To UBound(varAttributes)
strAttributes = strAttributes & " Tag: " & _
varAttributes(intCounter).TagString & vbCrLf & _
" Value: " & varAttributes(intCounter).TextString
Next
MsgBox "The attributes for the ""Valve"" Block are: " & _
vbCrLf & strAttributes
End If
Else
MsgBox """Valve"" not inserted in drawing."
End If
objSSet.Delete
End Sub
; Kết thúc.
```

Khi chạy chương trình ta có kết quả như hình 12.13.





***Hình 12.13.*** Hiện thị thuộc tính của Block.

## 12.12. NHẬN CÁC HẰNG SỐ THUỘC TÍNH

Lệnh **GetConstantAttributes** để nhận các hằng số thuộc tính của **Block**.

**Cú pháp:**

*varAttributes* = *BlockRef*.GetConstantAttributes()

**Trong đó:**

| | |
|---|---|
| *varAttributes* | Chứa các thuộc tính của khối có kiểu dữ liệu Variant |
| *BlockRef* | Đối tượng BlockRef. |

Đoạn mã chương trình dưới đây định nghĩa một Block valve trong bản vẽ hiện hành. Tên **Block** có tên là Valve. Các hằng số thuộc tính sẽ được hiển thị trong hộp thoại.

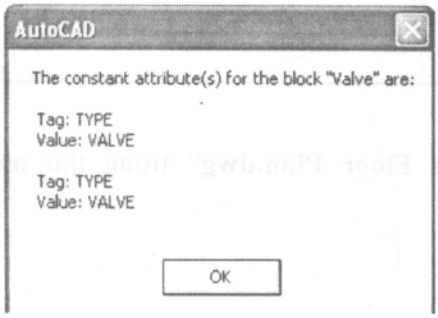
```
; Tên file VBA_GetAllAttributes.dvb.
Sub VBA_GetAllAttributes()
' Khai báo các đối tượng.
Dim objBlock As AcadBlock
Dim dblBasePt(0 To 2) As Double
Dim dblStartPt(0 To 2) As Double
Dim dblEndPt(0 To 2) As Double
Dim objAttribute As AcadAttribute
Dim dblAttHeight As Double
Dim intAttMode As Long
Dim strAttPrompt As String
Dim dblAttInsPt(0 To 2) As Double
Dim strAttTag As String
Dim strAttValue As String
Dim intCounter As Integer
Dim dblBlkInsPt(0 To 2) As Double
Dim objBlockRef As AcadBlockReference
Dim strAttributes As String
Dim varAttributes As Variant
' Định nghĩa điểm cơ sở của Block.
dblBasePt(0) = 0
dblBasePt(1) = 0
dblBasePt(2) = 0
' Tạo một Block Valve.
Set objBlock = ThisDrawing.Blocks.Add(dblBasePt, "Valve")
' Tạo các đường thẳng của Valve.
' Đường thẳng đầu tiên.
dblStartPt(0) = -0.25
dblStartPt(1) = -0.125
dblStartPt(2) = 0
dblEndPt(0) = 0.25
dblEndPt(1) = 0.125
dblEndPt(2) = 0
objBlock.AddLine dblStartPt, dblEndPt
' Đường thẳng thứ hai.
dblStartPt(0) = -0.25
```

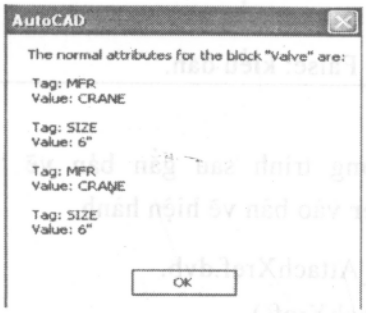
```
dblStartPt(1) = 0.125
dblStartPt(2) = 0
dblEndPt(0) = 0.25
dblEndPt(1) = -0.125
dblEndPt(2) = 0
objBlock.AddLine dblStartPt, dblEndPt
' Đường thẳng thứ ba.
dblStartPt(0) = 0.25
dblStartPt(1) = -0.125
dblStartPt(2) = 0
dblEndPt(0) = 0.25
dblEndPt(1) = 0.125
dblEndPt(2) = 0
objBlock.AddLine dblStartPt, dblEndPt
' Đường thẳng thứ tư.
dblStartPt(0) = -0.25
dblStartPt(1) = -0.125
dblStartPt(2) = 0
dblEndPt(0) = -0.25
dblEndPt(1) = 0.125
dblEndPt(2) = 0
objBlock.AddLine dblStartPt, dblEndPt
' Tạo các thuộc tính của khối.
' Chiều cao.
dblAttHeight = 0.125
' Phương thức thuộc tính.
IntAttMode = acAttributeModeConstant
strAttPrompt = "Valve type"
' Điểm chèn của thuộc tính.
dblAttInsPt(0) = 0
dblAttInsPt(1) = -0.25
dblAttInsPt(2) = 0
strAttTag = "TYPE"
' Mặc định giá trị thuộc tính.
strAttValue = "VALVE"
' Tạo các thuộc tính cho Block.
```

```
Set objAttribute = objBlock.AddAttribute(dblAttHeight, intAttMode, _
strAttPrompt, dblAttInsPt, strAttTag, strAttValue)
objAttribute.Alignment = acAlignmentMiddleCenter
objAttribute.TextAlignmentPoint = dblAttInsPt
' Định nghĩa thuộc tính thứ hai.
dblAttHeight = 0.125
intAttMode = acAttributeModeInvisible
' Nhập dòng nhắc khi chèn Block với thuộc tính.
strAttPrompt = "Manufacturer"
' Điểm chèn thuộc tính.
dblAttInsPt(0) = 0
dblAttInsPt(1) = -0.5
dblAttInsPt(2) = 0
strAttTag = "MFR"
' Mặc định giá trị thuộc tính.
strAttValue = "CRANE"
Set objAttribute = objBlock.AddAttribute(dblAttHeight, intAttMode, _
strAttPrompt, dblAttInsPt, strAttTag, strAttValue)
objAttribute.Alignment = acAlignmentMiddleCenter
objAttribute.TextAlignmentPoint = dblAttInsPt
' Tạo thuộc tính thứ ba của Block.
dblAttHeight = 0.125
intAttMode = acAttributeModeNormal
strAttPrompt = "Size"
dblAttInsPt(0) = 0
dblAttInsPt(1) = -0.75
dblAttInsPt(2) = 0
strAttTag = "SIZE"
' Mặc định giá trị thuộc tính.
strAttValue = "6"""
Set objAttribute = objBlock.AddAttribute(dblAttHeight, intAttMode, _
strAttPrompt, dblAttInsPt, strAttTag, strAttValue)
objAttribute.Alignment = acAlignmentMiddleCenter
objAttribute.TextAlignmentPoint = dblAttInsPt
' Điểm chèn Block.
dblBlkInsPt(0) = 0
```

```
dblBlkInsPt(1) = 0
dblBlkInsPt(2) = 0
Set objBlockRef = ThisDrawing.ModelSpace.InsertBlock(dblBlkInsPt, _
"Valve", 1, 1, 1, 0)
varAttributes = objBlockRef.GetConstantAttributes
strAttributes = ""
For intCounter = LBound(varAttributes) To UBound(varAttributes)
strAttributes = strAttributes & " Tag: " & _
varAttributes(intCounter).TagString & vbCrLf & _
" Value: " & varAttributes(intCounter).TextString & vbCrLf & vbCrLf
Next
' Hiện các thuộc tính của Block với người sử dụng.
MsgBox "The constant attribute(s) for the Block ""Valve"" are: " & _
vbCrLf & vbCrLf & strAttributes
varAttributes = objBlockRef.GetAttributes
strAttributes = ""
For intCounter = LBound(varAttributes) To UBound(varAttributes)
strAttributes = strAttributes & " Tag: " & _
varAttributes(intCounter).TagString & vbCrLf & _
" Value: " & varAttributes(intCounter).TextString & vbCrLf & vbCrLf
Next
MsgBox "The normal attributes for the Block ""Valve"" are: " & _
vbCrLf & vbCrLf & strAttributes
End Sub
; Kết thúc;
```

Hình 12.14 minh hoạ kết quả của chương trình





***Hình 12.14.*** Kết quả chương trình.

## 12.13. THAM KHẢO NGOÀI

Bản vẽ tham khảo ngoài dùng để quan sát các bản vẽ khác nhau trên màn hình đồ hoạ hiện hành và hiện lên trên bản vẽ hiện hành, tuy nhiên không thêm bớt hay sửa đổi được. Một trong các đặc điểm quan trọng của bản vẽ tham khảo ngoài là không hiệu chỉnh trong bản vẽ hiện hành. Mặc dù bản vẽ tham khảo ngoài được đặt tên (*lớp, kiểu đường, kiểu Text*...) và gắn vào bản vẽ hiện hành nhưng bản vẽ đó được bảo vệ.

Bản vẽ tham khảo ngoài có nhiều ứng dụng và đặc biệt hữu ích khi làm việc theo nhóm trên mạng hoặc theo một nhóm nghiên cứu.

### 12.13.1. Gắn một bản vẽ vào một bản vẽ khác

Lệnh **AttachExternalReference** dùng để tham khảo bản vẽ ngoài theo kiểu gắn một bản vẽ vào một bản vẽ khác.

**Cú pháp:**

ObjXRef = Space.AttachExternalReference (*Path, Name, InsertPt, XScale, _*
*YScale, ZScale, Rotation, Overlay*)

**Trong đó:**

| | |
|---|---|
| ObjXRef | Đối tượng tham khảo bản vẽ ngoài. |
| Space | Không gian vẽ hoặc không gian giấy. |
| *Path* | Đường dẫn đến tham khảo ngoài có kiểu dữ liệu String. |
| *Name* | Tên của tham khảo ngoài có kiểu dữ liệu String. |
| *InsertPt* | Là điểm chèn gồm ba phần tử (x, y, z) có kiểu dữ liệu Double. |
| *XScale* | Hệ số tỷ lệ theo trục X có kiểu dữ liệu Double. |
| *YScale* | Hệ số tỷ lệ theo trục Y có kiểu dữ liệu Double. |
| *ZScale* | Hệ số tỷ lệ theo trục Z có kiểu dữ liệu Double. |
| *Rotation* | Góc quay (đơn vị: *radian*) có kiểu dữ liệu Double. |
| *Overlay* | Có kiểu dữ liệu Boolean với 2 giá trị:<br>+ True : kiểu phủ lên.<br>+ False: kiểu dán. |

Đoạn mã chương trình sau gắn bản vẽ "**8th Floor Plan.dwg**" trong thư mục **ACAD \Sample folder** vào bản vẽ hiện hành.

```
; Tên file VBA_AttachXref.dvb.
 Sub VBA_AttachXref()
 Dim dblInsertPt(0 To 2) As Double
```

```
Dim strXrefPath As String
' Định nghĩa tham khảo ngoài để chèn vào.
strXrefPath = _
"C:\Program Files\ACAD 2005\Sample\8th Floor Plan.dwg"
' Định nghĩa điểm chèn tại 0,0,0.
dblInsertPt(0) = 0
dblInsertPt(1) = 0
dblInsertPt(2) = 0
' Gắn tham khảo bản vẽ ngoài đến bản vẽ hiện thời.
ThisDrawing.ModelSpace.AttachExternalReference strXrefPath, _
  "8th Floor Plan", dblInsertPt, 1, 1, 1, 0, False
' Thu nhỏ màn hình.
ZoomExtents
' Thông báo với người sử dụng.
MsgBox "8th Floor Plan.dwg attached to the current drawing at 0,0,0."
End Sub
; Kết thúc.
```

### 12.13.2. Các lựa chọn

#### *12.13.2.1. Detach*

Khi không cần tham khảo ngoài có thể loại bỏ nó bằng lệnh **Detach**.

**Cú pháp:**

ObjXRef.Detach

**Trong đó:**

| | |
|---|---|
| ObjXRef | Đối tượng tham khảo ngoài. |

Đoạn mã chương trình dưới đây sẽ loại bỏ tham khảo bản vẽ ngoài "8th Floor Plan" khỏi bản vẽ hiện hành.

```
; Tên file VBA_DetachXref.dvb.
Sub VBA_DetachXref()
' Báo lỗi nếu file không tồn tại.
On Error GoTo NOTFOUND
' Loại bỏ việc tham khảo bản vẽ ngoài.
ThisDrawing.Blocks.Item("8th Floor Plan").Detach
' Thông báo với người sử dụng.
```

```
MsgBox "8th Floor Plan.dwg was detached from the current drawing."
Exit Sub
NOTFOUND:
MsgBox "8th Floor Plan.dwg was not attached to this drawing."
End Sub
```

### *12.13.2.2. Reload*

Lựa chọn **Reload** dùng để cập nhật sự thay đổi của bản vẽ tham khảo ngoài và bản vẽ hiện hành mà không cần thoát khỏi ACAD hoặc mở lại bản vẽ của bạn. Lựa chọn này dùng cho một nhóm thiết kế cùng một bản vẽ qua mạng.

Lệnh **Reload** dùng để làm điều đó và có cú pháp như sau:

**Cú pháp:**

ObjXRef.Reload

**Trong đó:**

| | |
|---|---|
| ObjXRef | Đối tượng tham khảo ngoài. |

Đoạn mã chương trình dưới đây sẽ **Reload** bản vẽ "8th Floor Plan" trong bản vẽ hiện hành.

```
; Tên file VBA_ReloadXref.dvb
 Sub VBA_ReloadXref()
 On Error GoTo NOTFOUND
 ' Reload tham khảo ngoài.
 ThisDrawing.Blocks.Item("8th Floor Plan").Reload
 ' Thông báo với người sử dụng.
 MsgBox "8th Floor Plan.dwg was reloaded into the current drawing."
 Exit Sub
 ' Nếu tham khảo ngoài không tồn tại.
 NOTFOUND:
 MsgBox "8th Floor Plan.dwg was not attached to this drawing."
 End Sub
; Kết thúc.
```

### *12.13.2.3. Unload*

Lựa chọn **Unload** làm tăng tốc độ tính toán trong bản vẽ. Khi ta chọn **Unload** thì bản vẽ tham khảo ngoài sẽ không nhìn thấy trong bản vẽ hiện hành, tuy nhiên các thông tin về bản vẽ tham khảo ngoài vẫn tồn tại. Do vậy mà tốc độ mở bản vẽ và thời gian tái tạo bản vẽ

sẽ nhanh hơn, tốn ít bộ nhớ hơn. Trong trường hợp này để làm xuất hiện bản vẽ tham khảo ngoài ta chỉ cần **dùng** lệnh **Reload**.

Lệnh **Unload** dùng để bỏ bản vẽ tham khảo ngoài từ bản vẽ hiện hành.

**Cú pháp:**

ObjXRef.Unload

**Trong đó:**

| | |
|---|---|
| ObjXRef | Đối tượng bản vẽ tham khảo ngoài cần Unload. |

Đoạn mã chương trình sau sẽ Unload bản vẽ tham khảo ngoài "8th Floor Plan" từ bản vẽ hiện hành.

```
; Tên file VBA_UnloadXref.dvb.
 Sub VBA_UnloadXref()
 On Error GoTo NOTFOUND
 ' Unload tham khảo bản vẽ ngoài.
 ThisDrawing.Blocks.Item("8th Floor Plan").Unload
 ' Thông báo với người sử dụng.
 MsgBox "8th Floor Plan.dwg was unloaded from the current drawing."
 Exit Sub
 NOTFOUND:
 MsgBox "8th Floor Plan.dwg was not attached to this drawing."
  End Sub
; Kết thúc.
```

### *12.13.2.4. Blind*

Lựa chọn **Blind** này cho phép chuyển bản vẽ tham khảo ngoài thành **Block** của bản vẽ hiện hành.

**Cú pháp:**

ObjXRef.Bind *BindType*

**Trong đó:**

| | |
|---|---|
| ObjXRef | Đối tượng Block phá vỡ cần Blind. |

Đoạn mã chương trình dưới đây sẽ chuyển đổi bản vẽ tham khảo ngoài "8th Floor Plan" thành Block trong bản vẽ hiện hành.

```
; Tên file VBA_BindXref.dvb
 Sub VBA_BindXref()
 On Error GoTo NOTFOUND
 ' Bind tham khảo bản vẽ ngoài.
 ThisDrawing.Blocks.Item("8th Floor Plan").Bind False
 ' Thông báo với người sử dụng.
 MsgBox "8th Floor Plan.dwg was bound to the current drawing."
 Exit Sub
 NOTFOUND:
 MsgBox "8th Floor Plan.dwg was not attached to the drawing."
 End Sub
; Kết thúc.
```

## Chương 13

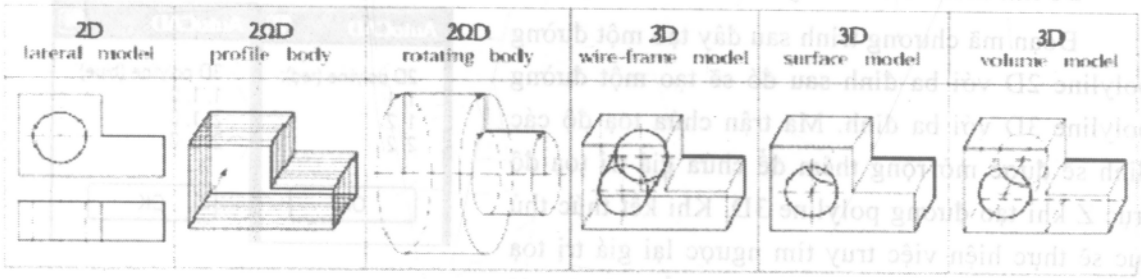
# THIẾT KẾ MÔ HÌNH BA CHIỀU

Chương này trình bày các lệnh tạo khối rắn và các bề mặt cơ bản trong không gian ba chiều. Đây là cơ sở để thiết kế các chi tiết máy ở dạng 3D.

## 13.1. TỔNG QUAN VỀ MÔ HÌNH 3D

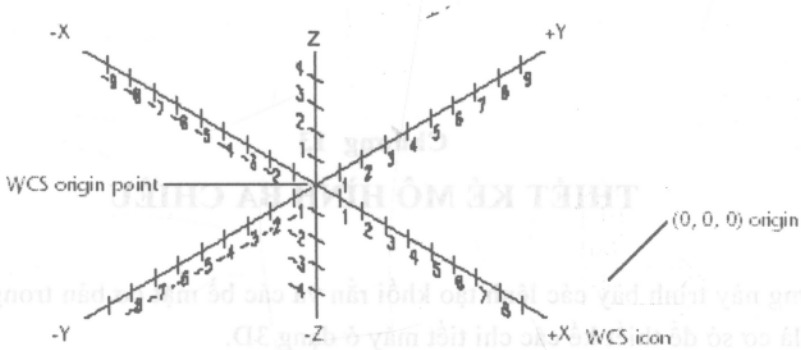
Mô hình hoá ba chiều là một lĩnh vực phát triển nhanh chóng trong ACAD, nó là một cuộc cách mạng trong việc ứng dụng máy tính vào quá trình thiết kế. Theo phương pháp vẽ truyền thống, các bản vẽ hai chiều (2D) trình bày vật thể trong mặt phẳng 2D, điều này có một số hạn chế là việc đọc các bản vẽ 2D để xây dựng lên các chi tiết 3D có thể xảy ra các sai sót nếu bản vẽ không rõ ràng. Ngược lại, các mô hình 3D không chỉ là vẽ một đối tượng mà là biểu diễn hình ảnh thực của vật thể. Do đó vật thể được thể hiện thật hơn và rõ ràng hơn các bản vẽ 2D. Hơn nữa ta có thể chuyển từ mô hình 3D sang các hình chiếu 2D và tạo các bản vẽ chế tạo ở dạng 2D, có các kiểu mô hình như hình 13.1.

Nói chung có ba loại mô hình được dùng để biểu diễn đối tượng vật lý trong các hệ CAD/CAM đó là: mô hình khung dây, mô hình bề mặt và mô hình vật thể rắn. Dưới đây là các dạng mô hình được biểu diễn trong CAD .



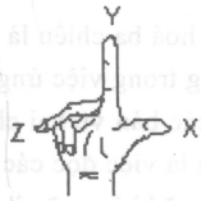

***Hình 13.1.*** Các mô hình khác nhau trong CAD.

Việc sử dụng VB&VBA để vẽ các chi tiết 3D trong không gian 3 chiều tương đối đơn giản. Nếu trong bản vẽ hai chiều ta chỉ nhập vào toạ độ X, Y thì trong bản vẽ ba chiều ta nhập thêm giá trị toạ độ theo trục Z trong hệ toạ độ WCS *(hệ toạ độ gốc mặc định trong bản vẽ ACAD)* hoặc hệ toạ độ UCS *(hệ toạ độ được định nghĩa bởi người sử dụng)*.




*Hình 13.2*. Hệ toạ độ WCS.

Hướng trục Z vuông góc với mặt phẳng XY và tuân theo quy tắc ***bàn tay phải***: ngón cái là trục X, ngón trỏ là trục Y và ngón giữa là trục Z. Chiều quay dương theo ngược chiều kim đồng hồ nhìn từ đỉnh trục về phía gốc toạ độ.



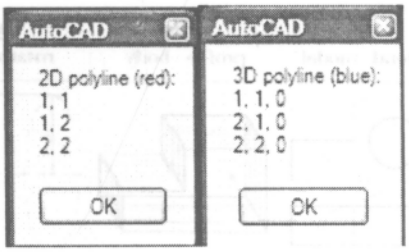

*Hình 13.3*. Quy tắc bàn tay phải.

## 13.2. NHẬP CÁC GIÁ TRỊ TỌA ĐỘ X, Y, Z TRONG HỆ TỌA ĐỘ BA CHIỀU

Việc nhập các giá trị toạ độ trong hệ toạ độ ba chiều WCS cũng tương tự như cách nhập các giá trị toạ độ trong hệ toạ độ hai chiều WCS, ngoài các giá trị toạ độ X, Y thì ta nhập thêm giá trị toạ độ cho trục Z.

***Để tìm hiểu rõ điều này ta xét các ví dụ sau:***

Đoạn mã chương trình sau đây tạo một đường polyline 2D với ba đỉnh sau đó sẽ tạo một đường polyline 3D với ba đỉnh. Ma trận chứa toạ độ các đỉnh sẽ được mở rộng thêm để chứa giá trị toạ độ trục Z khi tạo đường polyline 3D. Khi kết thúc thủ tục sẽ thực hiện việc truy tìm ngược lại giá trị toạ độ các điểm của đường polyline và hiển thị các giá trị toạ độ trong một hộp hội thoại như hình 13.4.




*Hình 13.4*. Hộp thoại hiển thị toạ độ.

```
; Tên file VBA_Polyline_2D_3D.dvb:
  Sub VBA_Polyline_2D_3D()
    Dim pline2DObj As AcadLWPolyline
    Dim pline3DObj As AcadPolyline
    Dim points2D(0 To 5) As Double
    Dim points3D(0 To 8) As Double
```

```
    points2D(0) = 1: points2D(1) = 1
    points2D(2) = 1: points2D(3) = 2
    points2D(4) = 2: points2D(5) = 2
     points3D(0) = 1: points3D(1) = 1: points3D(2) = 0
     points3D(3) = 1: points3D(4) = 2: points3D(5) = 0
     points3D(6) = 2: points3D(7) = 2: points3D(8) = 0
     Set pline2DObj = ThisDrawing.ModelSpace. _
                    AddLightWeightPolyline(points2D)
     pline2DObj.Color = acRed
     pline2DObj.Update
     Set pline3DObj = ThisDrawing.ModelSpace. _  AddPolyline(points3D)
     pline3DObj.Color = acBlue
     pline3DObj.Update
     Dim get2Dpts As Variant
     Dim get3Dpts As Variant
     get2Dpts = pline2DObj.Coordinates
     get3Dpts = pline3DObj.Coordinates
     MsgBox ("2D polyline (red): " & vbCrLf & _
        get2Dpts(0) & ", " & get2Dpts(1) & vbCrLf & _
        get2Dpts(2) & ", " & get2Dpts(3) & vbCrLf & _
        get2Dpts(4) & ", " & get2Dpts(5))
     MsgBox ("3D polyline (blue): " & vbCrLf & _
        get3Dpts(0) & ", " & get3Dpts(1) & ", " & _
        get3Dpts(2) & vbCrLf & _
        get3Dpts(3) & ", " & get3Dpts(4) & ", " & _
        get3Dpts(5) & vbCrLf & _
        get3Dpts(6) & ", " & get3Dpts(7) & ", " & _
        get3Dpts(8))
End Sub
; Kết thúc.
```

Trong thủ tục trên có sử dụng kiểu ***Variant*** cho các biến **get2Dpts** và **get3Dpts**. Kiểu ***Variant*** là một kiểu dữ liệu đặc biệt có thể chứa dữ liệu thuộc bất cứ dạng nào ngoại trừ các dữ liệu kiểu chuỗi có chiều dài cố định và dữ liệu có kiểu được định nghĩa bởi người sử dụng. Kiểu ***Variant*** có thể chứa các giá trị đặc biệt như là **Empty, Error, Nothing** và **NULL**. Kiểu ***Variant*** được sử dụng để truyền mảng dữ liệu vào/ra **ACAD ActiveX**, điều

này có nghĩa là các mảng dữ liệu phải được định kiểu là **Variant** thì mới có thể được chấp nhận và xử lý được bằng các phương thức và thuộc tính của **ACAD ActiveX**.

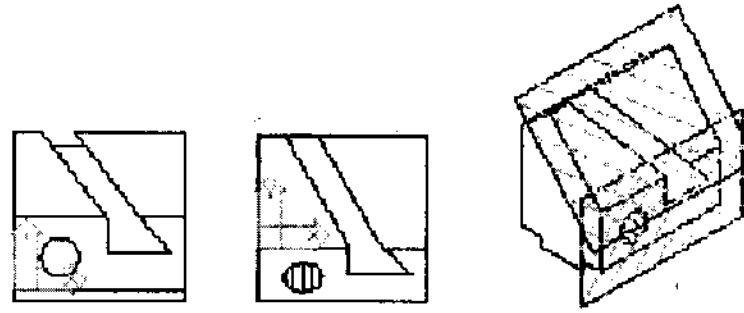
> ***Chú ý:*** *Trong ACAD, các mảng dữ liệu nhập vào khi sử dụng ngôn ngữ VB & VBA đều được tự động chuyển đổi sang kiểu dữ liệu **Variant**. **ACAD ActiveX** cung cấp một phương thức để chuyển đổi một mảng dữ liệu sang kiểu dữ liệu **Variant**. Phương thức này là **CreateTypedArray**.*

### 13.3. ĐỊNH NGHĨA MỘT HỆ TỌA ĐỘ UCS

Người sử dụng có thể định nghĩa một hệ toạ độ mới (UCS) để thay đổi vị trí của điểm gốc (0, 0, 0), phương của mặt phẳng XY và trục Z. Hệ toạ độ UCS có thể được đặt ở vị trí bất kỳ trong không gian 3D và tùy vào điểm nhìn biểu tượng của chúng sẽ hiện thị khác nhau. Số lượng UCS trong một bản vẽ không hạn chế, có thể tạo mới, lưu và gọi lại hệ toạ độ UCS đã được định nghĩa. UCS giúp cho việc thực hiện bản vẽ 3D được dễ dàng hơn. Tuy nhiên cùng một lúc chỉ có một trong hai hệ toạ độ WCS hoặc UCS là hiện hành.

+ Để hiển thị biểu tượng điểm gốc và các chiều của một hệ toạ độ UCS, ta có thể sử dụng thuộc tính **UCSIconAtOrigin**. Nếu biểu tượng UCS được bật để hiển thị *(xem thuộc tính **UCSIconOn**)* nhưng biểu tượng không hiển thị tại điểm gốc toạ độ mà được hiển thị trong hệ toạ độ WCS ta có thể sử dụng biến hệ thống **UCSORG** để định nghĩa lại .

+ Để tạo một hệ toạ độ UCS mới sử dụng phương thức **Add** phương thức này yêu cầu nhập vào bốn giá trị: (*toạ độ của điểm gốc, toạ độ của một điểm trên trục X, toạ độ của một điểm trên trục Y, tên của hệ toạ độ UCS*).



UCS thứ nhất UCS thứ hai Mô hình với 2 UCS

***Hình 13.5.*** Hình minh hoạ hệ toạ độ UCS.

+ Tất cả các giá trị toạ độ trong **ACAD ActiveX** đều được nhập trong hệ toạ độ gốc WCS. Do đó ta phải sử dụng phương thức **GetUCSMatrix** để trả về ma trận chuyển đổi của một hệ toạ độ **UCS** sau đó sử dụng ma trận chuyển đổi này để tính ra giá trị toạ độ tương ứng của hệ toạ độ gốc **WCS**.

+ Để kích hoạt một hệ toạ độ **UCS** sử dụng thuộc tính **ActiveUCS** trong đối tượng **Document.** Nếu thay đổi một **UCS** hiện hành thì **UCS** này cần phải được kích hoạt lại để cập nhật các thay đổi bằng cách gọi lại thuộc tính **ActiveUCS.**

Dưới đây là đoạn mã chương trình sau đây sẽ tạo một **UCS** mới, kích hoạt **UCS** này thành UCS hiện hành. Sau đó thủ tục sẽ yêu cầu người sử dụng chọn một điểm bất kỳ trong bản vẽ và trả về toạ độ của điểm đó theo cả hai toạ độ WCS và UCS.

```
; Tên file VBA_NewUCS.dvb.
Sub VBA_NewUCS()
   Dim ucsObj As AcadUCS
   Dim origin(0 To 2) As Double
   Dim xAxisPnt(0 To 2) As Double
   Dim yAxisPnt(0 To 2) As Double
   origin(0) = 4: origin(1) = 5: origin(2) = 3
   xAxisPnt(0) = 5: xAxisPnt(1) = 5: xAxisPnt(2) = 3
   yAxisPnt(0) = 4: yAxisPnt(1) = 6: yAxisPnt(2) = 3
   Set ucsObj = ThisDrawing.UserCoordinateSystems. _
         Add(origin, xAxisPnt, yAxisPnt, "New_UCS")
   ThisDrawing.ActiveViewport.UCSIconAtOrigin = True
   ThisDrawing.ActiveViewport.UCSIconOn = True
   ThisDrawing.ActiveUCS = ucsObj
   MsgBox "The current UCS is : " & ThisDrawing.ActiveUCS.Name _
         & vbCrLf & " Pick a point in the drawing."
   Dim WCSPnt As Variant
   Dim UCSPnt As Variant
   WCSPnt = ThisDrawing.Utility.GetPoint(, "Enter a point: ")
   UCSPnt = ThisDrawing.Utility.TranslateCoordinates _
         (WCSPnt, acWorld, acUCS, False)
   MsgBox "The WCS coordinates are: " & WCSPnt(0) & ", " _
         & WCSPnt(1) & ", " & WCSPnt(2) & vbCrLf & _
         "The UCS coordinates are: " & UCSPnt(0) & ", " _
         & UCSPnt(1) & ", " & UCSPnt(2)
End Sub
; Kết thúc.
```

## 13.4. MÔ HÌNH KHUNG DÂY *(Wireframe model)*

Còn được gọi **edge-verfex** *(đỉnh cạnh)* là phương pháp đơn giản nhất của mô hình hình học và thường được dùng để xác định các mô hình tính toán của chi tiết, đặc biệt trong các hệ thống vẽ trợ giúp bằng máy tính.

Mô hình khung dây không chứa đựng thông tin về bề mặt của chi tiết cũng không phân biệt phía trong và phía ngoài của bề mặt. Mô hình khung dây thường không rõ ràng đặc biệt khi biểu diễn chi tiết 3D và đôi lúc có thể biểu diễn bằng nhiều kiểu khác nhau.

Mô hình khung dây là một bản phác thảo của đối tượng 3D. Mô hình khung dây chỉ bao gồm các điểm trong không gian, các đường thẳng hoặc đường cong nối chúng lại với nhau. Không có các mặt trong mô hình khung dây. Toàn bộ các đối tượng của mô hình đều được nhìn thấy.


***Hình 13.6.*** Mô hình khung dây.

Ta có thể tạo các mô hình khung dây bằng cách đặt các đối tượng 2D trong không gian ba chiều. ACAD cũng cung cấp một số đối tượng khung dây 3D như là 3D polyline.

Lệnh **Add3Dpoly** dùng để tạo các đa tuyến trong 3D bao gồm các phân đoạn là các đoạn thẳng.

**Cú pháp:**

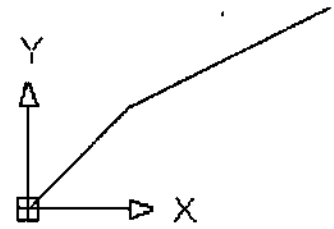
Object = space.Add3Dpoly(*PointsArray*)

**Trong đó:**

| | |
|---|---|
| Object | Đối tượng 3Dpolyline mới được tạo ra. |
| space | Không gian vẽ hoặc không gian giấy. |
| *PointsArray* | Các điểm của đa tuyến gồm ba phần tử (x, y, z) có kiểu dữ liệu Double. |

***Chú ý:** Để đóng mở đường **polyline** sử dụng thuộc tính **closed**.*

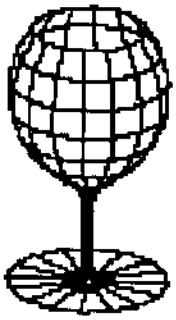
Đoạn mã chương trình sau tạo một đường 3DPolyline trong không gian vẽ như hình 13.7.




***Hình 13.7.*** 3DPolyline

```
; Tên file VBA_Add3DPoly.dvb.
 Sub VBA_Add3DPoly()
    'Khai báo các đối tượng.
    Dim polyObj As Acad3DPolyline
    Dim points(0 To 8) As Double
    ' Định nghĩa các điểm của 3DPolyline
    points(0) = 0: points(1) = 0: points(2) = 0
    points(3) = 10: points(4) = 10: points(5) = 10
    points(6) = 30: points(7) = 20: points(8) = 30
    ' Tạo đường 3DPolyline trong không gian vẽ.
    Set polyObj = ThisDrawing.ModelSpace.Add3DPoly(points)
    ZoomAll
 End Sub
; Kết thúc.
```

## 13.5. MÔ HÌNH BỀ MẶT *(Surface model)*



***Hình 13.8***. Mô hình mặt cong.

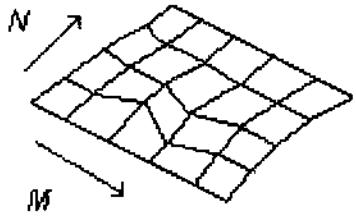
Mô hình bề mặt phức tạp hơn mô hình khung dây nhưng nó cũng biểu diễn đối tượng tốt hơn mô hình khung dây. Mô hình bề mặt bao gồm thêm cả các bề mặt được biểu diễn dưới dạng lưới.

Các mặt lưới chữ nhật (*lưới đa giác*) được dùng để biểu diễn bề mặt cong của đối tượng. Mật độ lưới được xác định bằng ma trận gồm M và N là các đỉnh (*tương tự như các hàng và cột*). Khi sử dụng bề mặt lưới thì không cần chi tiết về các thuộc tính vật lý của đối tượng như (*thể tích, khối lượng, trọng tâm*) nhưng vẫn có thể che các mặt khuất hay tô bóng đối tượng.

Bề mặt lưới có thể là mặt hở hoặc kín. Nếu một mặt lưới hở thì phải xác định chiều hở nếu cạnh bắt đầu và cạnh kết thúc của mặt lưới không trùng nhau.

Do yêu cầu của sản xuất, yêu cầu mẫu mã sản phẩm v.v. mà phải sử dụng các mặt 3D phức tạp (*vỏ xe hơi, máy bay, tàu thủy v.v.*). Để làm được điều đó bằng VB&VBA ta dùng lệnh sau:

Lệnh **Add3Dmesh** tạo mặt lưới đa giác bằng cách cho số đỉnh theo các hướng M, N và toạ độ mỗi đỉnh. Số đỉnh nằm trong khoảng từ 2-256 như hình 13.9. minh hoạ.




***Hình 13.9.*** Mặt lưới tự do

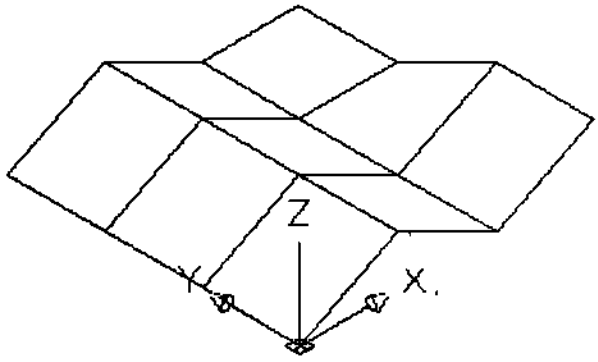
**Cú pháp:**

Object = space.Add3Dmesh(*M, N, PointsMatrix*)

**Trong đó:**

| | |
|---|---|
| Object | Đối tượng mặt lưới đa giác mới được khởi tạo. |
| space | Không gian vẽ hoặc không gian giấy. |
| *M* | Số đỉnh theo hướng M. |
| *N* | Số đỉnh theo hướng N. |
| *PointsMatrix* | Số điểm trong ma trận M. N có kiểu dữ liệu Double. |

Dưới đây là đoạn mã chương trình sau sẽ tạo mặt lưới có kích thước là 4x4 trong không gian vẽ như hình 13.10.



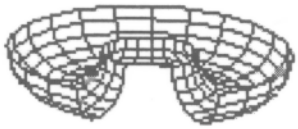

***Hình 13.10.*** Tạo lưới.

```
; Tên file VBA_Add3DMesh.dvb.
 Sub VBA_Add3DMesh()
   Dim meshObj As AcadPolygonMesh
```

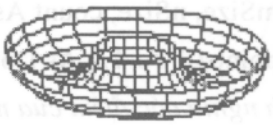
```
    Dim mSize, nSize, count As Integer
    Dim points(0 To 47) As Double
    ' Định nghĩa các điểm của ma trận.
    points(0) = 0: points(1) = 0: points(2) = 0
    points(3) = 2: points(4) = 0: points(5) = 1
    points(6) = 4: points(7) = 0: points(8) = 0
    points(9) = 6: points(10) = 0: points(11) = 1
    points(12) = 0: points(13) = 2: points(14) = 0
    points(15) = 2: points(16) = 2: points(17) = 1
    points(18) = 4: points(19) = 2: points(20) = 0
    points(21) = 6: points(22) = 2: points(23) = 1
    points(24) = 0: points(25) = 4: points(26) = 0
    points(27) = 2: points(28) = 4: points(29) = 1
    points(30) = 4: points(31) = 4: points(32) = 0
    points(33) = 6: points(34) = 4: points(35) = 0
    points(36) = 0: points(37) = 6: points(38) = 0
    points(39) = 2: points(40) = 6: points(41) = 1
    points(42) = 4: points(43) = 6: points(44) = 0
    points(45) = 6: points(46) = 6: points(47) = 0
    mSize = 4: nSize = 4
    ' Tạo mặt 3Dmesh trong không gian vẽ.
    Set meshObj = ThisDrawing.ModelSpace.Add3DMesh(mSize, nSize, points)
    ' Thay đổi hướng nhìn.
    Dim NewDirection(0 To 2) As Double
    NewDirection(0) = -1: NewDirection(1) = -1: NewDirection(2) = 1
    ThisDrawing.ActiveViewport.Direction = NewDirection
    ThisDrawing.ActiveViewport = ThisDrawing.ActiveViewport
    ZoomAll
End Sub
; Kết thúc.
```

***Chú ý:***

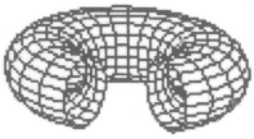
Một bề mặt được tạo ra chúng ta sử dụng thuộc tính **MClose** và **NClose** để đóng mặt cong lưới như hình 13.11.



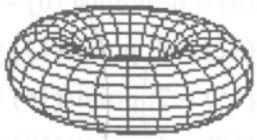
M : Mở
N: Mở



M: Đóng
N: Mở



M: Mở
N: Đóng



M: Đóng
N: Đóng

***Hình 13.11.*** Các trường hợp của mặt ứng với M và N khác nhau.

### 13.5.1. Thuộc tính MClose

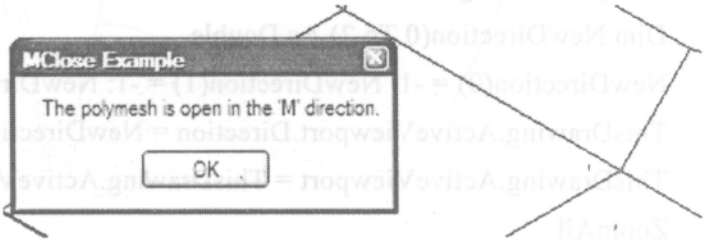
**Cú pháp:**

Object.MClose

**Trong đó:**

| | |
|---|---|
| Object | Mặt cong lưới. |
| MClose | Thuộc tính MClose có kiểu Boolean với hai giá trị như sau:<br>+ TRUE : Đóng mặt theo hướng M.<br>+ FALSE: Mở mặt theo hướng M. |

Dưới đây là đoạn mã chương trình tạo mặt lưới 4x4 trong không gian vẽ và đóng mặt theo hướng M.




***Hình 13.12.*** Hộp thoại hiển thị thông báo.

```
; Tên file VBA_MClose.dvb.
Sub VBA_MClose()
    Dim meshObj As AcadPolygonMesh
    Dim mSize, nSize, count As Integer
```

```
    Dim points(0 To 47) As Double
    ' Định nghĩa các điểm của ma trận.
    points(0) = 2: points(1) = 2: points(2) = 1
    points(3) = 2.5: points(4) = 2: points(5) = 0
    points(6) = 5: points(7) = 2: points(8) = 0
    points(9) = 5.5: points(10) = 2: points(11) = 1
    points(12) = 2: points(13) = 4: points(14) = 0.5
    points(15) = 2.5: points(16) = 4: points(17) = 0
    points(18) = 5: points(19) = 4: points(20) = 0
    points(21) = 5.5: points(22) = 4: points(23) = 0.5
    points(24) = 2: points(25) = 6: points(26) = 0.5
    points(27) = 2.5: points(28) = 6: points(29) = 0
    points(30) = 5: points(31) = 6: points(32) = 0
    points(33) = 5.5: points(34) = 6: points(35) = 0.5
    points(36) = 2: points(37) = 8: points(38) = 1
    points(39) = 2.5: points(40) = 8: points(41) = 0
    points(42) = 5: points(43) = 8: points(44) = 0
    points(45) = 5.5: points(46) = 8: points(47) = 1
    mSize = 4: nSize = 4
    ' Tạo mặt 3Dmesh trong không gian vẽ.
    Set meshObj = ThisDrawing.ModelSpace.Add3DMesh(mSize, nSize, points)
    ' Thay đổi hướng quan sát.
    Dim NewDirection(0 To 2) As Double
    NewDirection(0) = -1: NewDirection(1) = -1: NewDirection(2) = 1
    ThisDrawing.ActiveViewport.Direction = NewDirection
    ThisDrawing.ActiveViewport = ThisDrawing.ActiveViewport
    ZoomAll
    ' Đóng mặt phẳng theo hướng M.
    MsgBox "The polymesh is " & IIf(meshObj.MClose, "closed", "open") & " in
    the 'M' direction.", , "MClose Example"
    meshObj.MClose = True
    ThisDrawing.Regen acActiveViewport
    MsgBox "The polymesh is " & IIf(meshObj.MClose, "closed", "open") & " in
    the 'M' direction.", , "MClose Example"
  End Sub
; Kết thúc.
```

### 13.5.2. Thuộc tính NClose

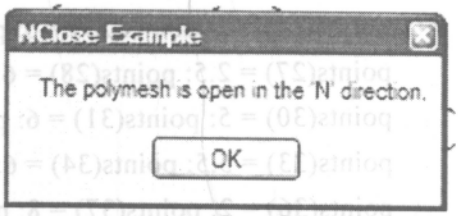
**Cú pháp:**

Object.NClose

**Trong đó:**

| | |
|---|---|
| Object | Bề mặt lưới. |
| NClose | Thuộc tính NClose có kiểu Boolean với hai giá trị như sau:<br>+ TRUE : Đóng mặt theo hướng N.<br>+ FALSE: Mở mặt theo hướng N. |

Đoạn mã chương trình sau tạo mặt lưới 4x4 trong không gian vẽ và đóng mặt theo hướng N như hình 13.13.

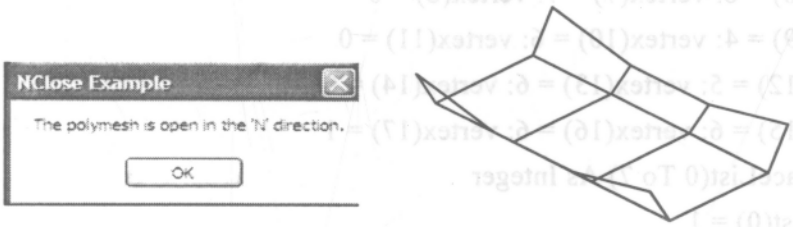



*Hình 13.13*. Hộp thoại hiển thị thông báo

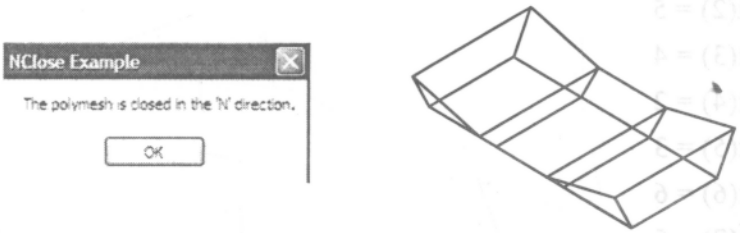
```
; Tên file VBA_NClose.dvb.
Sub VBA_NClose()
    Dim meshObj As AcadPolygonMesh
    Dim mSize, nSize, count As Integer
    Dim points(0 To 47) As Double
    ' Định nghĩa các điểm của ma trận.
    points(0) = 2: points(1) = 2: points(2) = 1
    points(3) = 2.5: points(4) = 2: points(5) = 0
    points(6) = 5: points(7) = 2: points(8) = 0
    points(9) = 5.5: points(10) = 2: points(11) = 1
    points(12) = 2: points(13) = 4: points(14) = 0.5
    points(15) = 2.5: points(16) = 4: points(17) = 0
    points(18) = 5: points(19) = 4: points(20) = 0
    points(21) = 5.5: points(22) = 4: points(23) = 0.5
    points(24) = 2: points(25) = 6: points(26) = 0.5
    points(27) = 2.5: points(28) = 6: points(29) = 0
    points(30) = 5: points(31) = 6: points(32) = 0
    points(33) = 5.5: points(34) = 6: points(35) = 0.5
    points(36) = 2: points(37) = 8: points(38) = 1
```

```
points(39) = 2.5: points(40) = 8: points(41) = 0
points(42) = 5: points(43) = 8: points(44) = 0
points(45) = 5.5: points(46) = 8: points(47) = 1
mSize = 4: nSize = 4
' Tạo mặt 3Dmesh trong không gian vẽ.
Set meshObj = ThisDrawing.ModelSpace.Add3DMesh(mSize, nSize, points)
' Thay đổi hướng quan sát đối tượng.
Dim NewDirection(0 To 2) As Double
NewDirection(0) = -1: NewDirection(1) = -1: NewDirection(2) = 1
ThisDrawing.ActiveViewport.Direction = NewDirection
ThisDrawing.ActiveViewport = ThisDrawing.ActiveViewport
ZoomAll
' Đóng mặt phẳng theo hướng N.
MsgBox "The polymesh is " & IIf(meshObj.NClose, "closed", "open") & " in the
'N' direction.", , "NClose Example"
meshObj.NClose = True
ThisDrawing.Regen acActiveViewport
MsgBox "The polymesh is " & IIf(meshObj.NClose, "closed", "open") & " in the
'N' direction.", , "NClose Example"
End Sub
```

; Kết thúc chương trình ta thu được các kết quả như hình 13.14.




a) Mở mặt theo hướng N



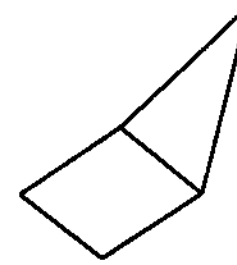

b) Đóng mặt theo hướng N

***Hình 13.14***. Kết quả chạy chương trình.

## 13.6. TẠO LƯỚI NHIỀU MẶT ĐA GIÁC

Sử dụng lệnh **AddPolyfaceMesh** để tạo lưới nhiều mặt đa giác, mỗi mặt có một số các đỉnh. Việc tạo lưới nhiều mặt đa giác cũng tương tự như tạo một mặt lưới chữ nhật, ta phải xác định toạ độ cho tất cả các đỉnh, sau đó xác định mặt bằng cách nhập vào chỉ số đỉnh cho các đỉnh thuộc mặt đó. Sau khi tạo lưới nhiều mặt đa giác có thể chỉ định các cạnh không thể nhìn thấy, chọn các cạnh theo layer hoặc tô màu cho các cạnh. Để tạo một cạnh không thể nhìn thấy được, ta có thể nhập vào chỉ số đỉnh thuộc cạnh đó giá trị âm.

Đoạn mã chương trình dưới đây sẽ tạo một đối tượng lưới nhiều mặt đa giác trong không gian vẽ 3D. Hướng nhìn của khung quan sát hiện hành được điều chỉnh để mặt lưới được quan sát dễ dàng hơn như hình 13.15.



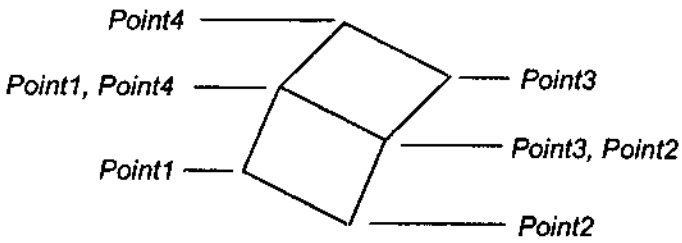
*Hình 13.15.*

```
; Tên file VBA_CreatePolyfaceMesh.dvb.
Sub VBA_CreatePolyfaceMesh()
   Dim vertex(0 To 17) As Double
   vertex(0) = 4: vertex(1) = 7: vertex(2) = 0
   vertex(3) = 5: vertex(4) = 7: vertex(5) = 0
   vertex(6) = 6: vertex(7) = 7: vertex(8) = 0
   vertex(9) = 4: vertex(10) = 6: vertex(11) = 0
   vertex(12) = 5: vertex(13) = 6: vertex(14) = 0
   vertex(15) = 6: vertex(16) = 6: vertex(17) = 1
    Dim FaceList(0 To 7) As Integer
    FaceList(0) = 1
    FaceList(1) = 2
    FaceList(2) = 5
    FaceList(3) = 4
    FaceList(4) = 2
    FaceList(5) = 3
    FaceList(6) = 6
    FaceList(7) = 5
    Dim polyfaceMeshObj As AcadPolyfaceMesh
    Set polyfaceMeshObj = ModelSpace.AddPolyfaceMesh _ (vertex, FaceList)
```

```
    Dim NewDirection(0 To 2) As Double
    NewDirection(0) = -1
    NewDirection(1) = -1
    NewDirection(2) = 1
    ThisDrawing.ActiveViewport.direction = NewDirection
    ThisDrawing.ActiveViewport = ThisDrawing.ActiveViewport
    ZoomAll
End Sub
; Kết thúc.
```

### 13.7. MẶT PHẲNG 3D

Lệnh **Add3DFace** dùng để tạo các mặt 3D có bốn hoặc ba cạnh như hình 13.16. Mỗi mặt được tạo bởi đối tượng 3D là một đối tượng đơn.




***Hình 13.16.*** Cách tạo mặt phẳng bằng lệnh 3DFace.

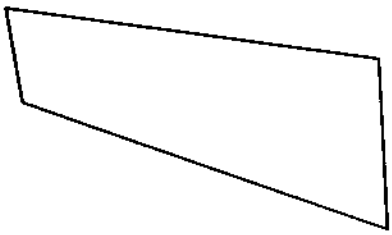
**Cú pháp:**

Object= space.Add3DFace(*Point1, Point2, Point3, Point4*)

**Trong đó:**

| | |
|---|---|
| Object | Đối tượng 3DFace mới được khởi tạo. |
| space | Không gian vẽ hoặc không gian giấy. |
| *Point1* | Điểm thứ nhất của mặt phẳng gồm ba phần tử (x, y, z) có kiểu dữ liệu Double. |
| *Point2* | Điểm thứ hai của mặt phẳng gồm ba phần tử (x, y, z) và có kiểu dữ liệu Double. |
| *Point3* | Điểm thứ ba của mặt phẳng gồm ba phần tử (x, y, z) có kiểu dữ liệu Double. |
| *Point4* | Điểm thứ tư của mặt phẳng gồm ba phần tử (x, y, z) có kiểu dữ liệu Double. |

Đoạn mã chương trình sau tạo một mặt phẳng 3DFace trong không gian vẽ như hình 13.17.



*Hình 13.17*

```
; Tên file VBA_Add3DFace.dvb.
 Sub VBA_Add3DFace()
    Dim faceObj As Acad3DFace
    Dim point1(0 To 2) As Double
    Dim point2(0 To 2) As Double
    Dim point3(0 To 2) As Double
    Dim point4(0 To 2) As Double
    ' Định nghĩa các điểm của mặt phẳng.
    point1(0) = 0#: point1(1) = 0#: point1(2) = 0#
    point2(0) = 5#: point2(1) = 0#: point2(2) = 1#
    point3(0) = 1#: point3(1) = 10#: point3(2) = 0#
    point4(0) = 5#: point4(1) = 5#: point4(2) = 1#
    ' Tạo mặt phẳng 3D trong không gian vẽ.
    Set faceObj = ThisDrawing.ModelSpace.Add3DFace(point1, point2, point3, point4)
    ZoomAll
 End Sub
; Kết thúc.
```

## 13.8. TẠO MÔ HÌNH KHỐI RẮN *(Solid model)*

Mức độ cao nhất trong mô hình hình học là mô hình 3D, mô hình vật thể rắn *(models solid)*. Nó cũng giống như mô hình khung dây hay mô hình bề mặt là các đường khuất đã được xoá. Tuy nhiên sự khác nhau là ở chỗ: mô hình khung dây và mô hình bề mặt chỉ dừng lại ở chỗ mô tả vật thể như là một mô hình toán học mà không thể hiện đặc tính khối lượng của chúng. Mô hình vật thể rắn đã khắc phục được thiếu sót này. Tính chất khối lượng của mô hình chi tiết rất quan trọng để dự đoán các tính chất tượng tự như trọng lượng, sự ổn định, mô men quán tính và thể tích... của sản phẩm cuối cùng chứa chi tiết. Một điểm mạnh nữa là ta có thể dễ dàng cắt qua mô hình vật rắn tới những chi tiết phía trong.

Mô hình vật rắn thường được ghi lại trong máy tính dưới dạng toán học như các thể tích viền bởi bề mặt. Kết quả là nhờ chúng có thể tính toán được những tính chất khối lượng của các chi tiết thường xuyên được sử dụng cho các công việc phân tích kĩ thuật như phân tích trạng thái vật lý bằng phương pháp phần tử hữu hạn và nghiên cứu động học để kiểm tra giao thoa. Ví dụ đối với hình trụ

Mô hình khung dây của hình trụ được định nghĩa trong máy tính như là hai đường tròn nối với nhau bằng hai đoạn thẳng trong khi đó mô hình vật thể rắn của hình trụ được biểu diễn như một đối tượng 3D chứa thể tích.

Việc sử dụng mô hình vật thể rắn có thể tạo các hình khối và liên kết chúng thành một vật thể phức tạp nhanh chóng hơn nhiều so với mô hình khung dây hay bề mặt. Tuy nhiên mô hình vật thể rắn cần tốn nhiều bộ nhớ, chúng đòi hỏi quá trình thao tác rộng đối với cấu trúc dữ liệu và liên kết toán học phức tạp và chỉ thích hợp cho bề mặt không quá phức tạp.

### 13.8.1. Khối hình chữ nhật

Lệnh **AddBox** được sử dụng để tạo khối hình lập phương hay khối hình hộp chữ nhật.

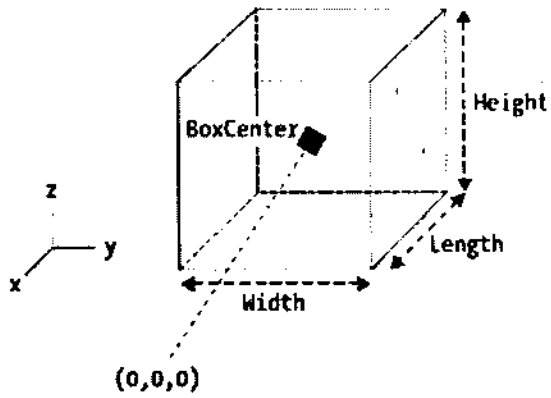
**Cú pháp:**

Object = space.AddBox*(Origin, Length, Width, Height)*

**Trong đó:**

| | |
|---|---|
| Object | Đối tượng hình hộp lập phương hoặc hình hộp chữ nhật. |
| Space | Không gian vẽ hoặc không gian giấy. |
| *Origin* | Tâm của hình hộp chữ nhật gồm có ba phần tử (x, y, z). |
| *Length* | Chiều dài hình hộp có kiểu dữ liệu là Double *(phải là kiểu số).* |
| *Width* | Chiều rộng hình hộp có kiểu dữ liệu là Double *(phải là kiểu số).* |
| *Height* | Chiều cao hình hộp có kiểu dữ liệu là Double *(phải là kiểu số).* |

Đoạn mã chương trình dưới đây sẽ tạo một hình hộp chữ nhật trong không gian vẽ có tâm (5, 5, 0), có kích thước 5mm x 7mm x 10mm như hình 13.18.




***Hình 13.18.*** Khối hình chữ nhật.

```
; Tên file VBA_AddBox.dvb.
Sub VBA_AddBox()
  ' Khai báo đối tượng hình hộp boxObj có  kiểu Acad3Dsolid trong VBA
  Dim boxObj As Acad3Dsolid
  ' Khai báo các thông số đầu vào
  Dim length As Double, width As Double, height As Double
  ' Khai báo tâm hình hộp.
  Dim center(0 To 2) As Double
  ' Định nghĩa hình hộp chữ nhật.
  center(0) = 5#: center(1) = 5#: center(2) = 0
  length = 5#: width = 7: height = 10#
  'Khởi tạo hình hộp chữ nhật trong không gian vẽ.
  Set boxObj = ThisDrawing.ModelSpace.AddBox(center, length, width, height)
  'Thay đổi hướng quan sát (khung nhìn)
  Dim NewDirection(0 To 2) As Double
  NewDirection(0) = -1: NewDirection(1) = -1: NewDirection(2) = 1
  'Gán hệ toạ độ hiện hành trong không gian vẽ.
  ThisDrawing.ActiveViewport.direction = NewDirection
  ThisDrawing.ActiveViewport = ThisDrawing.ActiveViewport
End Sub
; Kết thúc.
```

### 13.8.2. Khối hình nón

Lệnh **AddCone** sử dụng để tạo các khối nón.

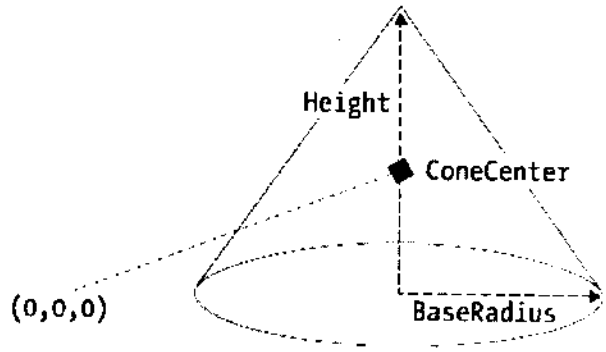
**Cú pháp:**

Object = Space.AddCone*(Center, BaseRadius, Height)*

**Trong đó:**

| | |
|---|---|
| Object | Đối tượng hình nón. |
| Space | Không gian vẽ hoặc không gian giấy. |
| *Center* | Tâm của hình nón gồm ba phần tử (x, y, z) có kiểu dữ liệu là Double. |
| *BaseRadius* | Bán kính của hình nón cơ sở có kiểu dữ liệu là Double *(kiểu số)*. |
| *Height* | Chiều cao của hình nón có dữ liệu là Double *(kiểu số)*. |

Đoạn mã chương trình dưới đây sẽ tạo ra một hình nón trong không gian vẽ với tâm có toạ độ O(0, 0, 0), chiều cao H=50 mm; bán kính r = 5 mm như hình 13.19.




***Hình 13.19.*** Khối hình nón.

```
; Tên file VBA_AddCone.dvb.
 Sub VBA_AddCone()
   ' Khai báo đối tượng hình nón có kiểu Acad3Dsolid trong VBA
   Dim coneObj As Acad3Dsolid
   'Khai báo các biến (bán kính, tâm, chiều cao của hình nón).
   Dim radius As Double
   Dim center(0 To 2) As Double
   Dim height As Double
   'Định nghĩa khối nón.
   center(0) = 0#: center(1) = 0#: center(2) = 0#
   radius = 5#
   height = 20#
   ' Khởi tạo hình nón trong không gian vẽ.
   Set coneObj = ThisDrawing.ModelSpace.AddCone(center, radius, height)
   'Thay đổi hướng quan sát.
    Dim NewDirection(0 To 2) As Double
   NewDirection(0) = -1: NewDirection(1) = -1: NewDirection(2) = 1
   'Gán hệ toạ độ hiện hành.
  ThisDrawing.ActiveViewport.direction = NewDirection
   ThisDrawing.ActiveViewport = ThisDrawing.ActiveViewport
  ZoomAll
 End Sub
; Kết thúc.
```

### 13.8.3. Khối hình nêm

Để tạo ra khối hình nêm ta sử dụng lệnh sau: **AddWedge**.

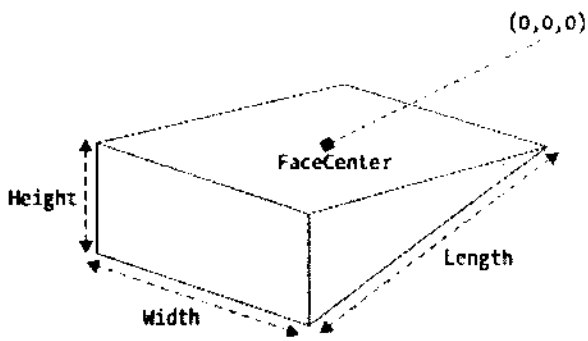
**Cú pháp:**

Object = Space.AddWedge*(Center, Length, Width, Height)*

**Trong đó :**

| | |
|---|---|
| Object | Đối tượng hình nêm. |
| Space | Không gian vẽ hoặc không gian giấy. |
| *Center* | Tâm của hình nêm gồm ba phần tử có kiểu dữ liệu là Double. |
| *Length* | Chiều dài của hình nêm theo phương X có kiểu dữ liệu là Double *(phải là kiểu số).* |
| *Width* | Chiều rộng của hình nêm theo phương Y có kiểu dữ liệu Double *(phải là kiểu số).* |
| *Height* | Chiều cao của hình nêm theo phương Z có kiểu dữ liệu Double *(phải là kiểu số).* |

Đoạn mã chương trình dưới đây sẽ tạo một hình nêm trong không gian vẽ với tâm có toạ độ O(5, 5, 0), chiều cao H = 20 mm, chiều rộng W = 15 mm và chiều dài L = 10 mm như hình 13.20.




***Hình 13.20.*** Khối hình nêm.

```
; Tên file VBA_AddWedge.dvb.
  Sub VBA_AddWedge()
    'Khai báo đối tượng hình nêm cú kiểu Acad3Dsolid trong VBA.
    Dim wedgeObj As Acad3Dsolid
    'Khai báo các biến (tâm, chiều dài, chiều rộng, chiều cao của hình nêm) có kiểu
     dữ liệu Double.
    Dim center(0 To 2) As Double
    Dim length As Double
    Dim width As Double
    Dim height As Double
```

```
    'Cho trước các thông số (tâm, chiều dài, chiều rộng, chiều cao của hình nêm).
    center(0) = 5#: center(1) = 5#: center(2) = 0#
    length = 10#: width = 15#: height = 20#
  'Khởi tạo hình nêm trong không gian vẽ.
  Set wedgeObj = ThisDrawing.ModelSpace.AddWedge(center, length, width,
  height)
 ZoomAll
 End Sub
; Kết thúc.
```

### 13.8.4. Khối hình trụ

Để tạo khối đặc hình trụ ta sử dụng lênh **AddCylinder** .

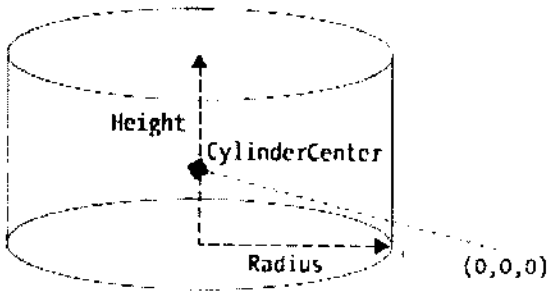
**Cú pháp:**

Object = Space.AddCylinder*(Center, Radius, Height)*

**Trong đó:**

| | |
|---|---|
| Object | Đối tượng hình trụ. |
| Space | Không gian vẽ hoặc không gian giấy. |
| *Center* | Tâm hình trụ gồm có ba phần tử (x, y, z) có kiểu dữ liệu Double. |
| *Radius* | Bán kính hình trụ có kiểu dữ liệu là Double *(phải là dạng số)* |
| *Height* | Chiều cao hình trụ có kiểu dữ liệu Double *(phải là dạng số)* |

Đoạn mã chương trình dưới đây sẽ tạo ra một hình trụ trong không gian vẽ có tâm: O(0, 0, 0), chiều cao: H = 20 mm và bán kính: r = 5 mm như hình 13.21.




***Hình 13.21.*** Khối trụ.

```
; Tên file VBA_ AddCylinder.dvb.
 Sub  VBA_ AddCylinder()
    'Khai báo đối tượng hình trụ có kiểu Acad3Dsolid trong VBA.
    Dim cylinderObj As Acad3Dsolid
    'Khai báo các biến (bán kính, tâm, chiều cao của hình trụ) có kiểu dữ liệu Double.
```

```
    Dim radius As Double
    Dim center(0 To 2) As Doubl e
    Dim height As Double
    'Định nghĩa hình trụ (tâm, bán kính, chiều cao).
    center(0) = 0#: center(1) = 0#: center(2) = 0#
    radius = 5#
    Height = 20#
    'Khởi tạo khối trụ trong không gian vẽ với các thông số cho trước.
    Set cylinderObj = ThisDrawing.ModelSpace.AddCylinder(center, radius, height)
    ZoomAll
  End Sub
; Kết thúc.
```

### 13.8.5. Khối cầu

Lệnh **AddSphere** dùng để tạo khối cầu.

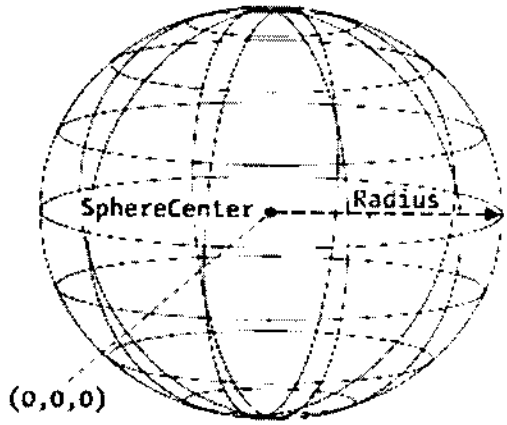
**Cú pháp:**

Object = space.AddSphere *( Center, Radius )*

**Trong đó:**

| | |
|---|---|
| Object | Đối tượng khối cầu. |
| space | Không gian vẽ hoặc không gian giấy. |
| *Center* | Tâm của hình cầu gồm có ba phần tử (x, y, z) có kiểu dữ liệu là Double. |
| *Radius* | Bán kính của hình cầu có kiểu dữ liệu Double *(phải là kiểu số).* |

Đoạn mã chương trình dưới đây sẽ tạo ra một hình cầu trong không gian vẽ có tâm O(5, 5, 0) và bán kính r = 5 mm như hình 13.22.




***Hình 13.22.*** Khối cầu.

```
; Tên file VBA_AddSphere.dvb.
 Sub VBA_AddSphere()
   'Khai báo đối tượng hình cầu có kiểu Acad3Dsolid trong VBA.
   Dim sphereObj As Acad3Dsolid
   'Khai báo các biến (bán kính, tâm ) có kiểu dữ liệu Double.
   Dim centerPoint(0 To 2) As Double
   Dim radius As Double
   'Định nghĩa khối cầu.
   centerPoint(0) = 5#: centerPoint(1) = 5#: centerPoint(2) = 0#
   radius = 5#
   'Khởi tạo khối hình cầu trong không gian vẽ.
   Set sphereObj = ThisDrawing.ModelSpace.AddSphere(centerPoint, radius)
   'Thay đổi khung nhìn của bản vẽ để quan sát đối tượng
   Dim NewDirection(0 To 2) As Double
   NewDirection(0) = -1: NewDirection(1) = -1: NewDirection(2) = 1
   'Bản vẽ thể hiện theo hướng nhìn mới.
   ThisDrawing.ActiveViewport.direction = NewDirection
   ThisDrawing.ActiveViewport = ThisDrawing.ActiveViewport
   ZoomAll
 End Sub
; Kết thúc.
```

### 13.8.6. Khối xuyến

Lệnh **AddTorus** dùng để tạo khối hình xuyến.

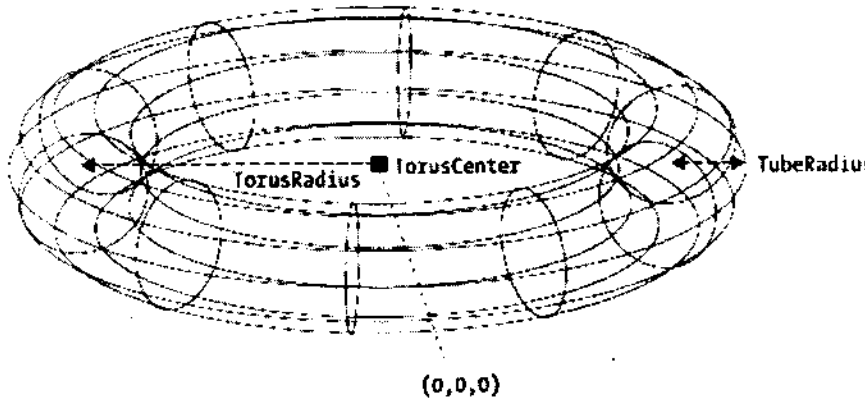
**Cú pháp:**

Object = space.AddTorus(*Center, TorusRadius, TubeRadius*)

**Trong đó:**

| | |
|---|---|
| Object | Đối tượng hình xuyến. |
| space | Không gian vẽ hoặc không gian giấy. |
| *Center* | Tâm của hình xuyến gồm có ba phần tử (x, y, z) có kiểu dữ liệu là Double. |
| *TorusRadius* | Khoảng cách từ tâm của khối xuyến đến tâm của vòng trong có kiểu dữ liệu Double *(phải là kiểu số).* |
| *TubeRadius* | Bán kính của vòng tròn tiết diện xuyến có kiểu dữ liệu là Double *(phải là kiểu số).* |

Đoạn mã chương trình dưới đây sẽ tạo một hình xuyến trong không gian vẽ với tâm có toạ độ O(5, 5, 0) và bán kính tiết diện r =5 mm, bán kính xuyến R = 15mm như hình 13.23.




***Hình 13.23.*** Mô hình khối xuyến.

```
; Tên file VBA_AddTorus.dvb.
 Sub VBA_AddTorus()
   'Khai báo đối tượng hình xuyến có kiểu Acad3Dsolid trong VBA.
   Dim torusObj As Acad3Dsolid
   'Khai báo các biến của các thông số đầu vào.
   Dim centerPoint(0 To 2) As Double
   Dim torusRadius As Double
   Dim tubeRadius As Double
   'Định nghĩa khối xuến.
   centerPoint(0) = 5: centerPoint(1) = 5: centerPoint(2) = 0
    torusRadius = 15
    tubeRadius = 5
    'Khởi tạo khối hình xuyến trong không gian vẽ.
    Set torusObj = ThisDrawing.ModelSpace.AddTorus(centerPoint, torusRadius,
    tubeRadius)
    ZoomAll
  End Sub
; Kết thúc.
```

### 13.8.7. Tạo khối rắn theo phương pháp đùn

Lệnh **AddExtrudedSolid** dùng để tạo khối rắn theo phương pháp đùn theo phương vuông góc với mặt chứa biên dạng đùn.

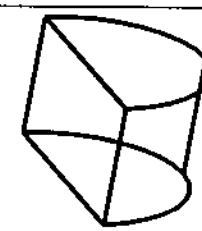
**Cú pháp:**

Object = space.AddExtrudedSolid*(Profile, Height, TaperAngle)*

**Trong đó:**

| | |
|---|---|
| Object | Đối tượng 3D được tạo thành từ phương pháp kéo dài. |
| space | Không gian vẽ hoặc không gian giấy. |
| *Profile* | Miền đối tượng. |
| *Height* | Chiều cao theo phương Z có kiểu dữ liệu là Double *(kiểu số)*. |
| *TaperAngle* | Góc nghiêng *(-90 độ đến 90 độ)* có kiểu dữ liệu là Double, đơn vị là radians. |

Đoạn mã chương trình dưới đây tạo một hình khối bằng phương pháp đùn theo phương Z, biên dạng được tạo bởi một cung tròn và một đoạn thẳng như hình 13.24.



***Hình 13.24.*** Kéo dài đối tượng.

```
; Tên file VBA_AddExtrudedSolid.dvb.
 Sub VBA_AddExtrudedSolid()
    Dim curves(0 To 1) As AcadEntity
    'Tạo miền cơ sở  (từ cung tròn và đường thẳng)
    Dim centerPoint(0 To 2) As Double
    Dim radius As Double
    Dim startAngle As Double
    Dim endAngle As Double
    ' Định nghĩa đối tượng (tâm, bán kính).
    centerPoint(0) = 5#: centerPoint(1) = 3#: centerPoint(2) = 0#
     radius = 2#
    startAngle = 0
    endAngle = 3.141592
 Set curves(0) = ThisDrawing.ModelSpace.AddArc(centerPoint, radius,  startAngle,
 endAngle)
```

```
Set curves(1) = ThisDrawing.ModelSpace.AddLine(curves(0).startPoint,
curves(0).endPoint)
    'Khởi tạo miền biên dạng trong mô hình 2D.
    Dim regionObj As Variant
    regionObj = ThisDrawing.ModelSpace.AddRegion(curves)
    'Khai báo chiều cao cần kéo dài
    Dim height As Double
    Dim taperAngle As Double
    height = 3
    taperAngle = 0
    'Tạo hình khối.
    Dim solidObj As Acad3Dsolid
Set solidObj = ThisDrawing.ModelSpace.AddExtrudedSolid(regionObj(0), height,
taperAngle)
    ' Thay đổi hướng quan sát đối tượng.
    Dim NewDirection(0 To 2) As Double
    NewDirection(0) = -1: NewDirection(1) = -1: NewDirection(2) = 1
    ThisDrawing.ActiveViewport.direction = NewDirection
    ThisDrawing.ActiveViewport = ThisDrawing.ActiveViewport
End Sub
; Kết thúc.
```

### 13.8.8. Tạo khối rắn đùn theo đường dẫn

Lệnh **AddExtrudedSolidAlongPath** dùng để đùn một biên dạng theo đường dẫn cho trước.

**Cú pháp:**

Object = space.AddExtrudedSolidAlongPath(*Profile, Path*)

**Trong đó:**

| | |
|---|---|
| Object | Đối tượng 3D mới được khởi tạo. |
| space | Không gian vẽ hoặc không gian giấy. |
| *Profile* | Miền biên dạng cần đùn. |
| *Path* | Đối tượng đường dẫn *(đường polyline, tròn, ellisp, spline, cung tròn).* |

Đoạn mã chương trình sau sẽ đùn chi tiết có biên dạng là một nửa đường tròn theo đường dẫn là một đường spline như hình 13.25.

*Hình 13.25*

```
; Tên file VBA_AddExtrudedSolidAlongPath.dvb.
 Sub VBA_AddExtrudedSolidAlongPath()
    Dim curves(0 To 1) As AcadEntity
    ' Định nghĩa biên dạng.
    Dim centerPoint(0 To 2) As Double
    Dim radius As Double
    Dim startAngle As Double
    Dim endAngle As Double
    centerPoint(0) = 5#: centerPoint(1) = 3#: centerPoint(2) = 0#
    radius = 2#
    startAngle = 0
    endAngle = 3.141592
    Set curves(0) = ThisDrawing.ModelSpace.AddArc(centerPoint, radius,
     startAngle, endAngle)
    Set curves(1) = ThisDrawing.ModelSpace.AddLine(curves(0).StartPoint,
     curves(0).EndPoint)
    ' Tạo miền biên dạng.
    Dim regionObj As Variant
    regionObj = ThisDrawing.ModelSpace.AddRegion(curves)
    ' Định nghĩa đường dẫn là đường Spline.
```

```
Dim splineObj As AcadSpline
Dim startTan(0 To 2) As Double
Dim endTan(0 To 2) As Double
Dim fitPoints(0 To 8) As Double
' Định nghĩa đường Spline.
startTan(0) = 10: startTan(1) = 10: startTan(2) = 10
endTan(0) = 10: endTan(1) = 10: endTan(2) = 10
fitPoints(0) = 0: fitPoints(1) = 10: fitPoints(2) = 10
fitPoints(0) = 10: fitPoints(1) = 10: fitPoints(2) = 10
fitPoints(0) = 15: fitPoints(1) = 10: fitPoints(2) = 10
Set splineObj = ThisDrawing.ModelSpace.AddSpline(fitPoints, startTan, endTan)
' Tạo đối tượng mô hình.
Dim solidObj As Acad3DSolid
Set solidObj =
ThisDrawing.ModelSpace.AddExtrudedSolidAlongPath(regionObj(0), splineObj)
ZoomAll
End Sub
; Kết thúc.
```

### 13.8.9. Tạo khối bằng phương pháp quay một biên dạng quay quanh một trục

Lệnh **AddRevolvedSolid** dùng để tạo khối 3D theo phương pháp quay một biên dạng quay quanh một trục xác định.

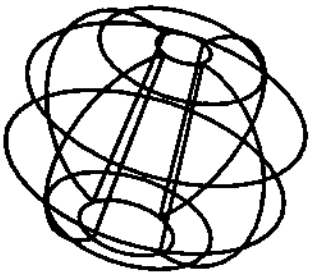
**Cú pháp:**

Object = Space.AddRevolvedSolid*(Profile, AxisPoint, AxisDir, Angle)*

**Trong đó:**

| | |
|---|---|
| Object | Đối tượng 3D được tạo thành từ phương pháp quay. |
| Space | Không gian vẽ hoặc không gian giấy. |
| *Profile* | Miền biên dạng đối tượng. |
| *AxisPoint* | Là điểm đầu tiên của trục xoay có kiểu dữ liệu là Double. |
| *AxisDir* | Là vector pháp của trục xoay có kiểu dữ liệu là Double. |
| *Angle* | Biến góc quay có kiểu dữ liệu Double *(đơn vị radians)*. |

Dưới đây là đoạn mã chương trình tạo ra một khối 3D theo phương pháp quay quanh một trục với một góc $360^0$ như hình 13.26.



***Hình 13.26.*** Tạo khối rắn theo phương pháp quay quanh một trục 1 góc $360^0$.

```
; Tên file VBA_AddRevolvedSolid.dvb.
 Sub VBA_AddRevolvedSolid()
    Dim curves(0 To 1) As AcadEntity
    ' Khai báo biên dạng đối tượng.
    Dim centerPoint(0 To 2) As Double
    Dim radius As Double
    Dim startAngle As Double
    Dim endAngle As Double
    'Định nghĩa biên dạng.
    centerPoint(0) = 5#: centerPoint(1) = 3#: centerPoint(2) = 0#
    radius = 2#
    startAngle = 0
    endAngle = 3.141592
    Set curves(0) = ThisDrawing.ModelSpace.AddArc(centerPoint, radius,
    startAngle,  endAngle)
    Set curves(1) = ThisDrawing.ModelSpace.AddLine(curves(0).startPoint,
    curves(0).endPoint)
    'Khởi tạo miền từ cung tròn và đường thẳng.
    Dim regionObj As Variant
    regionObj = ThisDrawing.ModelSpace.AddRegion(curves)
    regionObj(0).Color = acCyan
    ZoomAll
    MsgBox "Revolve the region to create the solid.", , "AddRevolvedSolid
    Example"
    'Khai báo trục quay.
    Dim axisPt(0 To 2) As Double
```

```
    Dim axisDir(0 To 2) As Double
    Dim angle As Double
    axisPt(0) = 7: axisPt(1) = 2.5: axisPt(2) = 0
    axisDir(0) = 11: axisDir(1) = 1: axisDir(2) = 3
    angle = 6.28
    'Khởi tạo 3D.
    Dim solidObj As Acad3Dsolid
    'Khởi tạo đối tượng theo phương pháp quay tròn.
    Set solidObj = ThisDrawing.ModelSpace.AddRevolvedSolid(regionObj(0),
     axisPt, axisDir, angle)
    solidObj.Color = acRed
    ZoomAll
    'Thay đổi hướng quan sát đối tượng.
    Dim NewDirection(0 To 2) As Double
    NewDirection(0) = -1: NewDirection(1) = -1: NewDirection(2) = 1
    ThisDrawing.ActiveViewport.direction = NewDirection
    ThisDrawing.ActiveViewport = ThisDrawing.ActiveViewport
    ZoomAll
    'Hiển thị thông báo đã hoàn thành.
    MsgBox "Solid created.", , "AddRevolvedSolid Example"
  End Sub
; Kết thúc.
```

## 13.9. CÁC PHÉP BIẾN HÌNH TRONG KHÔNG GIAN 3D

### 13.9.1 Xoay các đối tượng 3D

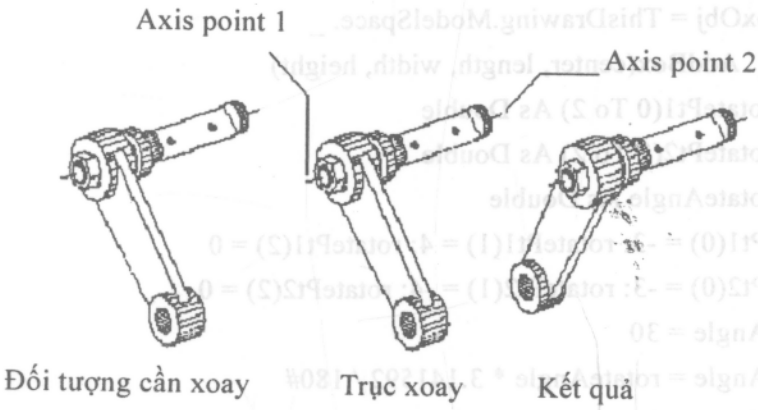
Lệnh **Rotate3D** được dùng để xoay các đối tượng 3D (hình 13.27).

**Cú pháp:**

Object.Rotate3D (*Point1, Point2, RotationAngle*)

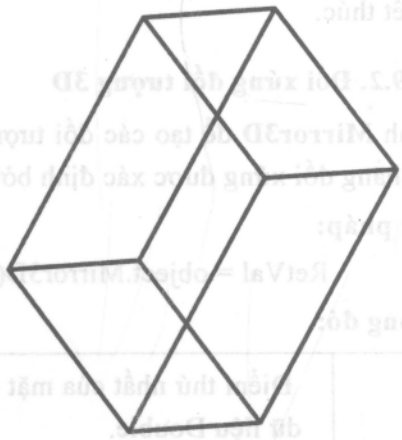
**Trong đó:**

| | |
|---|---|
| Object | Đối tượng được phương thức này áp dụng vào |
| *Point1* | Điểm thứ nhất của đường trục gồm ba phần tử (x, y, z) có kiểu dữ liệu Double. |
| *Point2* | Điểm thứ hai của đường trục gồm ba phần tử (x, y, z) có kiểu dữ liệu Double. |
| *RotationAngle* | Góc xoay quanh trục được chọn *(đơn vị radiants).* |




***Hình 13.27.*** Xoay đối tượng 3D.

Đoạn mã chương trình dưới đây tạo một khối solid chữ nhật sau đó xác định một trục xoay và xoay khối solid quanh trục đó một góc 30° như hình 13.28.



***Hình 13.28***

```
; Tên file VBA_Rotate_3Dbox.dvb.
Sub VBA_Rotate_3DBox()
  Dim boxObj As Acad3DSolid
  Dim length As Double
  Dim width As Double
  Dim height As Double
  Dim center(0 To 2) As Double
  center(0) = 5: center(1) = 5: center(2) = 0
  length = 5
  width = 7
  height = 10
```

```
    Set boxObj = ThisDrawing.ModelSpace. _
        . AddBox(center, length, width, height)
    Dim rotatePt1(0 To 2) As Double
    Dim rotatePt2(0 To 2) As Double
    Dim rotateAngle As Double
    rotatePt1(0) = -3: rotatePt1(1) = 4: rotatePt1(2) = 0
    rotatePt2(0) = -3: rotatePt2(1) = -4: rotatePt2(2) = 0
    rotateAngle = 30
    rotateAngle = rotateAngle * 3.141592 / 180#
    boxObj.Rotate3D rotatePt1, rotatePt2, rotateAngle
    ZoomAll
End Sub
; Kết thúc.
```

### 13.9.2. Đối xứng đối tượng 3D

Lệnh **Mirror3D** để tạo các đối tượng mới đối xứng với đối tượng sẵn có được chọn qua mặt phẳng đối xứng được xác định bởi 3 điểm như hình 13.29.

**Cú pháp:**

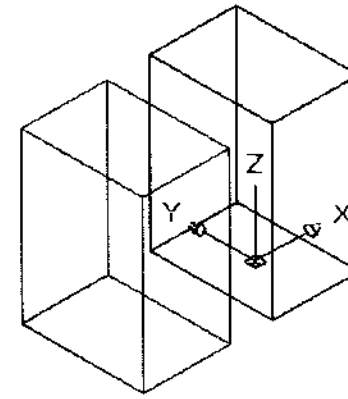
RetVal = object.Mirror3D(*Point1, Point2, Point3*)

**Trong đó:**

| | |
|---|---|
| *Point1* | Điểm thứ nhất của mặt phẳng đối xứng gồm ba phần tử (x, y, z) có kiểu dữ liệu Double. |
| *Point2* | Điểm thứ hai của mặt phẳng đối xứng gồm ba phần tử (x, y, z) có kiểu dữ liệu Double. |
| *Point3* | Điểm thứ ba của mặt phẳng đối xứng gồm ba phần tử (x, y, z) có kiểu dữ liệu Double. |


***Hình 13.29.*** Đối xứng các đối tượng 3D.

Đoạn mã chương trình dưới đây sẽ tạo một khối solid chữ nhật sau đó đối xứng khối solid qua một mặt phẳng và tô màu đỏ cho khối solid mới tạo như hình 13.30.



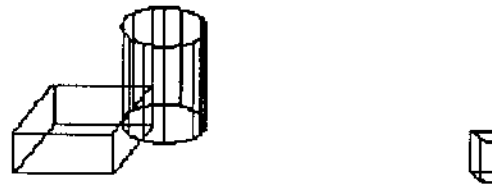

***Hình 13.30.*** Lấy đối xứng khối hộp chữ nhật.

```
; Tên file VBA_MirrorABox3D.dvb.
Sub VBA_MirrorABox3D()
   Dim boxObj As Acad3DSolid
   Dim length As Double
   Dim width As Double
   Dim height As Double
   Dim center(0 To 2) As Double
   center(0) = 5#: center(1) = 5#: center(2) = 0
   length = 5#: width = 7: height = 10#
   Set boxObj = ThisDrawing.ModelSpace. _
            AddBox(center, length, width, height)
   Dim mirrorPt1(0 To 2) As Double
   Dim mirrorPt2(0 To 2) As Double
   Dim mirrorPt3(0 To 2) As Double
   mirrorPt1(0) = 1.25: mirrorPt1(1) = 0: mirrorPt1(2) = 0
   mirrorPt2(0) = 1.25: mirrorPt2(1) = 2: mirrorPt2(2) = 0
   mirrorPt3(0) = 1.25: mirrorPt3(1) = 2: mirrorPt3(2) = 2
   Dim mirrorBoxObj As Acad3DSolid
   Set mirrorBoxObj = boxObj.Mirror3D _
            (mirrorPt1, mirrorPt2, mirrorPt3)
   mirrorBoxObj.Color = acRed
   ZoomAll
End Sub
; Kết thúc.
```

## 13.10. CHỈNH SỬA CÁC KHỐI 3D

### 13.10.1. Cộng, trừ, giao các Solid

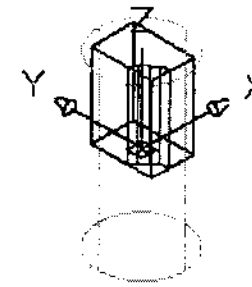
Mô hình các khối solid phức tạp có thể được tạo từ các khối solid đơn giản bằng cách sử dụng các phép đại số boole *(cộng, trừ, giao)* hình 13.31 thực hiện phép giao giữa hình hộp và khối trụ.



Khối Solid trước khi thực hiện phép giao  Kết quả khi thực hiện phép giao

***Hình 13.31.*** Giao hai khối Solid.

Đoạn mã chương trình sau đây sẽ tạo một khối solid chữ nhật và một khối solid trụ. Sau đó tìm phần giao nhau giữa hai khối solid và tạo một khối solid mới từ phần giao nhau đó. Để cho việc quan sát được thuận tiện, khối solid chữ nhật được tô mầu trắng, khối solid trụ tô mầu lục lam và phần giao nhau được tô màu đỏ như hình 13.32 minh hoạ.




***Hình 13.32.***

```
; Tên file VBA_FindInterferenceBetweenSolids.dvb.
Sub VBA_FindInterferenceBetweenSolids()
  ' Tạo khối đặc hình chữ nhật.
  Dim boxObj As Acad3DSolid
  Dim length As Double
  Dim width As Double
  Dim height As Double
  Dim center(0 To 2) As Double
  center(0) = 5: center(1) = 5: center(2) = 0
  length = 5
  width = 7
  height = 10
```

```
    Set boxObj = ThisDrawing.ModelSpace.AddBox(center, length, width, height)
    boxObj.Color = acWhite
    ' Định nghĩa khối Solid trụ.
    Dim cylinderObj As Acad3DSolid
    Dim cylinderRadius As Double
    Dim cylinderHeight As Double
    center(0) = 0: center(1) = 0: center(2) = 0
    cylinderRadius = 5
    cylinderHeight = 20
    Set cylinderObj = ThisDrawing.ModelSpace.AddCylinder _
            (center, cylinderRadius, cylinderHeight)
    cylinderObj.Color = acCyan
    ' Phần giao nhau giữu hai khối Solid.
    Dim solidObj As Acad3DSolid
    Set solidObj = boxObj.CheckInterference(cylinderObj, True)
    solidObj.Color = acRed
    ZoomAll
End Sub
; Kết thúc.
```

### 13.10.2. Cắt Solid thành hai phần

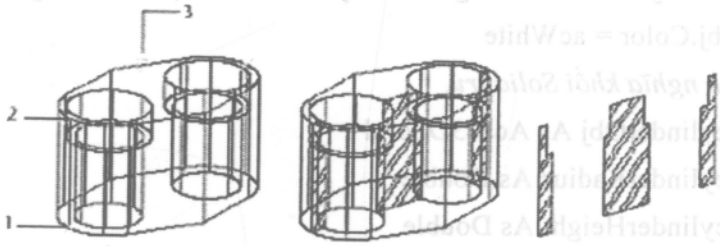
Lệnh **SliceSolid** dùng để cắt khối Solid làm hai phần.

**Cú pháp:**

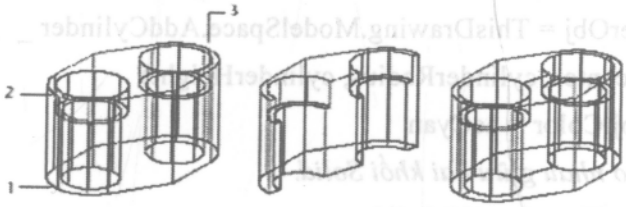
Object1 = Object2.SliceSolid(*Point1, Point2, Point3, Negative*)

**Trong đó:**

| | |
|---|---|
| Object1 | Đối tượng Solid mới được tạo ra. |
| Object2 | Đối tượng Solid gốc được chia thành hai Solid. |
| *Point1* | Điểm đầu tiên được chọn gồm ba phần tử (x, y, z) kiểu dữ liệu Double. |
| *Point2* | Điểm thứ 2 được chọn gồm ba phần tử (x, y, z) kiểu dữ liệu Double. |
| *Point3* | Điểm thứ 3 được chọn gồm ba phần tử (x, y, z) kiểu dữ liệu Double. |
| *Negative* | Có kiểu Boolean với hai giá trị như sau:<br>+ TRUE: Lấy phần Solid ở phía dưới.<br>+ FALSE: Lấy phần Solid phía trên. |



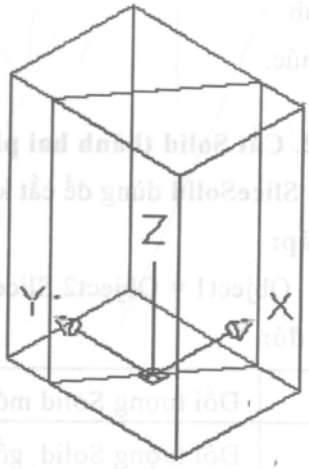

Ba điểm xác định mặt phẳng cắt  Mặt phẳng cắt  Mặt phẳng cắt ngang của khối Solid




Ba điểm xác định mặt phẳng  Một nửa mặt phẳng  Hai nửa sau khi cắt

***Hình 13.33.*** **Cắt Solid thành 2 phần.**

**Đoạn mã chương trình dưới đây tạo một solid hình chữ nhật sau đó cắt solid thành hai phần bằng một mặt phẳng xác định bởi 3 điểm cho trước như hình 13.34.**




***Hình 13.34.***

```
; Tên file VBA_SliceABox.dvb.
Sub VBA_SliceABox()
  Dim boxObj As Acad3DSolid
  Dim length As Double
```

```
    Dim width As Double
    Dim height As Double
    Dim center(0 To 2) As Double
    center(0) = 5#: center(1) = 5#: center(2) = 0
    length = 5#: width = 7: height = 10#
    Set boxObj = ThisDrawing.ModelSpace.AddBox(center, length, width, height)
    boxObj.Color = acWhite
     Dim slicePt1(0 To 2) As Double
    Dim slicePt2(0 To 2) As Double
    Dim slicePt3(0 To 2) As Double
     slicePt1(0) = 1.5: slicePt1(1) = 7.5: slicePt1(2) = 0
    slicePt2(0) = 1.5: slicePt2(1) = 7.5: slicePt2(2) = 10
    slicePt3(0) = 8.5: slicePt3(1) = 2.5: slicePt3(2) = 10
    Dim sliceObj As Acad3DSolid
    Set sliceObj = boxObj.SliceSolid(slicePt1, slicePt2, slicePt3, True)
   'Lấy mẫu đối tượng.
   sliceObj.Color = acRed
    ZoomAll
End Sub
; Kết thúc.
```

## 13.11. PHÂN TÍCH KHỐI RẮN

Mỗi một đối tượng 3DSolid có có tổng khối lượng mà ta có thể tính toán được. Ngoài ra ta có thể xác định được tâm của trọng lượng, tổng khối lượng của khối rắn, bán kính tròn xoay, quán tính và cả mô men quán tính.

Trong phần này tác giả tập trung vào phần tính khối lượng của một khối rắn.

Đoạn mã chương trình dưới đây hiển thị được khối lượng của một khối rắn được lựa chọn.

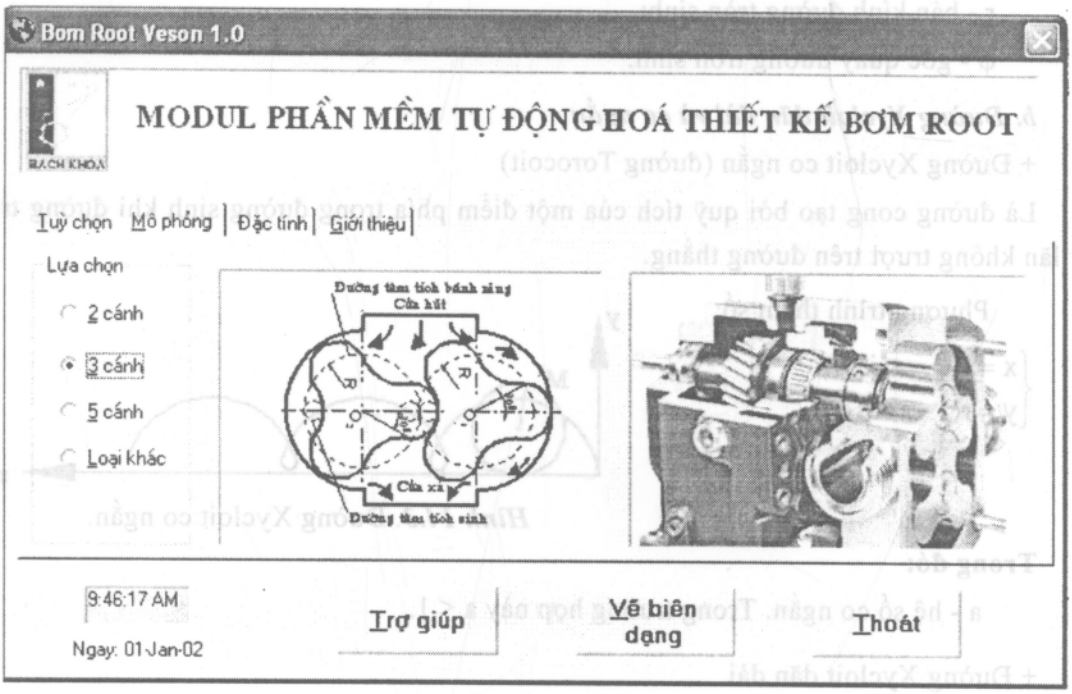
```
; Tên file mass.dvb.
Public Sub TessMassProperties
    Dim objEnt As Acad3DSolid
    Dim varPick As Variant
    Dim strMassProperties As String
    Dim varProperties As Variant
    Dim intI As Integer
```

```
On Errror Resume Next
  With ThisDrawing.Utility
   .GetEntity objEnt, varPick, vbCr & "Pick a solid:"
    if Err Then
        MsgBox "DO KHONG PHAI LA KHOI 3D"
    End Sub
End With
With objEnt
   strMassProperties  = "Volume: "
   strMassProperties  = strMassProperties & vbCr & " " & .Volume
   strMassProperties = strMassProperties & vbCr & vbCr &_ "Center of Gravity"
For Each varProperties In.Centrolid
    strMassProperties  = strMassProperties & vbCr & "          "& _ varProperties
Next
   strMassProperties = strMassProperties & vbCr & vbCr &_ "Moment of Inertia:"
For Each varProperties In.MomentOf Inertia
    strMassProperties  = strMassProperties & vbCr & "          "& _ varProperties
Next
..............
End With
; Kết thúc.
```

# Chương 14

# VẼ CÁC BIÊN DẠNG PHỨC TẠP TRONG CAD

Mặc dù ACAD đã hỗ trợ rất nhiều lệnh cho quá trình thiết kế. Tuy nhiên trong thực tế đôi khi người thiết kế gặp phải những biên dạng phức tạp đó là các đường bậc cao, ví dụ như biên dạng bơm Root là các đường (***EpiXycloit*** và ***HypoXycloit***) hay biên dạng bánh răng chốt, người thiết kế không thể thực hiện được bằng các lệnh cơ bản có trong ACAD. Do vậy người thiết kế phải lập trình để vẽ các chi tiết như vậy. Hình 14.1 dưới đây là một ví dụ minh hoạ về lập trình thiết kế biên dạng cánh bơm Root.




*Hình 14.1.* Modul phần mềm tự động hoá thiết kế bơm Root.

Chương này trình bày các ví dụ về lập các modul chương trình vẽ các đường bậc cao để độc giả tham khảo và từ đó thiết kế các ứng dụng khác cho riêng mình.

## 14.1. CƠ SỞ THIẾT KẾ CÁC ĐƯỜNG BẬC CAO

Các đường cong bậc cao vẽ trong môi trường ACAD được thực hiện bởi các đường Polyline, Spline sao cho đường này đi qua các các điểm được tính toán bởi một phương trình bậc cao cụ thể, để đường cong vẽ ra là một đối tượng và thỏa mãn là một đường cong trơn. Toạ độ các điểm mà đường cong đi qua thường là các mảng toạ độ x, y.

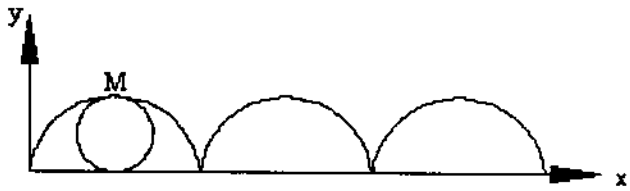
## 14.2. BIẾN DẠNG ĐƯỜNG XYCLOIT

Đường Xycloit là đường cong được tạo bởi quỹ tích của một điểm trên đường tròn sinh, khi đường tròn này lăn không trượt trên một đường thẳng được gọi là đường Xycloit.

### 14.2.1. Phương trình dạng tham số

***a. Đường Xycloit***

$$\begin{cases} x = r(\varphi - \sin\varphi) \\ y = r(1 - \cos\varphi \end{cases}$$




*Hình 14.2.* Đường Xycloit.

**Trong đó:**

r - bán kính đường tròn sinh;

φ - góc quay đường tròn sinh.

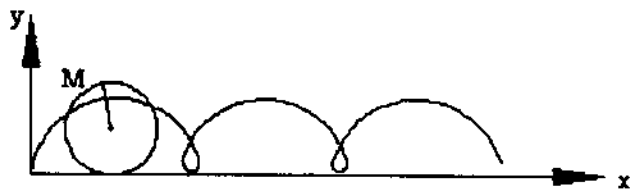
***b. Đường Xycloit dãn dài và co ngắn***

+ Đường Xycloit co ngắn (đường Tơrocoit)

Là đường cong tạo bởi quỹ tích của một điểm phía trong đường sinh khi đường tròn này lăn không trượt trên đường thẳng.

Phương trình tham số:

$$\begin{cases} x = r(\varphi - a\sin\varphi) \\ y = r(1 - a\cos\varphi) \end{cases}$$




*Hình 14.3.* Đường Xycloit co ngắn.

**Trong đó:**

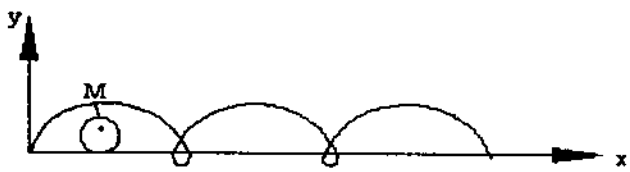
a - hệ số co ngắn. Trong trường hợp này a < 1.

+ Đường Xycloit dãn dài

Là đường cong tạo bởi quỹ tích của một điểm cố định phía ngoài đường sinh khi đường tròn này lăn không trượt trên đường thẳng.

Phương trình tham số:

$$\begin{cases} x = r(\varphi - a\sin\varphi) \\ y = r(1 - a\cos\varphi) \end{cases}$$




*Hình 14.4.* Đường Xycloit dãn dài.

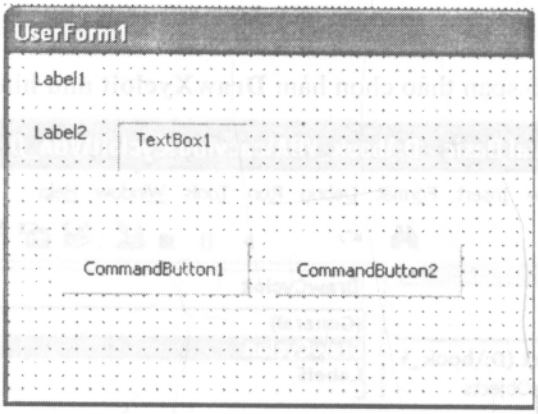
**Trong đó:**

a - hệ số co ngắn. Trong trường hợp này a > 1.

### 14.2.2. Thiết kế chương trình vẽ biên dạng đường Xycloit

+ ***B1:*** Mở trình soạn thảo Visual Basic Editor.

+ ***B2:*** Trên môi trường soạn thảo, chọn Add Form và thêm các nút điều khiển như sau:

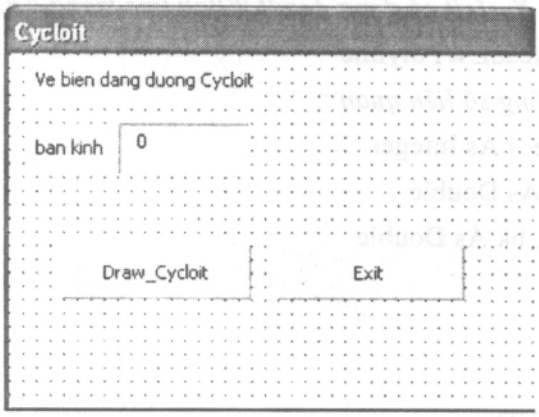



***Hình 14.5.*** Thiết kế giao diện.

+ ***B3:*** Thay đổi các thuộc tính của các điều khiển theo bảng dưới đây:

| | |
|---|---|
| UserForm1<br>Name: VBA_Xycloit<br>Caption: Xycloit | Label1<br>Name: Tieude<br>Caption: Ve bien dang duong Xycloit |
| Label 2<br>Name: vbabankinh<br>Caption: ban kinh | TexBox1<br>Name: r<br>Text: 0 |
| CommandButton1<br>Name: DrawXycloit<br>Caption: Draw_Xycloit | CommandButton2<br>Name: Thoat.<br>Caption: Exit. |

Sau khi chỉnh sửa xong giao diện có dạng như sau:

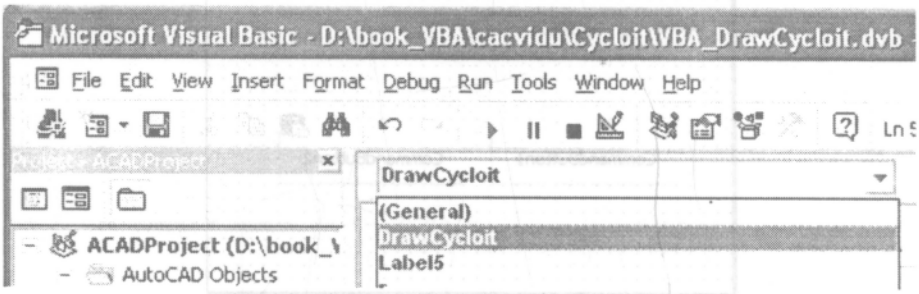



***Hình 14.6.*** Giao diện chương trình.

+ *B4:* Kích đúp chuột vào nút Exit và thêm đoạn mã lệnh như sau:

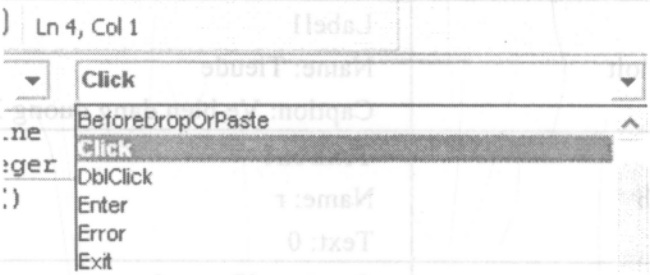
```
Private Sub Thoat_Click()
End
End Sub
```

+ *B5:* Trên cửa sổ soạn thảo chọn hàm **DrawXycloit** như hình vẽ.




*Hình 14.7.*

=>> chọn sự kiện Click.



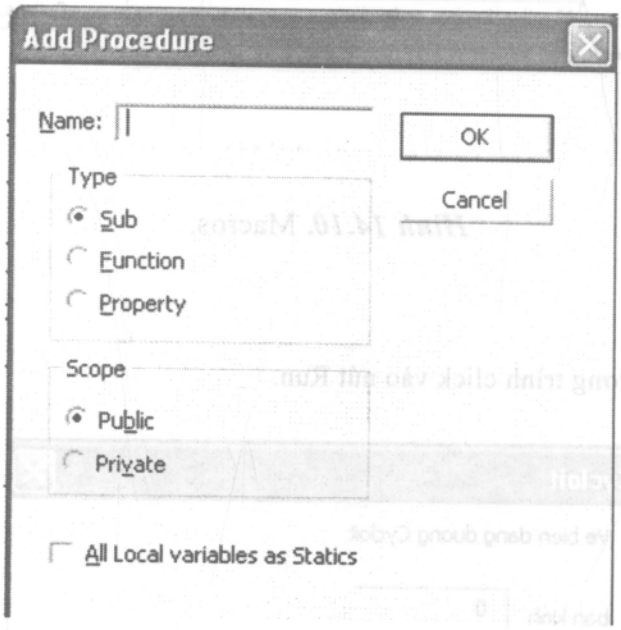

*Hình 14.8.*

Sau đó thêm đoạn mã chương trình sau:

```
Private Sub DrawXycloit_Click()
' Khai báo đường Xycloit có dạng AcadLWPolyline trong VBA.
Dim Xycloit As AcadLWPolyline
' Khai báo các thông số liên quan
Dim pi As Double, i As Integer
Dim p(0 To 719) As Double
Dim a As Double, bk As Double
bk = Val(r)
pi = 3.14
For i = 0 To 719
   If (i Mod 2) = 0 Then
      p(i) = r * (i * pi / 180 - Sin(i * pi / 180))
```

```
    Else
      p(i) = r * (1 - Cos(i * pi / 180))
    End If
  Next
  ' Tạo đường Xycloit trong không gian vẽ.
  Set Xycloit = ThisDrawing.ModelSpace.AddLightWeightPolyline(p)
  Xycloit.Color = acRed
  ' Hiển thị đối tượng trên màn hình.
  Xycloit.Update
End Sub
```

+ ***B6:*** Trong cửa sổ soạn thảo từ thanh công cụ vào Insert/ Procedure... Giao diện hộp thoại hiện ra như sau:

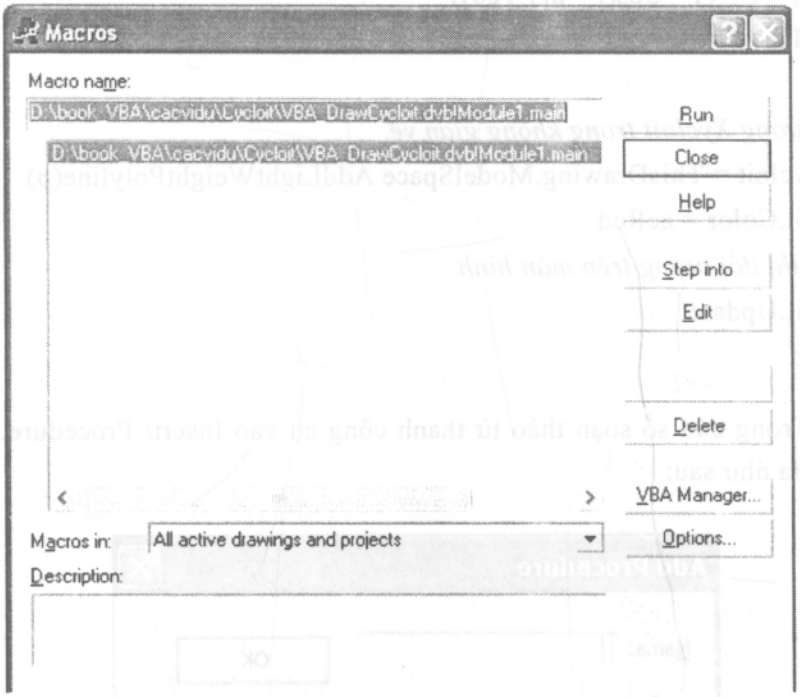



***Hình 14.9.*** Hộp thoại AddProcedure.

+ ***B7***: Trong Name TextBox nhập tên Marco Main vào với kiểu Sub và quyền truy vấn là Public để có thể gọi được ở mọi nơi ở trong chương trình. Thêm đoạn mã chương trình sau vào Module1:

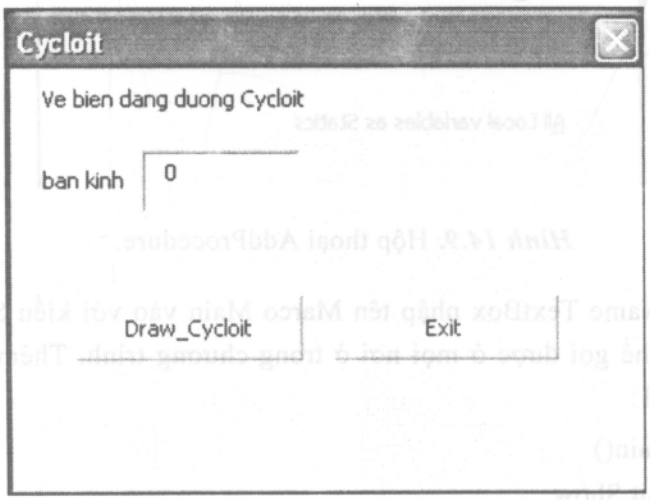
```
Public Sub main()
  VBA_Xycloit.Show
End Sub
```

+ ***B8***: Chạy Macro từ hộp thoại (hoặc dùng lệnh VBARUN từ command).




***Hình 14.10.*** Macros.

+ ***B9***: Chạy chương trình click vào nút Run.




***Hình 14.11.*** Giao diện chương trình.

+ ***B10***: Trong TextBox bán kính bạn nhập thông số bán kính vào và click vào nút công cụ Draw_Xycloit, kết quả chương trình chạy được như sau (biến chạy trong khoảng từ 0 - 2π):

***Hình 14.12.*** Kết quả chạy chương trình.

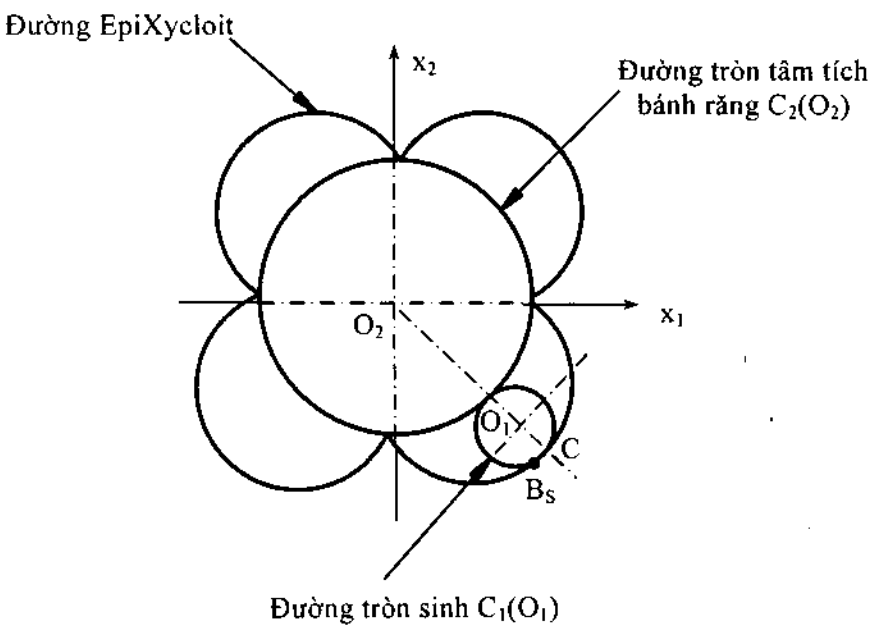
## 14.3. ĐƯỜNG EPIXYCLOIT

### 14.3.1. Cơ sở lý thuyết

Khi đường tròn sinh $C_1(O_1)$ lăn không trượt phía ngoài đường tròn tâm tích bánh răng $C_2(O_2)$ thì quỹ tích điểm $B_S$ cố định trên đường tròn sinh $C_1$ là đường EpiXycloit.

+ Phương trình dạng tham số:

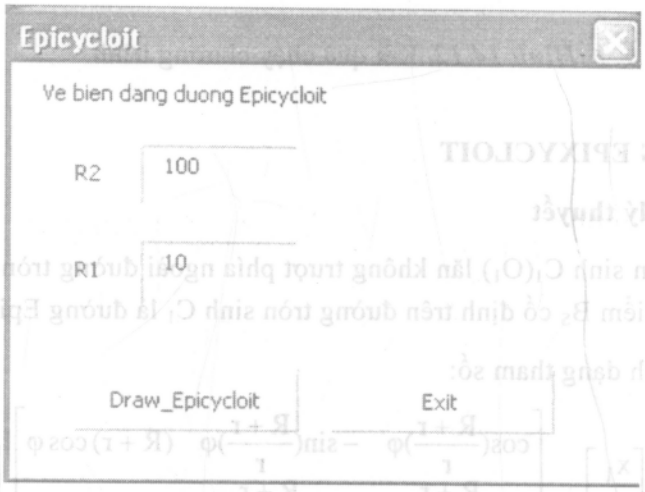
$$\begin{bmatrix} x_1 \\ x_2 \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\frac{R+r}{r})\varphi & -\sin(\frac{R+r}{r})\varphi & (R+r)\cos\varphi \\ \sin(\frac{R+r}{r})\varphi & \cos(\frac{R+r}{r})\varphi & (R+r)\sin\varphi \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -r \\ 0 \\ 1 \end{bmatrix}$$



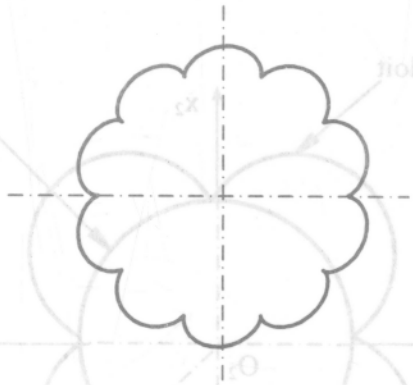

***Hình 4.13.*** Đường EpiXycloit.

### 14.3.2. Thiết kế chương trình vẽ biên dạng đường EpiXycloit

**Yêu cầu:** Thiết kế chương trình vẽ biên dạng đường EpiXycloit có giao diện chương trình như hình 14.13. Với hai thông số đầu vào là R2 (bán kính đường tròn tâm tích) và R1 (bán kính đường tròn sinh). Sau khi nhập số liệu xong chạy chương trình thu được kết quả trong MS như hình 14.14.




*Hình 14.14.* Giao diện chương trình.

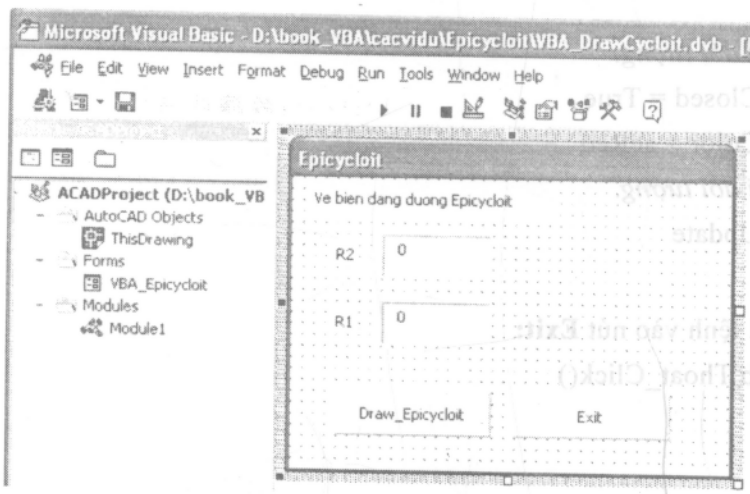


*Hình 14.15.* Kết quả chương trình.

**Thực hiện:**

Để viết chương trình trên, độc giả tham khảo vào ví dụ ở phần trên và thực hiện thêm các bước như sau:

+ ***B1:*** Trong môi trường soạn thảo VBA bạn thiết kế giao diện và thêm các điều khiển như sau:




*Hình 14.16.* Thiết kế giao diện.

+ ***B2:*** Thêm đoạn mã lệnh sau vào nút **Draw_EpiXycloit**:

```
Private Sub DrawEpiXycloit_Click()
  ' Khai báo đối tượng đường EpiXycloit.
  Dim EpiXycloit As AcadLWPolyline
  ' Khai báo các thông số.
  Dim pi As Double, i As Integer
  Dim p(0 To 359) As Double
  Dim bkR2 As Double, bkR1 As Double
  ' Nhập bán kính đường tròn tâm tích.
  bkR2 = Val(R2)
  bkR1 = Val(R1)
  pi = 3.14
For i = 0 To 359
   If (i Mod 2) = 0 Then
   p(i) = (bkR2 + bkR1) * Cos(i * pi / 180) - bkR1 * Cos(((bkR2 + bkR1) / bkR1) *
            i * pi / 180)
   Else
   p(i) = (bkR2 + bkR1) * Sin(i * pi / 180) - bkR1 * Sin(((bkR2 + bkR1) / bkR1) *
            i * pi / 180)
   End If
Next
' Tạo đường EpiXycloit trong không gian vẽ.
Set Epicloit = ThisDrawing.ModelSpace.AddLightWeightPolyline(p)
```

```
  ' Làm kín đối tượng.
  Epicloit.Closed = True
  Epicloit.Color = acRed
  ' Hiển thị đối tượng.
  Epicloit.Update
End Sub
```

+ Thêm mã lệnh vào nút **Exit:**

```
Private Sub Thoat_Click()
    End
End Sub
```

+ ***B3:*** Lưu và chạy chương trình như các bước ở ví dụ trên.

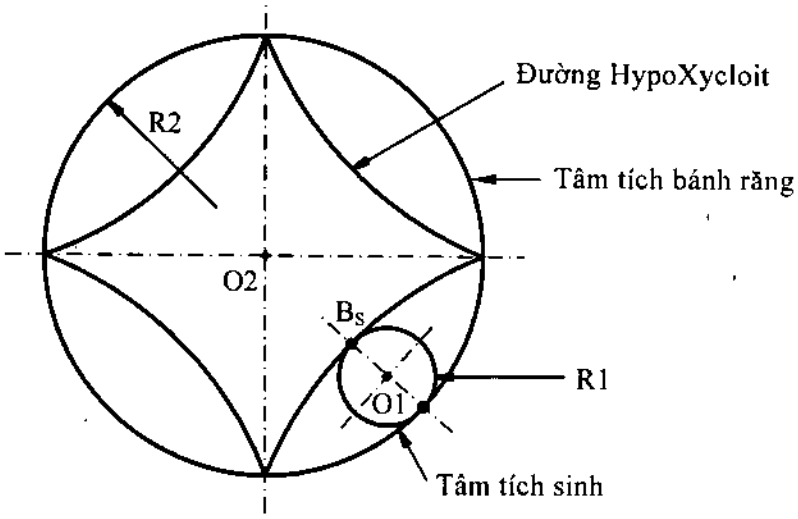
## 14.4. ĐƯỜNG HYPOXYCLOIT

### 14.4.1. Cơ sở lý thuyết

Là đường cong được tạo bởi quỹ tích một điểm cố định trên đường tâm tích sinh $C(O_1R_1)$ tròn sinh khi đường này lăn không trượt phía trong đường tâm tích bánh răn $C(O_2R_2)$ tròn lăn.

+ Phương trình dạng tham số:

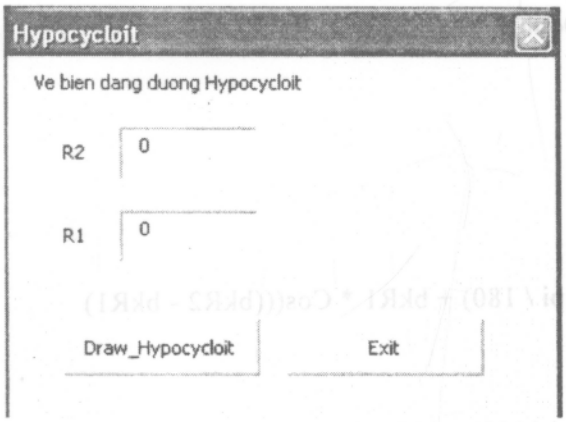
$$\begin{bmatrix} x_1 \\ x_2 \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(-\frac{R+r}{r})\varphi & -\sin(-\frac{R+r}{r})\varphi & (R+r)\cos\varphi \\ \sin(-\frac{R+r}{r})\varphi & \cos(-\frac{R+r}{r})\varphi & (R+r)\sin\varphi \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r \\ 0 \\ 1 \end{bmatrix}$$



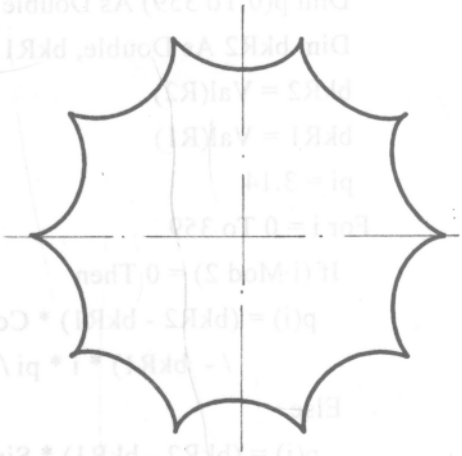

***Hình 4.17.*** Đường HypoXycloit.

### 14.4.2. Thiết kế chương trình vẽ biến dạng đường HypoXycloit

**Yêu cầu:** Thiết kế chương trình vẽ đường HypoXycloit có giao diện chương trình như hình 14.17. Với hai thông số đầu vào là R2 (*bán kính đường tròn tâm tích*) và R1 (*bán kính đường tròn sinh*). Sau khi nhập số liệu xong chạy chương trình thu được kết quả trong môi trường không gian vẽ như hình 14.18.




*Hình 14.18.* Giao diện chương trình.

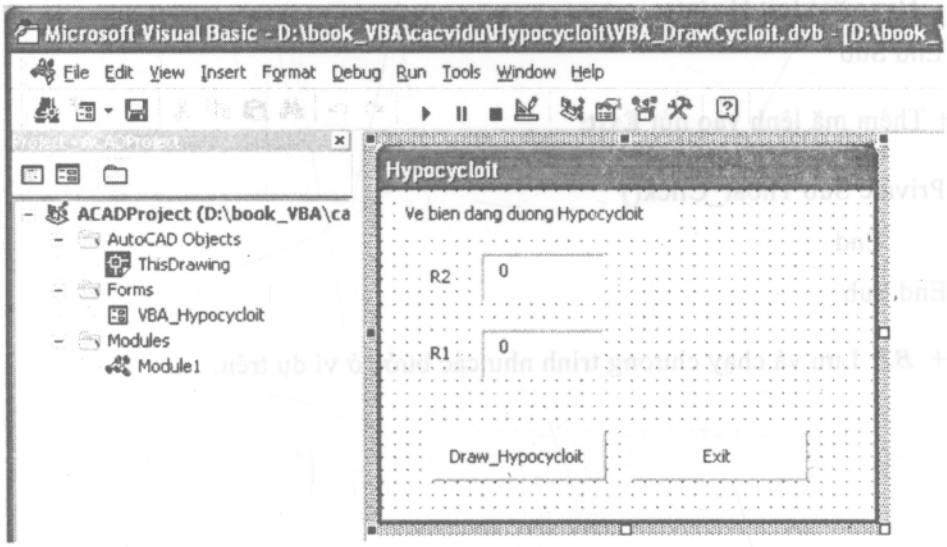


*Hình 14.19.* Kết quả chương trình.

**Thực hiện:**

Để viết chương trình trên, các bạn tham khảo vào ví dụ ở phần trên và thực hiện như sau:

+ ***B1:*** Trong môi trường soạn thảo VBA bạn thiết kế giao diện và thêm các điều khiển như sau:




*Hình 14.20.* Thiết kế giao diện.

+ ***B2:***

- Thêm đoạn mã lệnh sau vào nút **Draw_HypoXycloit.**

```
Private Sub DrawHypoXycloit_Click()
   Dim HypoXycloit As AcadLWPolyline
   Dim pi As Double, i As Integer
   Dim p(0 To 359) As Double
   Dim bkR2 As Double, bkR1 As Double
   bkR2 = Val(R2)
   bkR1 = Val(R1)
   pi = 3.14
  For i = 0 To 359
    If (i Mod 2) = 0 Then
      p(i) = (bkR2 - bkR1) * Cos(i * pi / 180) + bkR1 * Cos(((bkR2 - bkR1)
             / -  bkR1) * i * pi / 180)
    Else
      p(i) = (bkR2 - bkR1) * Sin(i * pi / 180) + bkR1 * Sin(((bkR2 - bkR1)
             /-bkR1) * i * pi / 180)
    End If
  Next
  Set HypoXycloit = ThisDrawing.ModelSpace.AddLightWeightPolyline(p)
  HypoXycloit.Closed = True
  HypoXycloit.Color = acRed
  HypoXycloit.Update
End Sub
```

+ Thêm mã lệnh vào nút **Exit:**

```
Private Sub Thoat_Click()
    End
End Sub
```

+ ***B3:*** Lưu và chạy chương trình như các bước ở ví dụ trên.

## Phụ lục

# BẢNG TRA CỨU

| Tên | Ý nghĩa | Trang |
|---|---|---|
| 3 | | |
| 3DFace | Mặt phẳng có ba cạnh hoặc bốn cạnh | 217 |
| 3DPolyline | Mặt phẳng 3D được tạo từ các đường thẳng gấp khúc | 208 |
| 3DSolid | Khối rắn với các mặt phẳng tự do | 220 |
| A | | |
| AcCmColor | Kiểu màu trong ACAD | 132 |
| Arc | Đối tượng cung tròn | 85 |
| Application | Một trường hợp ứng dụng trong ACAD | 19 |
| Attribute | Dùng để thể hiện các thuộc tính của các đối tượng trong CAD | 185 |
| ActiveUCS | Đặt hệ toạ độ trong bản vẽ | 207 |
| ActiveViewport | Đặt chế độ khung nhìn trong bản vẽ | 71 |
| AltTolerancePrecision | Độ chính xác của giá trị ghi kích thước | 163 |
| Annotation | Giá trị dung sai | 171 |
| AltUnitsFormat | Định dạng đơn vị | 162 |
| ArrowheadType | Các kiểu mũi tên khi ghi kích thước | 148 |
| AngleFormat | Định dạng đơn vị đo góc (*độ, radians*) | 154 |
| Add3Dface | Tạo mặt phẳng 3D từ bốn đỉnh | 217 |
| Add3Dmesh | Tạo mặt lưới đa giác | 210 |
| Add3Dpoly | Tạo các đa tuyến trong 3D | 208 |
| AddArc | Tạo đối tượng cung tròn (*tâm, bán kính, điểm đầu, điểm cuối*) | 85 |
| AddBox | Tạo khối hình lập phương, chữ nhật | 219 |
| AddCircle | Tạo đối tượng đường tròn (*tâm, bán kính*) | 84 |
| AddLine | Vẽ đoạn thẳng đi qua hai điểm (*điểm đầu,điểm cuối* ) | 74 |
| AddLightweightPolyline | Tạo đối tượng gồm nhiều đoạn thẳng liên tiếp đi qua các điểm | 76 |

**Bảng tra cứu** (tiếp theo)

| Tên | Ý nghĩa | Trang |
|---|---|---|
| AddMLine | Tạo đối tượng gồm nhiều đoạn thẳng liên tiếp song song và cách đều một khoảng xác định | 78 |
| AddXLine | Tạo đối tượng là đường thẳng vô tận đi qua hai điểm | 79 |
| AddRay | Tạo đối tượng đường thẳng bị hạn chế một đầu, còn đầu kia là vô tận | 80 |
| AddSpline | Tạo một đường cong tự do | 81 |
| AddEllip | Tạo đường Ellipse trong bản vẽ | 87 |
| AddPoint | Tạo các điểm trong bản vẽ | 90 |
| AddSolid | Vẽ đối tượng là đa giác được tô 2D | 91 |
| AddText | Tạo một dòng chữ trong bản vẽ | 92 |
| AddMText | Tạo nhiều dòng chữ trong bản vẽ | 93 |
| AddCone | Tạo các khối nón | 220 |
| AddWedge | Tạo khối hình nêm | 222 |
| AddCylinder | Tạo khối hình trụ | 223 |
| AddSphere | Tạo khối cầu | 224 |
| AddTorus | Tạo khối hình xuyến | 225 |
| AddExtrudedSolid | Tạo khối rắn theo phương pháp đùn | 227 |
| AddExtrudedSolidAlongPath | Tạo khối rắn đùn theo đường dẫn | 228 |
| AddRevolvedSolid | Tạo khối rắn bằng phương pháp quay một biên dạng quay quanh một trục | 230 |
| AddDimAligned | Ghi kích thước thẳng | 149 |
| AddDimRotated | Ghi kích thước có đường kính nghiêng với đường chuẩn một góc xác định bất kỳ | 151 |
| AddDim3PointAngular | Ghi kích thước góc | 153 |
| AddDimDiametric | Ghi kích thước đường kính | 155 |
| AddDimRadial | Ghi kích thước bán kính | 159 |
| AddDimOrdinate | Ghi toạ độ điểm | 160 |
| AddTolerance | Ghi dung sai kích thước | 164 |
| AddLeader | Đánh số chi tiết và ghi chi dẫn | 171 |
| AddHatch | Khởi tạo mặt cắt | 115 |
| ArrayPolar | Sao chép các đối tượng theo mảng tròn | 100 |

**Bảng tra cứu** (tiếp theo)

| Tên | Ý nghĩa | Trang |
|---|---|---|
| ArrayRectangular | Sao chép các đối tượng theo mảng chữ nhật | 102 |
| AttachExternalReference | Tham khảo ngoài | 198 |
| Add | Khởi tạo một bản vẽ | 61 |
| ActiveLayer | Tạo một lớp mới | 126 |
| **B** | | |
| Block | Định nghĩa, khởi tạo và sử dụng Block | 173 |
| Blocks collection | Liên kết các khối Blocks trong bản vẽ | 173 |
| Blockef | Nhận các hằng số thuộc tính | 193 |
| Blue | Màu | 131 |
| Blind | Chuyển bản vẽ tham khảo ngoài thành Block của bản vẽ hiện hành | 201 |
| **C** | | |
| Circle | Đường tròn (*tâm, bán kính*) | 84 |
| Color | Làm việc với kiểu màu trong ACAD | 131 |
| ColorName | Tên kiểu màu | 131 |
| Copy | Sao chép các đối tượng | 95 |
| **D** | | |
| Delete | Xóa các đối tượng | 107 |
| Detach | Khi không tham khảo ngoài | 199 |
| DimAligned | Đường kích thước thẳng | 149 |
| DimRotated | Đường kích thước nghiêng với đường chuẩn một góc xác định | 151 |
| Dim3PointAngular | Kích thước góc qua ba điểm | 153 |
| DimDiametric | Kích thước đường kính | 155 |
| DimRadial | Kích thước bán kính (*cung tròn, góc lượn, đường tròn*). | 159 |
| DimOrdinate | Toạ độ điểm trong không gian vẽ | 160 |
| **E** | | |
| Explode | Phá vỡ Block | 184 |
| Ellipse | Đường Ellipse | 87 |
| EndAngle | Góc kết thúc của cung tròn | 85 |
| ExtensionLineColor | Đặt màu cho đường gióng kích thước | 162 |

**Bảng tra cứu** (tiếp theo)

| Tên | Ý nghĩa | |
|---|---|---|
| **F** | | |
| Freeze | Đóng và làm tan băng lớp trên khung nhìn | 130 |
| **G** | | |
| GetAttributes | Chèn các thuộc tính Block | 188 |
| GetConstantAttributes | Nhận các hằng số thuộc tính | 193 |
| Green | Màu xanh da trời | 131 |
| **H** | | |
| HasAttributes | Kiểm tra các thuộc tính của Block | 192 |
| Hatch | Mặt cắt | 115 |
| HatchStyle | Kiểu mặt cắt | 123 |
| **I** | | |
| InsertBlock | Chèn Block | 180 |
| IsoPenWidth | Chiều rộng bút vẽ | 120 |
| **J** | | |
| **K** | | |
| **L** | | |
| LayerOn | Ẩn hiện lớp | 129 |
| Layer | Làm việc với lớp trong ACAD | 129 |
| Lock | Khoá, mở khoá cho lớp | 130 |
| Linetype | Kiểu đường trong acad.lin | 135 |
| LineWeight | Độ rộng của đường | 137 |
| Line | Đường thẳng đi qua hai điểm | 74 |
| Leader | Đường chỉ dẫn | 171 |
| LightweightPolyline | Đường gồm nhiều đoạn thẳng liên tiếp nối với nhau | 76 |
| **M** | | |
| Mirror | Đối xứng các đối tượng qua một trục đối xứng cho trước | 99 |
| Move | Di chuyển các đối tượng | 104 |
| Mclose | Đóng, mở mặt theo hướng M | 212 |
| ModelSpace | Không gian vẽ | 74 |
| Mline | Các đường thẳng song song với nhau | 78 |

**Bảng tra cứu** (tiếp theo)

| Tên | Ý nghĩa | |
|---|---|---|
| **N** | | |
| Nclose | Đóng, mở mặt theo hướng N | 214 |
| Name | Tên các đối tượng trong bản vẽ | 95 |
| **O** | | |
| Open | Mở một bản vẽ | 61 |
| Offset | Sao chép các đối tượng theo khoảng cách xác định cho trước | 97 |
| **P** | | |
| PatternAngle | Tạo độ nghiêng của các đường cắt so với mẫu chọn | 123 |
| PatternSpace | Đặt khoảng cách giữa các đường gạch mặt cắt | 125 |
| PatternType | Các kiểu mặt cắt | 116 |
| PatternName | Tên của mặt cắt | 116 |
| Point | Đối tượng điểm | 90 |
| PaperSpace | Không gian giấy | 74 |
| PolyLine | Đường gồm nhiều đoạn thẳng nối với nhau | 76 |
| PrimaryUnitsPrecision | Độ chính xác của kích thước | 163 |
| **Q** | | |
| **R** | | |
| Rotate | Quay các đối tượng | 105 |
| Reload | Cập nhật sự thay đổi của bản vẽ tham khảo ngoài và bản vẽ hiện hành | 200 |
| Red | Màu đỏ | 131 |
| Ray | Đường thẳng bị chặn một đầu | 81 |
| **S** | | |
| Save | Ghi một bản vẽ | 61 |
| SaveAs | Ghi một bản vẽ | 61 |
| ScaleEntity | Phóng to, thu nhỏ các đối tượng theo 1 tỷ lệ | 107 |
| SliceSolid | Cắt khối rắn thành hai phần | 237 |
| Spline | Đường cong tự do | 81 |
| StartAngle | Góc ban đầu của cung tròn | 86 |

**Bảng tra cứu** (tiếp theo)

| Tên | Ý nghĩa | Trang |
|---|---|---|
| **T** | | |
| TransformBy | Biến đổi các đối tượng | 109 |
| TextHeight | Chiều cao chữ, số trong ACAD | 169 |
| ToleranceDisplay | Ghi dung sai kích thước | 170 |
| ToleranceJustification | Vị trí giá trị dung sai của kích thước | 170 |
| ToleranceLowerLimit | Sai lệch dưới dung sai kích thước | 171 |
| ToleranceUpperLimit | Sai lệch trên dung sai kích thước | 171 |
| TextColor | Màu các kiểu chữ trong ACAD | 167 |
| TextGrap | Khoảng cách giữa giá trị kích thước và đường kích thước | 168 |
| TextInside | Giá trị kích thước ở bên trong đường gióng | 169 |
| **U** | | |
| Unsaved | Thực hiện việc không ghi bản vẽ | 61 |
| Unload | Loại bỏ bản vẽ tham khảo ngoài từ bản vẽ hiện hành. | 200 |
| UCS | Hệ trục toạ độ được xác định bởi người sử dụng | 206 |
| Update | Khôi phục lại các chi tiết trong bản vẽ *(object.Update)* | 76 |
| **V** | | |
| VBAIDE | Mở trình soạn thảo VBA | 32 |
| VBALOAD | Nạp file vào trong môi trường ACAD | 28 |
| VBARUN | Chạy Macro | 30 |
| VBAMAN | Hiển thị hộp thoại VBA Manager | 32 |
| Viewport | Đối tượng khung nhìn (*khung nhìn tĩnh, khung. nhìn động*) | 69 |
| **W** | | |
| WindowState | Trạng thái cửa sổ | 62 |
| WBlock | Ghi Block lên đĩa | 182 |
| WCS | Hệ toạ độ gốc | 203 |

**Bảng tra cứu** (tiếp theo)

| Tên | Ý nghĩa | Trang |
|---|---|---|
| **X** | | |
| XLine | Đường thẳng vô tận đi qua hai điểm | 79 |
| **Y** | | |
| **Z** | | |
| Zoom | Quan sát bản vẽ | 65 |
| ZoomAll | Quan sát toàn bộ bản vẽ | 68 |
| ZoomExtent | Phóng to, thu nhỏ bản vẽ khi quan sát | 68 |
| ZoomCenter | Zoom về tâm màn hình đồ hoạ | 67 |
| ZoomScaled | Xác định tỷ lệ để quan sát bản vẽ | 66 |
| ZoomWindow | Phóng to, thu nhỏ phần lựa chọn trên màn hình đồ hoạ | 65 |

# TÀI LIỆU THAM KHẢO


[1] **Nguyễn Hồng Thái;** Bài giảng môn học kỹ thuật lập trình AutoLisp và VBA; Đại học Bách khoa Hà Nội.

[2] **Nguyễn Hồng Thái, Nguyễn Tiến Dũng, Vương Văn Thanh;** Phần mềm mô phỏng hình động học bơm Root; Tạp chí Khoa học và Công nghệ, trang 53 ÷ 56; số 55-2006.

[3] **Nguyễn Hồng Thái, Nguyễn Quang Huy;** "Lập trình tự động hoá thiết kế cơ khí trong Solidworks"; trang 215 ÷ 219; tập 2, Tuyển tập Hội nghị Khoa học toàn quốc về Cơ kỹ thuật và Tự động hoá, Hà Nội 2006.

[4] **Trịnh Đồng Tính, Nguyễn Hồng Thái;** "Báo cáo Đề tài cấp bộ B2006-01-19"; Hà Nội, 9-2007.

[5] **Trần Hữu Quế;** Vẽ kỹ thuật cơ khí tập 1; Nhà xuất bản Đại học và Giáo dục chuyên nghiệp, Hà Nội 1992.

[6] **JoeSutphin;** Auto Card 2006 VBA Aprogrammer's Reference; Apress; Hardcover, 2nd edition; Published September 2005; 776 pages;

[7] **Paul J. Drake;** Jr. Dimenfiening and Toleranciny Handbook; McGraw-Hill NewYork San Francisco Washington.

2 0 7 3 1 0


**Giá: 58.000$^{đ}$**