

# CHƯƠNG 1: GIỚI THIỆU CHUNG

## I. Ngôn ngữ lập trình trong CAD

Autocad cho ta các cách để có thể tự động quá trình vẽ như sau:

Sử dụng các ngôn ngữ lập trình, sau đó kết xuất ra bản vẽ dưới dạng các file văn bản DXF.

Tự động hóa bằng các file Script.

Lập trình trong môi trường CAD như Lisp, Object ARX, VBA.

AutoDesk cung cấp cho chúng ta một bộ các phần mở rộng để kiểm soát AutoCad từ ngôn ngữ. Những phần mở rộng này được gọi là Object ARX. Đây là một phương pháp tiếp cận với CAD một cách chuyên nghiệp nhất tuy nhiên nó lại quá phức tạp.

AutoLisp là một ngôn ngữ lập trình thông dịch, Nó là một phiên bản mới nhất về ngôn ngữ lập trình nhân tạo cũ nhất mà ngày nay vẫn còn được sử dụng. Autolisp nằm trong bộ Common LISP. LISP viết tắt của LIST Processor !. Nói chung Lisp dễ học bởi cú pháp của nó đơn giản nhưng nó không tương tác được với các cơ sở dữ liệu như Excel, access. Nên việc sử dụng nó tạo ra các ứng dụng phức tạp là rất khó. Tuy nhiên mức độ phức tạp cũng đã được giảm bớt đi rất nhiều trong Visual LISP!

VBA viết tắt của Visual Basic Application. Cũng như Object Arx thì VBA cũng là một môi trường lập trình hướng đối tượng sử dụng ngôn ngữ VB. Ưu điểm của VBA là

Sử dụng VB, một ngôn ngữ lập trình tương đối thông dụng và dễ học.

VBA nằm trong CAD nên tốc độ chạy cũng tương đối nhanh.

Dễ dàng trong việc tạo ra các giao diện (hộp thoại, menu).

Tương tác với các ứng dụng khác và các cơ sở dữ liệu khác.

Cho phép ta ghi Project ra file riêng hoặc tích hợp luôn vào bản vẽ. Tạo điều kiện để phát triển ứng dụng một cách mềm dẻo trong việc chia sẻ dữ liệu giữa các ứng dụng khác trong môi trường Window.

## II. Tổng quan về Activex Automation

### 2 ưu điểm của AutoCAD ActiveX

Activex automation là chuẩn mực được tạo ra bởi hãng Microsoft, trước đây được gọi là OLE activex, cho phép một ứng dụng Windows này kiểm soát một ứng dụng Windows khác qua mô hình các đối tượng rõ ràng.

AutoCAD ActiveX là giao diện cho phép người lập trình làm việc với các đối tượng của AutoCAD. AutoCAD Activex cho phép bạn sử dụng một cách tự động không chỉ trong phạm vi AutoCAD mà ngoài cả AutoCAD. Các đối tượng của AutoCAD có thể được truy nhập đến bởi nhiều ngôn ngữ lập trình khác nhau như Ms word VBA, Ms Excel VBA,...

## 2 ưu điểm của AutoCAD ActiveX

Tất cả các ngôn ngữ lập trình đều có thể làm việc với các đối tượng trong AutoCAD (không giới hạn với C++ và AutoLISP như trước)

Chia sẻ dữ liệu với các ứng dụng khác trên môi trường Windows (Excel, Word...)

### Mô hình client server.

Mặc dù Activex luôn luôn bao gồm một cuộc hội thoại giữa hai ứng dụng, nó không phải là cuộc hội thoại hai chiều giữa các thành phần tương đương. Mỗi thành phần chương trình Activex Automation bao gồm hai chương trình với các vai trò khác nhau. Client là ứng dụng khởi tạo cuộc hội thoại. Server là ứng dụng hồi đáp client. Mã Activex Automation chạy trong client, trong khi các hành động mã này được kiểm soát thực hiện trên server. Hình dưới đây trình bày mối quan hệ giữa Client và Server trong một cuộc trao đổi Activex Automation đặc thù.

ClientClientStarts conversationRequest ObjectSupplies ObjectInvokes MethodClose conversationServerPrinciple Activex Automation Supplies Object

Dưới đây là một số ứng dụng bạn có thể dùng để kiểm soát các server activex, kể cả AutoCAD.

Visual Basic

Excel VBA

Word VBA

## 8. Mô hình đối tượng của Automation.

Một Server Activex Automation (chẳng hạn như AutoCAD) thực hiện các chức năng qua các đối tượng. Một đối tượng là một đại diện của thành ứng dụng. Một đối tượng được phân biệt với các đối tượng khác bởi ba đặc tính:

Phân loại của đối tượng

Các đặc tính của đối tượng

Phương thức của đối tượng

Ví dụ đối tượng Line, đặc tính của đối tượng cho phép bạn xác định:

- Màu sắc
- Lớp
- Điểm khởi đầu
- Điểm kết thúc
- Độ dài

Phương thức đối tượng là

- Sao chép

- xóa
- đối xứng qua gương
- di chuyển
- quay

Chú ý, các Autocad activex sẽ không hoạt động nếu cad đang thực hiện lệnh.

### **III. Ngôn ngữ lập trình AutoLisp**

#### **1. Giới thiệu chung**

Ưu điểm: Tốc độ chạy nhanh

Nhược điểm:

Là ngôn ngữ lập trình thông dịch, ko cấu trúc

Không kết nối đc với các cơ sở dữ liệu như Access, Excel...

#### **2. Căn bản về AutoLisp**

##### **2.1. Xây dựng biểu thức AutoLisp**

Cấu trúc dữ liệu cơ bản của Lisp là danh sách (List)

Danh sách là tập hợp các phần tử chứa trong các dấu ngoặc đơn, các phần tử đc cách nhau bởi một hoặc nhiều dấu cách

Danh sách có 2 loại: Biểu thức toán học (expression) và danh sách dữ liệu (data list).

Phần tử đầu tiên của của một biểu thức luôn luôn là một hàm

Một biểu thức bao gồm tên hàm và các tham số chứa trong các dấu ngoặc đơn. Khác với biểu thức toán học, các tham số trong biểu thức Lisp là các tham số có thứ tự.

Tham số là các giá trị cung cấp cho hàm để tính toán

AutoLisp trả về kết quả tính toán từ biểu thức.

Ví dụ: (+ 30 20 50) Trả về kết quả: 100

##### **2.2. Cách nhập biểu thức AutoLisp**

Biểu thức Lisp có thể đc nhập như các dòng lệnh của AutoCAD, chú ý rằng biểu thức của Lisp luôn luôn nằm trong dấu ngoặc đơn ( ... ).

Khi nhập trực tiếp bằng dòng lệnh, bạn nên chú ý kéo dài cửa sổ AutoCAD Text Window ra để có thể theo dõi đc kết quả cũng như lỗi của nó.

Command: (( + 140 10) 30)↵

##### **2.3. Các hàm số học**

**Hàm cộng (+):** (+ [number 1] [number 2] [number 3]....)

Danh sách bắt đầu bằng dấu + báo cho lisp đó là hàm, các phần tử đứng

sau nó sẽ là tham số.

Dữ liệu số đc chia làm 2 loại:

Số nguyên (ko có dấu chấm)

Số thập phân (có dấu chấm)

Kết quả trả về sẽ có kiểu là kiểu rộng nhất trong các kiểu của tham số.

Ví dụ: Command: (+ 140 10 30)↵

180

Command: (+ 140 10 30.0)↵

180.0

Command: (+ 140 10.10 30)↵

180.1

Command: (+ 10 a)↵

; error: bad argument type: numberp: nil

**Hàm trừ (/)**

**Hàm nhân (\*)**

**Hàm chia (/)**

Command: (/ 120 10 3)↵

4

Command: (\* 3.75 3.775)↵

14.0625

Lisp lưu trữ tới 14 số thập phân nhưng kết quả trả về trên màn hình sẽ chỉ có 6 chữ số có nghĩa tính từ trái sang phải.

Ví dụ: 18[(3+6+9):(96)12]

Command: (((18 (/ (+ 3 6 9) (9 6))) 12)

0

### **3. Biến trong Lisp**

**Ký hiệu**

Tên biến cũng như tên hàm ko phân biệt chữ hoa, chữ thường.

Biến không nhất thiết cần khai báo

**Gán giá trị cho biến**

Sử dụng hàm Setq để gán giá trị cho một biến. Cú pháp như sau:

(SetQ Symbol1 value1 [symbol2 value2] ...)

Giống như các hàm khác, hàm SetQ trả về một giá trị. Giá trị này có thể là “nil” (rỗng), “T” (True) hoặc các số, chuỗi, danh sách.

Ví dụ:

Command (Setq x 3)↵

3

Command (Setq x 3 y 4)↵

4

Command (Setq z (+ x y))↵

7

Command (Setq A “Xyabg”)↵

Xyabg

### ***Phạm vi biến***

Biến chỉ có tác dụng trong phạm vi bản vẽ

### ***Sử dụng biến trong dòng lệnh***

Command: (setq x 10)↵

10

Command: (!X)

10

Command: circle↵

3P/..<Center point>: Nhập tọa độ tâm

Diameter <Radius>: !x

## **4. File chương trình Lisp**

### **4.1. File lisp**

#### ***Quy định chung:***

File Lisp có phần mở rộng là \*.lsp

Một biểu thức có thể viết trên nhiều dòng

Các biểu thức không phân biệt chữ hoa, chữ thường

Chuỗi chú thích bắt đầu bằng dấu chấm phẩy

#### ***Các tải file lisp vào trong AutoCAD***

Sử dụng hàm Load

Command: (Load E:/autoLisp/CHT.lsp)

Hoặc

Command: (Load E:\\toLisp\\T.lsp)

Sử dụng hàm Appload (Application Load)

## 4.2. Hàm tự tạo

Ngoài các hàm AutoCAD cung cấp, ta còn có thể tạo ra các hàm tự tạo.

### ***Định nghĩa hàm tự tạo***

Bằng cách sử dụng hàm Defun (define function). Cú pháp như sau:

(Defun Function\_Name Argument\_List expresstion...)

Function\_Name: tên hàm tự tạo. Tuân theo quy tắc đặt tên biến

Argument\_List: gồm hai phần ngăn cách nhau bởi dấu /, Phần thứ nhất chứa các tham số cần thiết khi gọi là hàm. Phần thứ hai chứa các biến cục bộ của hàm.

Expression: các biểu thức tính toán của hàm. Các biểu thức này sẽ lần lượt đc tính toán theo thứ tự từ trên xuống dưới.

### ***Ví dụ:***

(Defun ZA()

(command "Zoom" "all")

)

### ***Biến toàn cục và biến cục bộ***

Biến toàn cục là các biến hoạt động trong phạm vi bản vẽ

Biến cục bộ là biến đc định nghĩa trong phạm vi hàm và giá trị của nó sẽ mất đi khi hàm kết thúc.

## 4.3. Tạo lệnh AutoCAD mới.

### ***Tham số C:***

Để có thể sử dụng hàm tự tạo như là một lệnh trong AutoCAD, ta đặt ký hiệu C: vào trước tên hàm trong phần định nghĩa hàm tự tạo.

Ví dụ:

(Defun C: (/PT1 PT2)

(Setq PT1 (getpoint "\n nhập diem thu nhât"))

(Setq PT2 (getpoint "\n nhập diem thu hai"))

(command "Line" PT1 PT2 " ");

(Princ)

) ;ket thuc

### ***Tham số S::Startup***

Khi khởi động AutoCAD, hàm S::startup định nghĩa trong file ACADRx.lsp sẽ được tự động gọi và thi hành. Đây là hàm duy nhất có tính chất này.

## 5. Nhập dữ liệu

### 5.1. Nhập dữ liệu người dùng

#### ***Nhập tọa độ một điểm: Hàm getpoint***

(GetPoint [pt] [prompt])

Hàm này sẽ trả về một danh sách. Danh sách này thuộc dạng danh sách lưu trữ dữ liệu (Data Storage list). Loại danh sách này khác với biểu thức ở chỗ, phần tử đầu tiên của danh sách không phải là một hàm. Khi ta nhập dữ liệu vào, để AutoCAD nhận biết được kiểu dữ liệu danh sách ta dùng hàm Quote (hoặc dấu ‘).

Ví dụ ta vẽ đường thẳng đi qua một điểm có tọa độ (2,2,0) và một điểm nhận được từ người dùng.

```
(Defun C: (/PT1)
```

```
(setq PT1 (getpoint “\n Nhập diem thu nhap”))
```

```
(Command “Line” PT1 Quote(2 2 0))
```

```
)
```

Hoặc:

```
(Defun C: (/PT1)
```

```
(setq PT1 (getpoint “\n Nhập diem thu nhap”))
```

```
(Command “Line” PT1 “2,2,0”)
```

```
12
```

```
)
```

Trình tự các tham số hàm Command tương ứng với trình tự nhập lệnh tại dòng nhắc. Cú pháp đầy đủ: (Command [argument]...)

#### ***Nhập số nguyên (integer): Hàm getint***

Cú pháp: (Getint [prompt])

Ví dụ:

```
Command: (Getint “\n Enter an integer”)
```

```
12.0
```

```
Requires an integer value
```

```
Try again: 23
```

```
23
```

#### ***Nhập dữ liệu số thực (real): Hàm Getreal***

Cú pháp như sau: (Getreal [Prompt])

#### ***Nhập dữ liệu kiểu chuỗi (string)***

Cú pháp như sau: (Getstring [Prompt])

## 5.2. Kiểm soát dữ liệu nhập vào

Hàm getint cung cấp danh sách các giá trị nhập vào hợp lệ bằng cách gán các bit kiểm tra (bit code) và danh sách các từ khóa. Các loại hàm nhập dữ liệu như Getpoint, getcorner, getint, Getreal,... (ngoại trừ hàm GetstringO đều bị kiểm soát bởi hàm initget. Hàm có tác dụng đối với hàm nhập dữ liệu tiếp theo sau nó. Cú pháp của hàm như sau:

(Initget [bits] [string])

Bits là một số nguyên. Giá trị tham số này bằng tổng các bit code tương ứng với các chế độ kiểm soát mà t among muốn

Tham số String chứa danh sách các từ khóa.

### Bit code

#### Chế độ kiểm soát

1 Giá trị phải đc nhập vào; không chấp nhậ giá trị null

2 Giá trị nhập vào phải khác không

4 Giá trị nhập vào không đc là số âm

128 Cho phép nhập chuỗi ký tự không có trong danh sách các từ khóa. Các bit code khác sẽ đc ưu tiên trước.

### Hàm GetKword

Hàm này yêu cầu nhập dữ liệu ở dạng từ khóa. Cú pháp như sau:

(GetKWord [Prompt])

Hàm getkword chỉ chấp nhận 2 bit code trong hàm initget là 1 và 128.

Ví dụ:

Command: (initget 1 “Y N”)←

Nil

Command: (setq abc (getkword “\n ban co ghi lai khong ? <Y/N>”))←

## 5.3. Biến hệ thống

### Lấy giá trị biến hệ thống

(Getvar Varname)

### Gán giá trị biến hệ thống

(Setvar varname value)

### Biến hệ thống quan trọng

CmdEcho

Value =1: Kết quả tính toán trung gian sẽ đc hiện lên trên màn hình.



Value =0: Kết quả tính toán trung gian sẽ ko đc hiện lên trên màn hình.

13

## **6. Một số hàm cơ bản**

### **6.1. Hàm chuyển kiểu dữ liệu**

***Chuyển đổi một số thành số thực: Hàm (Atof String)***

Command: (Atof “15.4a”)

15.4

***Chuyển đổi một chuỗi thành một số nguyên: hàm (Atoi String)***

Command: (Atof “15.4a”)

15

***Chuyển đổi một số thành một chuỗi (real to string):Hàm (Rtos Number [mode [precision]])***

Precision: số chữ số thập phân

Mode là kiểu số (scientific, decimal,...)

Mode

Format

1 Scientific

2 Decimal

3 Engineering

4 Architectural

Number: số sẽ đc chuyển qua kiểu (mode) với số chữ số thập phân đc quy định trong (precision) sau đó đc chuyển thành chuỗi tương ứng. Nếu không có 2 tham số này thì AutoCAD sẽ lấy biến hệ thống trong Units để thực hiện phép toán.

Ví dụ

(Rtof 215 2) trả về: “2.15E+2”

***Chuyển đổi một số nguyên thành chuỗi: hàm (Itoa integer)***

Ví dụ:

(Itoa 21) trả về “21”

(Itoa 30.2) trả về lỗi

***Hàm ASCII***

Cú pháp: (Ascii String): chuyển đổi ký tự đầu tiên của chuỗi thành mã ký tự ascii tương ứng.

Ví dụ:

(Ascii "Abc") trả về 65.

(Ascii "9Ac") trả về 57.

### ***Hàm CHR***

Cú pháp: (Chr Integer): chuyển đổi mã ascii thành ký tự tương ứng trong bản mã ASCII. Các mã ascii chuẩn có giá trị từ 32 đến 126.

## **6.2.**

### **Hàm toán học**

#### ***Hàm kiểm soát dạng số***

(Fix Number): trả về phần nguyên của một số thực.

(Float Number): Chuyển số Number thành kiểu số thực.

(Abs Number): trả về trị tuyệt đối của một số.

#### ***Hàm max,min***

(Max Number1 Number 2 ...) Chú ý không chấp nhận chuỗi.

(Min Number1 Number 2 ...)

#### ***Hàm lượng giác***

(Sin Angle) Trả về giá trị của một góc, đơn vị Angle là radians.

(Cos Angle)

(Atan Angle). Giá trị trả về từ  $\pi/2$  đến  $\pi/2$

#### ***Hàm lũy thừa, khai căn, logarit***

14

(Expt Base Power)

(Expt 4.0 4) trả về giá trị 64.0

(SQRT Number)

(Log Number) trả về logarit của một số.

(Exp Number) Trả về e mũ n.

### 6.3.

#### **Các hàm về khoảng cách và góc đo**

##### ***Hàm Cvunit (convert units)***

Cú pháp: (cvunit Value From To)

Value: số nguyên, số thực, hoặc tọa độ điểm 2D, 3D

From Đơn vị đo hiện tại (kiểu chuỗi)

To Đơn vị đo sẽ chuyển sang (kiểu chuỗi)

Ví dụ:

(Cvunit Pi "RADIANS" "DEGREE") trả về 3.14159

(Cvunit '(1 3) "FT" "IN" trả về (12.0 36.0)

##### ***Hàm Angle***

Cú pháp (Angle PT1 PT2): Trả về góc (Radians) giữa đường thẳng đi qua 2 điểm với trục X trong mặt phẳng XY. Nếu 2 điểm này không nằm trên mặt phẳng XY, nó sẽ đc chiếu lên mặt phẳng XY và tính góc.

Ví dụ

Command: (Angle '(5 6.10) (quote (10 5)))↵

6.06664

### 6.4.

#### **Các hàm về chuỗi**

##### ***Hàm hiển thị thông tin kiểu chuỗi***

Cú pháp: (Princ [Expr [file]]) Hàm này in ra màn hình hoặc in ra file

Cú pháp: (Print [Expr [file]]) Hàm này in ra màn hình hoặc in ra file trên một dòng mới.

Cú pháp: (Prin1 [Expr [file]]) Hàm này in ra màn hình hoặc in ra file trên một dòng mới.

Ví dụ:

**Command:** (princ "\nabc \nabc\n")

abc

abc

"\nabc \nabc\n"

**Command:** (prin1 "\nabc \nabc\n")

"\nabc \nabc\n""\nabc \nabc\n"

**Command:** (print "\nabc \nabc\n")

"\nabc \nabc\n" "\nabc \nabc\n"

Các ký tự đặc biệt cho hàm Princ

\n: xuống dòng

\t: để cách ra như một khoảng Tab

### ***Hàm StrCase***

Cú pháp: (StrCase String [switch])

Nếu switch <> nil hàm sẽ trả về chuỗi String trong đó các ký tự hoa đc chuyển thành chữ thường

Nếu Switch ko có hoặc bằng nil thì hàm sẽ trả về chuỗi String trong đó các ký thường sẽ đc chuyển thành ký tự hoa.

### ***Hàm StrCat***

Cú pháp: (StrCat String [string2]...): kết nối các chuỗi tham số.

### ***Hàm StrLen***

Cú pháp: (StrLen String [string]): trả về chiều dài của một xâu. Nếu có nhiều xâu hàm sẽ trả về tổng chiều dài của các xâu tham số.

### **Hàm SubStr**

Cú pháp: (SubStr String Start [length]): Trả về một xâu con của xâu String bắt đầu từ vị trí Start và dài length ký tự. Nếu không có length, nó sẽ lấy đến tận cuối xâu.

7.

### **Xử lý danh sách**

7.1.

#### **Tổng quan**

Danh sách (List) được phân làm 3 loại chính

Biểu thức (Expression list): chứa tên hàm và các tham số của hàm. Biểu thức trả về giá trị

Tọa độ điểm (Point Coordinate list): có hàm quote hoặc dấu ‘ ở đằng trước. Chứa tọa độ X, Y, Z của một điểm.

Kho dữ liệu (Data storage List): Cũng như Point coordinate list. Nhưng nó chứa dữ liệu bất kỳ.

7.2.

#### **Tạo danh sách**

Tất các dữ liệu của AutoCAD (Autocad database) đều đc lưu dưới dạng danh sách và đc đánh số thứ tự theo mã (DXF Code). Khi viết chương trình, để quản lý dữ liệu, thông thường ta lưu vào các biến. Nhưng khi số lượng lưu trữ dữ liệu tăng lên. Lisp không cung cấp kiểu dữ liệu động và mảng động. Để giải quyết vấn đề này, ta sử dụng kiểu dữ liệu List.

Một trong nhiều phương pháp tạo ra danh sách là sử dụng hàm List

Cú pháp: (List expression)

Ví dụ

Command: (set q L1 (list “abc” 10 30.0 “hoang”)

(“abc” 10 30.0 “hoang”)

Trong ví dụ trên L1 có 4 phần tử.

Ngoài ra ta có thể sử dụng hàm Quote hoặc dấu ‘ ở đằng trước. Ví dụ:

Command: (setq a1 (quote("abc" 1 30 40)))←

```
("abc" 1 30 40)
```

Hoặc

```
Command: (setq a1 ("abc" 1 30 40))↵
```

```
("abc" 1 30 40)
```

Sự khác nhau cơ bản giữa List và quote (hoặc ‘). List tạo ra danh sách định giá trị, khi tạo ra danh sách, lisp sẽ định giá trị và kiểu dữ liệu tương ứng cho các phần tử trong danh sách. Còn quote tạo ra danh sách chưa định giá trị. Chúng ta xem xét ví dụ sau:

```
Command: (setq a2 '( a b c))↵
```

```
(A B C)
```

```
Command: (setq a2 (list a b c))↵
```

```
(nil nil nil)
```

(Vì a b c là 3 biến chưa có giá trị)

### 7.3.

#### **Hàm xử lý danh sách cơ bản**

##### ***Hàm Car***

Cú pháp: (Car List): Dùng để lấy giá trị đầu tiên của danh sách.

Ví dụ:

```
Command: (Car a1)↵
```

```
“abc”
```

##### ***Hàm CDR***

Cú pháp: (CDR list): Tạo ra một danh bằng cách loại bỏ phần tử đầu tiên của danh sách gốc.

Ví dụ:

16

```
Command: (setq a3 (CDR a1))↵
```

```
(1 30 40)
```

##### ***Hàm CADR, CADDR***

(CADR list) trả về phần tử thứ 2 của danh sách.

(CADDR list) trả về phần tử thứ 3 của danh sách.

### ***Hàm Last***

Cú pháp: (Last list): Trả về phần tử cuối cùng của danh sách.

### ***Hàm Length***

Cú pháp: (Length List): trả về số lượng phần tử có trong danh sách.

## **7.4.**

### **Hàm xử lý danh sách nâng cao**

#### ***Hàm Assoc (association)***

Cú pháp: (Assoc Item AList): Danh sách Alist phải chứa phần tử Item, và là danh sách phức hợp. Hàm sẽ trả về một danh sách con của Alist mà phần tử đầu tiên là Item. Nếu không tìm thấy phần tử Item trong Alist thì hàm sẽ trả về giá trị nil.

Ví dụ:

**Command:** (Setq Alist '((1 "Name" "NGuyen hoang anh") (2 "Toan" 10) (3 "Ly" 6)))

((1 "Name" "NGuyen hoang anh") (2 "Toan" 10) (3 "Ly" 6))

**Command:** (setq Toan (assoc 2 Alist))

(2 "Toan" 10)

**Command:** !toan

(2 "Toan" 10)

**Command:** (last (assoc 2 Alist))

10

#### ***Hàm Cons (construct)***

Cú pháp: (Cons Item List) bổ xung phần tử Item vào vị trí đầu tiên của danh sách.

Ví dụ:

**Command:** (Setq AL (list (Cons 'COLOR 4) (Cons 'LAYER 0)))

#### ***Hàm Member***

Cú pháp: (Member Item list): trả về một danh sách bắt đầu bằng phần tử Item.

Ví dụ:

**Command:** (Setq L '(1 2 3 4 5)←

( 1 2 3 4 5)

Command: (member 3 L)←

(3 4 5)

### ***Hàm Append***

Cú pháp: (Append list1 list2 ... ): Gộp nhiều danh sách thành một danh sách.

Command: (setq L2 (Append (member 3 L) (list 6)))

(3 4 5 6)

## **8.**

### **Biểu thức điều kiện**

#### **8.1.**

### **Biểu thức điều kiện**

#### ***Các hàm so sánh***

#### ***Hàm***

#### ***Cú pháp***

#### ***Giải thích***

=

(= Atom Atom ..)

***Trả về giá trị T nếu tất cả các phần tử bằng nhau. Tham số kiểu số hoặc kiểu chuỗi.***

/=

(/= Atom Atom ..)

***Trả về T nếu các phần tử đôi một khác nhau***

17

<

(< Atom Atom ..)

***Trả về T nếu mỗi phần tử nhỏ hơn phần tử đứng bên phải nó***

<=; >=; >

***Tương tự***

***Tương tự***



***EQ***

***(EQ Expr1 Expr2)***

***So sánh sự trùng nhau của 2 danh sách.***

***Equal***

***(Equal Expr1 Expr2 [fuzz])***

***Hàm định giá trị các biểu thức và kiểm tra các giá trị này có bằng nhau hay không. Đối với dữ liệu kiểu số Fuzz quy định sai số trong phép so sánh***

***Các hàm kiểm tra dữ liệu***

***Hàm***

***Cú pháp***

***Giải thích***

***Atom***

***(Atom Item.)***

***Trả về giá trị Nil nếu Item là list***

***Listp***

***(Listp Item)***

***Trả về T nếu Item là danh sách hoặc giá trị Nil***

***Numberp***

***(numberp Item)***

***Trả về T nếu giá trị là số nguyên***

***Minusp***

***(minusp Item)***

***Trả về T nếu giá trị là số âm***

***Zerop***

***(Zerop Item)***

***Trả về T nếu dữ liệu =0 or 0.0***

***Null***

***(Null Item)***

***Kiểm tra một biến có rỗng hay ko***

***Type***

***(Type Item)***

***Trả về kiểu dữ liệu của biến***

**8.2.**

## **Hàm If và Progn**

### ***Cấu trúc rẽ nhánh với If***

Cú pháp:

(If LogicExpr ThenExpr)

(If LogicExpr ThenExpr ElseExpr)

Chú ý:

Hàm If chỉ chấp nhận một biểu thức ThenExpr và ElseExpr. Nếu sử dụng nhiều hơn một biểu thức thì phải sử dụng cấu trúc Progn

(Progn Expression ....)

Ví dụ:

(DEFUN C:11 ()

(if (= (getvar "pickstyle") 0) (setvar "pickstyle" 1) (setvar "pickstyle" 0))

(PRINC)

)

Bài tập

Giải phương trình bậc nhất

Nhập vào 2 điểm, kiểm tra xem chúng có nằm trong giới hạn Limmax và Limmin hay không. Nếu có hãy đưa ra khoảng cách giữa 2 điểm đó. Sử dụng hàm (Distance PT1 PT2).

### **8.3.**

#### **Hàm logic**

(Setq A 10 b 20 c 30)

#### ***Hàm***

#### ***Cú pháp***

#### ***Ví dụ***

#### ***And***

***(And Expression .. )***

*(And (> 10 0) (< 10 20)) trả về giá trị T*

*Or*

*(or Expression .. )*

*(Or (= A 10) (< b 0)) trả về giá trị T*

*Not*

*(Not Item)*

*(Not nil) = T. (Not T) = nil*

18

## **8.4.**

### **Hàm Cond (Condition)**

Hàm if cho phép ta rẽ tối đa 2 nhánh. Hàm Cond giúp ta rẽ nhiều hơn 2 nhánh. Cú pháp hàm như sau:

*(Cond*

*(Test1 result1 ... )*

*(Test2 result2 ... )*

*.....*

*(Testn resultn ... )*

*)*

Bài tập

Giải phương trình bậc 2.

## **9.**

### **Vòng lặp**

#### **9.1.**

#### **Vòng lặp cơ bản**

#### ***Vòng lặp Repeat.***

Hàm Repeat tạo ra vòng lặp với số lần nhất định

Cú pháp: *(Repeat Number Expr .. )*

Ví dụ:

```
(Setq I 10)
(setq j 1)
(Repeat 10
  (setq i (+ 2 i))
  (setq j (* 2 j))
) ; kết thúc hàm repeat
Lặp 10 lần khi đó  $I = 10 + 2 * 10 = 30$ ;  $j = 2^{10}$ .
```

### ***Vòng lặp While.***

Hàm while tạo ra vòng lặp có điều kiện. Vòng lặp này sẽ kết thúc khi điều kiện Testexpr không thỏa mãn.

```
Cú pháp: (While testexpr Expr ..)
Ví dụ: Vẽ đường thẳng đi qua 2 điểm
(defun c:L2P (/ ch PT1 pt2)
  (setq ch "Y")
  (while (or (= ch "Y") (= ch "y"))
    (setq PT1 (getpoint "\nnh?p vào ?i?m th? nh?t: "))
    (setq PT2 (getpoint "\nnh?p vào ?i?m th? hai: "))
    (command "Line" PT1 PT2 ""))
  (initget 1 "Y N y n")
  (Setq ch (getkeyword "\n b?n có v? n?a không (Y/N): "))
) ; ket thuc while
) ; ket thuc defun
```

## **9.2.**

### **Hàm foreach**

Cú pháp: (Foreach Name List Expr ...)

Hàm Foreach duyệt từng phần tử trong danh sách LIST. Tại mỗi thời điểm, giá trị của từng phần tử trong danh sách sẽ đc gán cho biến Name. Sau đó các biểu thức Expr sẽ đc định giá trị.

Ví dụ:

```
(setq I 0)
(foreach So (list 1 2 3 4 5 6 7 8 9 10)
  (Setq I (1+ i))
  (Princ (Strcat "\nCác giá trị thứ “ (itoa i) “trong danh sách là” (itoa so)))
)
```

19

**10.**

**Tập hợp các đối tượng được chọn**

**10.1.**

**Hàm Ssget**

***Hàm SSget***

Cú pháp: (Ssget [Mode][PT1][PT2][PTList][FilterList])

***Mode***

***Phương pháp chọn***

***Cú pháp***

*None*

*Sử dụng mọi phương pháp chọn (hay dùng)*

*(Ssget)*

*<Point>*

*Chọn đối tượng đi qua điểm Point*

*(Ssget point)*

*“L”*

*Last: chọn đối tượng đc tạo ra cuối cùng*

*(Ssget “l”)*

*“P”*

*Previous*

*(ssget “p”)*

*“W”*

*Window*

*(Ssget “W” PT1 PT2)*

*“C”*

*Crossing window*

*(Ssget “C” PT1 PT2)*

*“F”*

*Fence*

(Ssget "F" PTlist  
"WP")

*Window polygon*

(ssget "wp" PTlist  
"CP")

*Crossing polygon*

(ssget "cp" PTlist  
"X")

*All*

(ssget "x")

Ví dụ:

Setq PT1 '(0 0 0) PT2 '(4 0 0) PT3 '(4 4 0) PT4 '(0 4 0)  
(setq SS (ssget "WP" (list pt1 pt2 pt3 pt4)))

***Bảng Group code.***

***Sử dụng filter***

Ví dụ:

(ssget '( (0 . "text") (40 . 2.5)))

(ssget '( (0 . "line") (62 . 4)))

Sử dụng các phép so sánh. Chú ý các phép so sánh chỉ áp dụng đối với các group code có kiểu số (nguyên hoặc thực). Các phép so sánh bao gồm "\*" "=" "/=" "!=" ">" "<" ">=" "<=" "<>"

Ví dụ:

(Ssget '((0 . "Circle") (4 . "<>") (40 . 50) (4 . ">, <,\*") (10 0.0 0.0 0.0)))

Đường tròn bán kính group code 40, tọa độ tâm group code 10

Sử dụng các phép toán logic. Các phép toán logic And Or Not

Ví dụ:

(Ssget "X" '((4 . "<OR") (0 . "Text") (0 . "Line") (0 . "Circle") (4 . "Or>"))))

Một vài group code: tọa độ điểm đầu, cuối của Line: 10,11. Giá trị Text: 12.

**10.2.**

**Hàm SSLength**

Cú pháp: (SSLength ss)

### 10.3.

#### **SSName**

Cú pháp: (Ssname ss index): trả về tên của đối tượng thứ index trong tập hợp chọn SS. (lưu ý: số đầu tiên của tập hợp chọn index=0).

Ví dụ xóa phần tử thứ nhất của danh sách chọn

(command “erase” (Ssname SS 0) “”)

Hàm EntGet (entity get)

Cú pháp: (ENTGET entname)

(SETQ DS (ENTGET (SSNAME SS DEM)))

(SETQ KDL (CDR (ASSOC 0 DS)))

20

### 11.

#### **Lập trình với cơ sở dữ liệu của AutoCAD.**

##### 11.1.

#### **Lấy Record dữ liệu đối tượng**

Hàm EntGet (entity get)

Cú pháp: (ENTGET entname): trả về danh sách biểu diễn Record dữ liệu của đối tượng có mã là entname.

##### 11.2.

#### **Hiệu chỉnh record đối tượng**

Các bước hiệu chỉnh

Tạo ra record mới chứa các field đã thay đổi bằng hàm Subst.

Thay thế Record cũ của đối tượng bằng record mới bằng hàm Entmod.

Cập nhật sự thay đổi của đối tượng lên màn hình (thay cho lệnh regen).  
Sử dụng hàm EntUpd

#### **Hàm Subst**

Để tạo ra một record mới bằng cách sử đổi một số phần tử của record cũ.

Cú pháp: (Subst new\_Item Old\_Item list)

Nếu không tìm thấy old\_Item, hàm sẽ trả về danh sách giống danh sách ban đầu.

Ví dụ:

(Setq al '(A 22 34 "yes" B 22))

(Subst 22 11 al) trả về A 11 34 "yes" B 22

(setq El '((1.<entity name 2d3314>) (0."Line") (5. "20") (100. "ACDB entity") (67 . 0) (8 . "0") (100 . "acdbline") (10 0 0 0) (11 5.0 5.0 0))

(setq el (subst '(8 . "Layer Dim") '(8 . 0) EL))

### ***Hàm Entmode***

Hàm entmode thay thế record cũ trong cơ sở dữ liệu bằng một record mới.

Cú pháp: (Entmode Elist)

Elist là một danh sách có dạng một record đối tượng trong đó mã đối tượng nằm ở group code 1. Hàm này sẽ tìm trong cơ sở dữ liệu đối tượng có mã như trong code 1 và thực hiện việc thay thế.

Các trường hợp không thể thay thế đc hàm sẽ trả về giá trị nil:

Không tìm thấy đối tượng

Thay đổi mã đối tượng

Thay đổi mã handle

Hiệu chỉnh đối tượng viewport

Thay đổi kiểu đối tượng,,,

### ***Hàm Entity update***

Cú pháp: (EntUpd Ename): Dùng để cập nhật sự thay đổi record của đối tượng có tên là Ename lên màn hình đồ họa.

### **11.3.**

#### **Tạo đối tượng mới.**

Hàm (entmake Elist) tạo ra đối tượng mới

Các quy định cho elist



Tham số elist không nhất thiết phải có đủ các thông số

Field thứ nhất bắt buộc phải là group code 0 chứa kiểu đối tượng

Mã đối tượng AutoCAD sẽ tự đặt khi đối tượng đc tạo ra

**12.**

**Phân tích ví dụ:**

```
;  
*****  
*****  
;* TRAN ANH BINH CDCHUCE *  
;  
*****  
*****
```

(defun myerror (s) ; If an error (such as CTRLC) occurs  
; while this command is active...

21

(cond

((= s "quit / exit abort") (princ))

((/= s "Function cancelled") (princ (strcat "\nError: " s)))

)

(setvar "cmdecho" CMD) ; Restore saved modes

(setvar "osmode" OSM)

(setq \*error\* OLDERR) ; Restore old \*error\* handler

(princ)

)

;

```
*****  
*****
```

(DEFUN C:CD (/ CMD SS LTH DEM PT DS KDL N70 GOCX GOCY  
PT13 PT14 PTI PT13I PT14I

```

PT13N PT14N O13 O14 N13 N14 OSM OLDERR PT10 PT11)
(SETQ CMD (GETVAR "CMDECHO"))
(SETQ OSM (GETVAR "OSMODE"))
(SETQ OLDERR *error*
*error* myerror)
(PRINC "Please select dimension object!")
(SETQ SS (SSGET))
(SETVAR "CMDECHO" 0)
(SETQ PT (GETPOINT "Point to trim or extend:"))
(SETQ PT (TRANS PT 1 0)) ; chuyen tu current UCS sang WCS (world)
(COMMAND "UCS" "W")
(SETQ LTH (SSLENGTH SS))
(SETQ DEM 0)
(WHILE (< DEM LTH)
(PROGN
(SETQ DS (ENTGET (SSNAME SS DEM)))
(SETQ KDL (CDR (ASSOC 0 DS)))
(IF (= "DIMENSION" KDL)
(PROGN
(SETQ PT10 (CDR (ASSOC 10 DS)))
(SETQ PT11 (CDR (ASSOC 11 DS)))
(SETQ PT13 (CDR (ASSOC 13 DS)))
(SETQ PT14 (CDR (ASSOC 14 DS)))
(SETQ N70 (CDR (ASSOC 70 DS)))
(IF (OR (= N70 32) (= N70 33) (= N70 160) (= N70 161))
(PROGN
(SETQ GOCY (ANGLE PT10 PT14))
(SETQ GOCX (+ GOCY (/ PI 2)))
)
)
(SETVAR "OSMODE" 0)
(SETQ PTI (POLAR PT GOCX 2))
(SETQ PT13I (POLAR PT13 GOCY 2)) ; tao ra mot diem moi

```

```
(SETQ PT14I (POLAR PT14 GOCY 2)) ; (polar PT angle distance)
(SETQ PT13N (INTERS PT PTI PT13 PT13I NIL)) ; nill thi cac duong
thang se dc keo dai,
(SETQ PT14N (INTERS PT PTI PT14 PT14I NIL)); kha nill thi se khong
dc keo dai
(SETQ O13 (ASSOC 13 DS))
(SETQ O14 (ASSOC 14 DS))
(SETQ N13 (CONS 13 PT13N))
(SETQ N14 (CONS 14 PT14N))
```

22

```
(SETQ DS (SUBST N13 O13 DS))
(SETQ DS (SUBST N14 O14 DS))
(ENTMOD DS)
)
)
(SETQ DEM (+ DEM 1))
)
)
(COMMAND "UCS" "P")
(SETVAR "CMDECHO" CMD)
(SETVAR "OSMODE" OSM)
(setq *error* OLDERR) ; Restore old *error* handler
(PRINC)
)
;
*****
*****
```

23

#### **IV.**

#### **Ngôn ngữ lập trình Visual Lisp**

#### **V.**

#### **Cơ bản về ngôn ngữ lập trình Visual Basic**

#### **VI.**

#### **Làm quen với VBA.**

##### **1.**

#### **VBA Projects**

Chương trình VBA được tổ chức trong các project, 1 Project là tập hợp các module: mã, lớp, form

VBA Project có thể được lưu trong file bản vẽ DWG (chế độ nhúng) hoặc được lưu trong 1 file riêng (\*.DVB)

##### **2.**

#### **Tạo mới project**

#### **Mở VBA Manager,**

Chọn menu Tools/Macro/VBA Manager. Hoặc trên command line, gõ lệnh VBAMAN.

#### **Tạo mới VBA project**

Mở VBA Manager, chọn New, project mới sẽ có tên là ACADProject.

#### **Đổi tên project.**

Bạn phải sử dụng VBA IDE, nhấn Alt+F11, click phải chuột vào tên project, nhấn Rename

#### **Lưu cất (save) project**

Project nhúng được lưu khi lưu bản vẽ

Project ko nhúng phải lưu qua VBA IDE (chức năng File/Save) hoặc VBA Manager

#### **Tải (load) 1 project đã có**

Project nhúng được load ngay khi bạn mở bản vẽ chứa nó

Project không nhúng được lưu lại dưới dạng file \*.DVB, để load project này: trên cửa sổ VBA Manager, nhấn nút Load.

Trên hộp thoại OpenFileDialog, chọn file DVB cần mở

Để xem nội dung của Project, nhấn Alt+F11 hoặc dùng lệnh VBAIDE trên dòng commandline

### 3.

#### **Tổ chức các project với VBA Manager**

Giới thiệu chức năng các nút trong hộp thoại VBA Manager

### 4.

#### **Soạn thảo project với VBA IDE**

24

Giới thiệu các cửa sổ của VBA IDE

Cửa sổ Project Manager

Objects (đối tượng)

Forms

Gồm các hộp thoại do người lập trình tạo ra, sử dụng trong project

Modules

Các hàm dùng chung trong toàn bộ Project được tổ chức trong các module riêng (phụ thuộc vào chức năng của chúng)

Class Modules

Định nghĩa các lớp đối tượng của người dùng

Để thêm 1 component (form, module, class module) vào project

Chọn project cần thêm thành phần component

Trên menu [Insert], chọn [UserForm], [Module], [Class Module] để thêm các thành phần này vào Project

Module, Class module được soạn thảo trên cửa sổ Code

UserForm được soạn thảo trên cửa sổ UserForm

Để soạn thảo các thành phần

Trên cửa sổ Project Explorer, chọn thành phần cần soạn thảo

Nhấn nút [View code] để mở cửa sổ Code

Nhấn nút [View object] để mở cửa sổ UserForm

Sử dụng Option Explicit

Thêm các ActiveX:

25

Sử dụng trình soạn thảo Code: Text Editor.

**5.**

### **Làm việc với các Macro**

#### **Để mở cửa sổ Macro**

Chọn menu Tools/Macro/Macro.

Hoặc sử dụng lệnh VBARUN trên dòng commandline

Hộp thoại Macro liệt kê tất cả các hàm của bản vẽ, project (tùy theo lựa chọn tại mục [Macro in])

#### **Để chạy Macro**

Mở hộp thoại Macro

Nhấn nút [Run]

#### **Để sửa Macro**

Mở hộp thoại Macro

Nhấn nút [Edit]

#### **Để tạo Macro mới**

Trên mục [Macro name], nhập vào tên của Macro

Nhấn nút [Create]

Trên hộp thoại [Select project] chọn project để tạo Macro

**VII.**

**Căn bản về VBA.**

1.

### **Mô hình đối tượng của AutoCAD.**

Hầu như mọi Server Activex Automation cung cấp nhiều hơn một đối tượng cho các client. AutoCAD cung cấp cho các client Activex Automation khoảng 100 đối tượng, với tổng số 2500 phương thức và thuộc tính. Do vậy CAD có một khung làm việc đơn giản dễ hiểu để quản lý tất cả các đối tượng và mối quan hệ giữa chúng.

2.

### **Object Hierarchy.**

Gợi thiệu:

26

27

3.

### **Đối tượng Collection.**

AutoCAD nhóm hầu hết các đối tượng (Object) trong một collections. Cho dù các collections này chứa nhiều loại dữ liệu khác nhau, nhưng chúng có thể có chúng được tạo ra bằng cách sử dụng các kỹ thuật tương tự nhau. Mỗi một collection có một method để thêm một object vào collection. Hầu hết các collections sử dụng Add method cho mục đích này. Ví dụ, để thêm một thực thể ta sử dụng method Add<Entityname>. For example, to add a line you would use the AddLine method.

4.

### **Property & Method (thuộc tính & phương thức).**

Mỗi một Object chứa nhiều thuộc tính và phương thức khác nhau.

5.

### **Truy cập đến đối tượng trong Object Hierarchy.**

Ta có thể truy cập đến objects một cách trực tiếp hoặc thông qua một biến được định nghĩa trước.

Để truy cập đến đối tượng một cách trực tiếp.

Ví dụ, dòng code sau thêm một đường line trên model space. Chú ý rằng cây phả hệ bắt đầu bằng ThisDrawing.

```
Dim startPoint(0 To 2) As Double, endPoint(0 To 2) As Double
```

```
Dim LineObj as AcadLine
```

```
startPoint(0) = 0: startPoint(1) = 0: startPoint(2) = 0
```

```
endPoint(0) = 30: endPoint(1) = 20: endPoint(2) = 0
```

```
Set LineObj = ThisDrawing.ModelSpace.AddLine(startPoint,endPoint)
```

Để truy cập đối tượng thông qua một biến được khai báo trước. Đầu tiên ta phải khai báo biến với kiểu tương ứng, sau đó đặt biến đó lên cấp trên của đối tượng đó trong cây phả hệ.

Ví dụ: Đoạn code sau khai báo biến (moSpace) thuộc kiểu AcadModelSpace và

gán biến này cho lớp model space hiện hành:

```
Dim moSpace As AcadModelSpace
```

```
Set moSpace = ThisDrawing.ModelSpace
```

The following statement then adds a line to the model space using the userdefined

variable:

```
Dim startPoint(0 To 2) As Double, endPoint(0 To 2) As Double
```

```
Dim LineObj as AcadLine
```

```
startPoint(0) = 0: startPoint(1) = 0: startPoint(2) = 0
```

```
endPoint(0) = 30: endPoint(1) = 20: endPoint(2) = 0
```

```
Set LineObj = moSpace.AddLine(startPoint,endPoint)
```

Truy cập đến *Application Object*

Bởi vì ThisDrawing object cung cấp link tới Document object, bạn cũng có thể trở dẫn tới thư mục gốc của cây phả hệ (Application object), Chúng ta cũng có thể truy cập đến Document object trong object hierarchy thông qua đường dẫn vòng. Document object có một thuộc tính gọi là Application, nó cung cấp link tới Application object.

Ví dụ: ThisDrawing.Application.Update

## 6.

### **Truy cập đến đối tượng Collection trong Object Hierarchy.**

Hầu hết các collection objects có thể được truy cập đến trong Document object. Document object chứa thuộc tính cho mỗi đối tượng trong Collection objects. Ví dụ, Ví dụ sau định nghĩa một biến và gán nó cho tập hợp đối tượng layer của bản vẽ hiện hành:



Dim layerCollection as AcadLayers

***Set layerCollection = ThisDrawing.Layers***

Documents collection, MenuBar collection, and MenuGroups collection

28

được truy cập thông qua Application object. The Application object chứa một thuộc tính cho mỗi đối tượng nằm trong các collection này. Ví dụ, Đoạn code sau định nghĩa một biến và gán chúng cho tập hợp đối tượng MenuGroups cho ứng dụng.

Dim MenuGroupsCollection as AcadMenuGroups

Set MenuGroupsCollection = ThisDrawing.Application.MenuGroups

***Thêm một đối tượng vào Collection.***

Dim newLayer as AcadLayer

Set newLayer = ThisDrawing.Layers.Add("MyNewLayer")

***Truy cập đến một đối tượng trong collection.***

Truy cập đến một đối tượng thuộc tập hợp, sử dụng Item method. Ta có thể sử dụng tên của Item hoặc chỉ số (số thứ tự của Item) trong tập hợp. Lưu ý là Item method là method mặc định của collection. Ví dụ, 2 cách viết sau là như nhau:

ThisDrawing.Layers.Item("ABC")

ThisDrawing.Layers("ABC")

***Xoá một đối tượng khỏi Collection Object***

Sử dụng Delete method. Ví dụ, đoạn mã sau xoá layer ABC:

Dim ABCLayer as AcadLayer

Set ABCLayer = ThisDrawing.Layers.Item("ABC")

ABCLayer.Delete

7.

**Truy cập đến Object Hierarchy bằng VB, VBA trong các môi trường khác.**

Ta cũng có thể truy cập đến Object Hierarchy của cad bằng các ngôn ngữ khác như VB, VBA trong Excel, Word,.... Việc đầu tiên ta phải Tham chiếu đến AutoCAD Type Library. Để làm việc này ta làm như sau:

Chọn References option từ Project menu, hộp thoại Reference dialog hiện lên. Trong References dialog box, chọn Type library for AutoCAD, and then click OK.

Khái niệm về Type library for AutoCAD. Các đối tượng, thuộc tính và phương thức được đưa ra bởi kỹ thuật tự động hoá đối tượng có chứa một type library. Một type library là một file hoặc một phần của file miêu tả kiểu của một hoặc nhiều đối tượng. Type libraries không chứa Object; chúng chứa các thông tin về Object. Để truy cập đến type library, applications và browsers có thể xác định được đặc trưng của từng object, như là giao diện của đối tượng, tên và địa chỉ của từng thành phần của dao diện đó. Trước khi bạn sử dụng, Bạn phải tham chiếu đến type library. AutoCAD VBA sẽ tự tham chiếu đến AutoCAD type library. Đối các môi trường khác bạn phải tạo tham chiếu tới AutoCAD type library, acad16enu.tlb. File này mặc định nằm trong thư mục C:\program files\common files\autodesk shared.

Tuy nhiên, sẽ tốt hơn nếu bạn add type library reference vì những ưu điểm sau:

Chương trình sẽ chạy nhanh hơn, ổn định hơn.

Function, properties, methods có thể được kiểm tra khi bạn đánh code.

Chức năng tìm kiếm và thả xuống sau dấu chấm của các đối tượng.

```
Sub Ch2_ConnectToAcad()
```

```
Dim acadApp As AcadApplication
```

```
On Error Resume Next
```

```
Set acadApp = GetObject(, "AutoCAD.Application.16")
```

```
If Err Then
```

```
Err.Clear
```

```
Set acadApp = CreateObject("AutoCAD.Application.16")
```

```
If Err Then
```

29

```
MsgBox Err.Description
```

```

Exit Sub
End If
End If
MsgBox "Now running " + acadApp.Name + " version " +
acadApp.Version
End Sub
Dim acadDoc As AcadDocument
Set acadDoc = acadApp.ActiveDocument
.....
acadApp.visible = True

```

Nếu AutoCad đang chạy GetObject sẽ không sinh ra lỗi. Ngược lại nếu sinh ra lỗi, tức là CAD chưa chạy. Khi đó CreatObject sẽ khởi động CAD. Nếu có nhiều session của CAD đang chạy thì chương trình sẽ lấy Session đầu tiên trong Windows Running Object Table.

30

## **CHƯƠNG 2: LÀM VIỆC VỚI MÔI TRƯỜNG AUTOCAD**

### **I.**

#### **Mở, đóng và ghi lại bản vẽ**

##### **1.**

##### **Mở bản vẽ.**

Để mở bản vẽ, sử dụng phương thức Open method. Sử dụng hàm DIR của Visual Basic để kiểm tra sự tồn tại của bản vẽ trước khi mở. Bạn cũng có thể thay đổi tên và đường dẫn của bản vẽ trong đường dẫn mặc định của AutoCAD (Mục Option của CAD).

```

Sub Ch3_OpenDrawing()
Dim dwgName As String
dwgName = "c:\campus.dwg"
If Dir(dwgName) <> "" Then
ThisDrawing.Application.Documents.Open dwgName
Else
MsgBox "File " & dwgName & " does not exist."

```

End If  
End Sub

## 2.

### **Tạo mới bản vẽ.**

Tạo mới một bản vẽ, sử dụng phương thức Add.

```
Sub Ch3_NewDrawing()  
Dim docObj As AcadDocument  
Set docObj = ThisDrawing.Application.Documents.Add  
End Sub
```

## 3.

### **Lưu bản vẽ.**

Để ghi bản vẽ hiện hành. có 2 phương thức sau:

```
Sub Ch3_SaveActiveDrawing()  
ThisDrawing.Save ' ghi bản vẽ hiện hành.  
ThisDrawing.SaveAs "MyDrawing.dwg" ' ghi bản vẽ với tên mới.  
End Sub
```

Để kiểm tra xem bản vẽ đã được ghi hay chưa, ta sử dụng hàm Saved:

```
Sub Ch3_TestIfSaved()  
If Not (ThisDrawing.Saved) Then  
If MsgBox("Do you wish to save this drawing?", vbYesNo) = vbYes  
Then  
ThisDrawing.Save  
End If  
End If  
End Sub
```

## II.

### **Điều khiển cửa sổ bản vẽ**

#### 1.

### **Điều khiển của sổ AutoCAD.**

Mục đích: Khi bạn làm việc với 1 ứng dụng khác, bạn cần nhập số liệu từ người dùng chẳng hạn. Bạn cần thu nhỏ hoặc kiểm tra tình trạng của cửa sổ AutoCAD.

Sử dụng methods và properties có trong Application object, bạn có thể thay đổi position, size và visibility của cửa sổ AutoCAD. Bạn cũng có thể sử

dụng WindowState property để minimize, maximize và có thể kiểm tra tình trạng hiện tại của cửa sổ AutoCAD.

31

Các thuộc tính của Position và size của Application window là WindowTop, WindowLeft, Width, and Height properties

***Điều chỉnh cửa sổ Autocad:***

```
Sub Ch3_PositionApplicationWindow()  
ThisDrawing.Application.WindowTop = 0  
ThisDrawing.Application.WindowLeft = 0  
ThisDrawing.Application.Width = 400  
ThisDrawing.Application.Height = 400  
End Sub
```

***Maximize the Application window***

```
Sub Ch3_MaximizeApplicationWindow()  
ThisDrawing.Application.WindowState = acMax  
End Sub
```

***Minimize the Application window***

```
Sub Ch3_MinimizeApplicationWindow()  
ThisDrawing.Application.WindowState = acMin  
End Sub
```

***Lấy tình trạng của Application window***

```
Sub Ch3_CurrentWindowState()  
Dim CurrWindowState As Integer  
Dim msg As String  
CurrWindowState = ThisDrawing.Application.WindowState  
msg = Choose(CurrWindowState, "normal", "minimized", "maximized")  
MsgBox "The application window is " + msg  
End Sub
```

***Make the Application window invisible***

```
Sub Ch3_HideWindowState()  
ThisDrawing.Application.Visible = False  
End Sub
```

## 2.

### **Điều khiển của sổ bản vẽ.**

Điều khiển của sổ bản vẽ cũng giống như điều khiển cửa sổ AutoCAD, ví dụ như bạn có thể minimize, maximize, reposition, resize, và kiểm tra tình trạng của bất kỳ Document window nào.

Document window có thể minimized hoặc maximized bằng cách sử dụng thuộc tính WindowState, và bạn có thể lấy tình trạng của Document window thông qua thuộc tính WindowState.

*Ví dụ như ta gán cho Width and Height của bản vẽ hiện hành là 400 x 400 pixel.*

```
Sub Ch3_SizeDocumentWindow()  
ThisDrawing.Width = 400  
ThisDrawing.Height = 400  
End Sub
```

### ***Maximize the active Document window***

```
Sub Ch3_MaximizeDocumentWindow()  
ThisDrawing.WindowState = acMax  
End Sub
```

### ***Minimize the active Document window***

```
Sub Ch3_MinimizeDocumentWindow()  
ThisDrawing.WindowState = acMin  
End Sub
```

### ***Find the current state of the active document window***

32

```
Sub Ch3_CurrentWindowState()  
Dim CurrWindowState As Integer  
Dim msg As String
```

```

CurrWindowState = ThisDrawing.WindowState
msg = Choose(CurrWindowState, "normal", "minimized", "maximized")
MsgBox "The document window is " + msg
End Sub

```

### 3.

#### **Điều khiển sự hiển thị bên trong cửa sổ bản vẽ.**

Bạn cũng có thể thay đổi sự hiển thị bên trong cửa sổ bản vẽ bằng cách sử dụng phương thức views, viewports, và zooming. AutoCAD ActiveX cung cấp rất nhiều cách để điều khiển sự hiển thị bên trong cửa sổ bản vẽ.

Di chuyển đến các vùng khác nhau trên bản vẽ .

Phóng to thu nhỏ hay di chuyển đến các vị trí khác nhau trên bản vẽ.

Ghi lại các khung nhìn và lấy ra khi cần thiết

Hiển thị nhiều khung nhìn của một bản vẽ bằng cách sử dụng splitting the screen trong nhiều tiled viewports.

Position và Size the Document Window Use the Document object to modify the position and size of any document window. .

#### ***Define a Zoom Window***

Sử dụng ZoomWindow or ZoomPickWindow method.

```
Sub Ch3_ZoomWindow()
```

```
' ZoomWindow
```

```
MsgBox "Perform a ZoomWindow with:" & vbCrLf & "1.3, 7.8, 0" &
vbCrLf & _ "13.7, 2.6, 0" , "ZoomWindow"
```

```
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
```

```
ThisDrawing.Application.ZoomWindow point1, point2
```

```
' ZoomPickWindow
```

```
MsgBox "Perform a ZoomPickWindow" , "ZoomPickWindow"
```

ThisDrawing.Application.ZoomPickWindowEnd Sub

### III.

#### Lấy và thiết lập các thông số hệ thống

##### 1.

#### Lấy và thiết lập các biến hệ thống

Trong cây phả hệ, Document object cung cấp 2 phương thức SetVariable và GetVariable để thiết lập và lấy giá trị của các biến hệ thống của AutoCAD.

Ví dụ: ThisDrawing.SetVariable "MAXSORT", 100

##### 2.

#### Grid và Snap.

#### Sử dụng Snap, Grid Alignment

Các điều khiển Snap và Grid nằm trong class Viewport.

Sub textsnap()

Dim Vp As AcadViewport

Set Vp = ThisDrawing.ActiveViewport

Vp.SnapOn = True 'Bật chế độ snap lên

Dim newBasePoint(0 To 1) As Double

newBasePoint(0) = 1: newBasePoint(1) = 1

33

Vp.SnapBasePoint = newBasePoint ' Thay đổi gốc của snap

Dim Xspacing as Double, ySpacing as Double

xSpacing =20: ySpacing=20

VP.SetSnapSpacing(xspacing,ySpacing) ' Thay bước nhảy của chuột

Dim rotationAngle As Double

rotationAngle = 0.575

Vp.SnapRotationAngle = totationangle ' Thay đổi góc quay của Snap sang 30 độ hay 0.575 radians

' reset the viewport



```
ThisDrawing.ActiveViewport = ThisDrawing.ActiveViewport
```

```
End Sub
```

Chú ý: Với các lệnh hiệu chỉnh các active Object như active layer, active linetype, ActiveTextStyle, ta không thể dùng phương thức update như khi ta hiệu chỉnh các đối tượng thông thường được. Thay đổi các active object sẽ không được hiển thị ngay trên bản vẽ. Muốn quan sát sự thay đổi này, ta phải dùng phương thức Regen lại viewport như sau: ThisDrawing.Regen (acNameViewports)

Riêng đối với ActiveUCS hoặc ActiveViewport  
ThisDrawing.ActiveViewport = ThisDrawing.ActiveViewport. Phương thức này áp dụng được cho tất cả các active object.

Các điều khiển Grid cũng tương tự như Snap.

### **Sử dụng Ortho Mode**

```
ThisDrawing.ActiveViewport.OrthoOn = True
```

**3.**

### **Lấy và thiết lập biến hệ thống trong Option.**

Trong mục options có 9 Objects, mỗi một object đặc trưng cho một tab của Options dialog box. Các object này cung cấp đường dẫn tới tất cả các registry lưu trong options trong Options dialog box. Bạn có thể tùy biến tất cả các thiết lập của AutoCAD bằng cách sử dụng properties trong các Object này. Các Object bao gồm:

PreferencesDisplay

PreferencesDrafting

PreferencesFiles

PreferencesOpenSave

PreferencesOutput

PreferencesProfiles

PreferencesSelection

PreferencesSystem

PreferencesUser

Để truy xuất đến Preferences object, sử dụng các thuộc tính của Application object:

```
Dim acadPref as AcadPreferences
```

```
Set acadPref = ThisDrawing.Application.Preferences
```

Ví dụ sau thiết lập 2 râu của chuột dài kín màn hình:

```
Sub Ch2_PrefsSetCursor()
```

```
Dim acadPref As AcadPreferences
```

```
Set acadPref = ThisDrawing.Application.Preferences
```

```
acadPref.Display.CursorSize = 100
```

```
End Sub
```

**IV.**

### **Sử dụng command line trong VBA**

Sử dụng SendCommand method để gửi lệnh một cách trực tiếp cho AutoCAD. Phương thức SendCommand sẽ gửi một chuỗi đơn tới dòng lệnh command line. Chuỗi phải chứa các tham số cho

34

lệnh, thứ tự viết trong chuỗi như viết trong macro đã học trong phần Customize menu. . Phím cách hoặc mã ASCII tương ứng sẽ tương đương với phím ENTER trên bàn phím keyboard.

Ví dụ sau vẽ một đường tròn tâm A(2, 2, 0), bán kính R= 4. sau đó sử dụng lệnh Zoom All.

```
Sub Ch3_SendACommandToAutoCAD()
```

```
ThisDrawing.SendCommand "_Circle 2,2,0 4 "
```

```
ThisDrawing.SendCommand "_zoom a "
```

```
End Sub
```

**V.**

## Nhập dữ liệu người dùng

Các điều khiển nhập dữ liệu người dùng nằm trong class Utility.

### 1.

#### Nhập Chuỗi

GetString method prompts cho phép người dùng nhập một chuỗi từ bàn phím.

Cú pháp như sau:

***RetVal = UtilityObject.GetString(HasSpaces[, Prompt])***

***HasSpaces*** điều khiển phím spaces khi bạn nhập chuỗi.

0: Spaces Bar không cho phép (SPACEBAR sẽ kết thúc nhập chuỗi)

1: Chuỗi nhập vào có thể chứa dấu cách (ENTER sẽ kết thúc nhập chuỗi).

***Prompt*** không bắt buộc, là chuỗi sẽ được đưa ra tại dòng lệnh với mục đích thông báo.

***RetVal*** giá trị trả về, kiểu Variant với 3 tọa độ (3D).

Ví dụ:

```
Sub Ch3_GetStringFromUser()
```

```
Dim retVal As String
```

```
retVal = ThisDrawing.Utility.GetString(1, vbCrLf & "Enter your name: ")
```

```
MsgBox "The name entered was: " & retVal
```

```
End Sub
```

### 2.

#### Nhập tọa độ một điểm

GetPoint method prompts cho phép người dùng nhập tọa độ một điểm từ bàn phím hoặc bằng một kích chuột trái trên bản vẽ

Cú pháp như sau:

***RetVal = UtilityObject.GetPoint([Point][, Prompt])***

***Point*** không bắt buộc, nếu có sẽ xuất hiện dây thun từ điểm này. Point kiểu Variant với 3 tọa độ (3D).

***Prompt*** không bắt buộc, là chuỗi sẽ được đưa ra tại dòng lệnh với mục đích thông báo.

***RetVal*** giá trị trả về, kiểu Variant với 3 tọa độ (3D).

Ví dụ:

```

Sub Ch3_GetPointsFromUser()
Dim startPnt As Variant
Dim endPnt As Variant
Dim prompt1 As String
Dim prompt2 As String
prompt1 = vbCrLf & "Enter the start point of the line: "
prompt2 = vbCrLf & "Enter the end point of the line: "
' Get the first point without entering a base point
startPnt = ThisDrawing.Utility.GetPoint(, prompt1)
' Use the point entered above as the base point
endPnt = ThisDrawing.Utility.GetPoint(startPnt, prompt2)
' Create a line using the two points entered
ThisDrawing.ModelSpace.AddLine startPnt, endPnt
ThisDrawing.Application.ZoomAll

```

35

End Sub

3.

### **Nhập một ký tự điển hình cho Option**

GetKeyword Method prompts cho phép người dùng nhập một ký tự điển hình từ bàn phím cho một mục lựa chọn

Cú pháp như sau:

***RetVal = UtilityObject.GetKeyword([Prompt])***

***Prompt*** không bắt buộc, là chuỗi chứa lựa chọn sẽ được đưa ra dòng lệnh với mục đích thông báo lựa chọn.

***RetVal*** giá trị trả về, kiểu chuỗi.

Ví dụ:

```

Sub Ch3_KeyWord()
Dim keyWord As String
ThisDrawing.Utility.InitializeUserInput 1, "Line Circle Arc"

```

```
keyWord = ThisDrawing.Utility.GetKeyword(vbCrLf & "Enter an option  
(Line/Circle/Arc): ")
```

```
MsgBox keyWord, , "GetKeyword Example"
```

```
End Sub
```

4.

**Nhập số thực, số nguyên.**

**The GetInteger Method**

Phương thức GetInteger cho phép người dung nhập vào một số nguyên (giá trị nhập vào từ -32,768 to +32,767).

Cú pháp như sau:

```
intUserIntegerInput = UtilityObject.GetInteger([Prompt])
```

Ví dụ:

```
Public Sub TestGetInteger()
```

```
Dim intInput As Integer
```

```
With ThisDrawing.Utility
```

```
intInput = .GetInteger(vbCrLf & "Enter an integer: ")
```

```
.Prompt vbCrLf & "You entered " & intInput
```

```
End With
```

```
End Sub
```

**The GetReal Method**

Phương thức GetReal cho phép người dung nhập vào một số thực (giá trị nhập vào từ -32,768 to +32,767).

Cú pháp như sau:

```
dblUserRealInput = UtilityObject.GetReal([Prompt])
```

5.

**GetCorner Method, GetAngle Method, GetDistance Method**

**The GetCorner Method**

Trả về điểm góc của một hình chữ nhật

Cú Pháp như sau:

```
varUserCornerInput = UtilityObject.GetCorner(BasePoint [,Prompt])
```

**NAME**

**TYPE**

**DESCRIPTION**

BasePoint

Variant

Mảng 3 giá trị kiểu doubles mô tả điểm gốc của hình chữ nhật.

36

#### **NAME TYPE DESCRIPTION**

Prompt

String

Optional. A prompt for input.

varUserCornerInput

Variant

Giá trị trả về 3D

#### **The GetDistance Method**

Phương thức GetDistance nhập khoảng cách từ người dùng. Khác với GetReal, GetDistance có thể nhập vào một số thực, số thực này tương ứng với đơn vị đang sử dụng, hoặc ta cũng có thể pick 2 điểm trên bản vẽ. Cú pháp của phương thức như sau:

*dblUserDistanceInput* = *UtilityObject.GetDistance([BasePoint]*  
*[,Prompt])*

**NAME**

**TYPE**

**DESCRIPTION**

BasePoint

Variant

Là một mảng 3 tử kiểu double thể hiện tọa độ 3D của điểm đầu mà từ đó ta bắt đầu đo (trong WCS). Nếu bạn không cung cấp điểm này, bạn phải nhập vào 2.

Prompt

String

Chuỗi thông báo tại dòng lệnh

dblUserDistanceInput

Double

Chú ý: Hàm cho phép ta nhập vào một số âm. Nhưng khi ta pick 2 điểm trên bản vẽ thì giá trị trả về sẽ một số dương (khoảng cách giữa 2 điểm đó)

Khi ta nhập khoảng cách bằng cách pick 2 điểm, tại vị trí con chuột sẽ xuất hiện dây thun. Mặc định là tọa độ 3D. Có thể dùng là InitializeUserInput với Bit code 16 để chiếu các tọa độ sang 2D. AutoCAD sẽ tính khoảng cách 2 điểm khi đã chiếu sang 2D.

6.

## **GetEntity Method, GetSubEntity Method**

### **GetEntity Method**

Sử dụng phương thức GetEntity để chọn một đối tượng AutoCAD bằng cách pick một thực thể từ bản vẽ. Phương thức có cú pháp như sau:

*UtilityObject.GetEntity PickedEntity, PickedPoint[, Prompt]*

**NAME**

**TYPE**

**DESCRIPTION**

PickedEntity

AcadEntity object

Là dữ liệu truyền ra. Đối tượng này sẽ tham chiếu tới đối tượng vừa được pick trên bản vẽ.

PickPoint

Variant

Dữ liệu truyền ra. Là một mảng 3 phần tử kiểu double, là tọa độ điểm bạn vừa pick trong hệ tọa độ WCS.

Prompt

String

Chuỗi thông báo trên dòng lệnh

Ví dụ sau lấy một thực thể bằng cách pick đối tượng trên bản vẽ:

```
Public Sub TestGetEntity()
```

```
Dim objEnt As AcadEntity
```

```
Dim varPick As Variant
```

```
On Error Resume Next
```

```
With ThisDrawing.Utility
```

```
.GetEntity objEnt, varPick, vbCr & "Pick an entity: "
```

```
If objEnt Is Nothing Then 'check if object was picked.
```

```
.Prompt vbCrLf & "You did not pick as entity"
```

Exit Sub

37

End If

.Prompt vbCr & "You picked a " & objEnt.ObjectName

.Prompt vbCrLf & "At " & varPick(0) & "," & varPick(1)

End With

End Sub

GetEntity trả về một lỗi nếu giá trị nhập vào là *null*, như khi bạn pick không trúng một thực thể đồ họa nào cả, hoặc khi bạn ấn *Enter* mà không chọn bất kỳ một thực thể nào

### **The GetSubEntity Method**

Sử dụng GetSubEntity để nhập một thực thể phức. Một thực thể phức là một thực thể chứa nhiều thực thể đơn khác ví dụ như đường polyline, block. Phương thức có cú pháp như sau:

***UtilityObject.GetSubEntity PickedEntity, PickPoint, Matrix, Context[, Prompt]***

**NAME**

**TYPE**

**DESCRIPTION**

PickedEntity

AcadEntity

Object Output. Trả về tham chiếu tới đối tượng vừa được pick.

PickPoint

Variant

Output. Là một mảng 3 phần tử kiểu double, là tọa độ điểm bạn vừa pick trong hệ tọa độ WCS.

Matrix

Variant

Output. Trả về 1 mảng đến 4x4 phần tử kiểu doubles Chứa ma trận chuyển đổi của đối tượng vừa được chọn.

Context



Variant

Output. Trả về một mảng kiểu long integer chứa ObjectIds cho mỗi block cha, chứa các đối tượng được chọn, nếu thực thể đó là Block.

Prompt

String

Chuỗi thông báo tại dòng lệnh

Thực thể vừa chọn trong Model của block, tham số Matrix là ma trận chuyển từ hệ tọa độ block sang hệ tọa độ WCS của bản vẽ. Nó bao gồm tất cả các phép biến đổi từ một thực thể được lưu trong block sang một thực thể ở bản vẽ, như scale, rotation phép biến đổi tọa độ.

Tham số Context output là một mảng các ObjectIds của các đối tượng chứa đối tượng vừa được chọn. Ví dụ có một đường thẳng, nằm trong block có tên là BL1. BL1 lại nằm trong block cha (Block nòng) có tên là BL2. Thì Context sẽ là mảng 2 phần tử chứa ID của BL1 và BL2.

Ví dụ:

Chú ý: Thực thể nhập vào bằng GetEntity hoặc bằng GetSubEntity có thể là đối tượng ẩn (Invisible). Vì khi bạn chọn, thay vì pick object bạn có thể nhập vào ký tự L (Last Object selection). Last Object Selection có thể đã bị đóng băng hay thuộc layer vừa bị tắt đi.

38

## CHƯƠNG 3: TẠO VÀ SỬA CÁC THỰC THỂ ĐỒ HỌA

### I.

#### Tạo đối tượng bản vẽ

##### 1.

#### Xác định đối tượng chứa thực thể.

Đối tượng đồ họa có thể được tạo trong ModelSpace collection, PaperSpace Collection và Block object. Để tạo thêm một thực thể đồ họa, ta dùng phương thức Addxxx. Ví dụ, để vẽ một đường thẳng trong ModelSpace.

***Set lineObj = ThisDrawing.ModelSpace.AddLine(startPoint,endPoint)***

Cách khác, khai báo biến:

Dim moSpace As AcadModelSpace

Dim paSpace As AcadPaperSpace

Set moSpace = ThisDrawing.ModelSpace

Set paSpace = ThisDrawing.PaperSpace

***Set lineObj = moSpace.AddLine(startPoint,endPoint)***

2.

**Vẽ Line, Arc, Circle, and Ellipse objects**

**Vẽ Line**

***Set LineObject = Object.AddLine(StartPoint, EndPoint)***

**NAME**

**DATA TYPE**

**DESCRIPTION**

StartPoint

Variant

Mảng 3 phần tử kiểu Double là tọa độ 3D điểm đầu của đường thẳng trong hệ tọa độ WCS

EndPoint

Variant

Mảng 3 phần tử kiểu Double là tọa độ 3D điểm đầu của đường thẳng trong hệ tọa độ WCS

**Vẽ polyline**

***Set LWPolylineObject = Object.AddLightWeightPolyline(Vertices)***

**NAME**

**DATA TYPE**

**DESCRIPTION**

Vertices

Variant

Một mảng kiểu doubles chỉ ra danh sách tọa độ các điểm 2D trong hệ tọa độ WCS có dạng như sau(i.e., p1x, p1y, p2x, p2y, etc.). Mảng này tối thiểu phải có 4 phần tử (2 điểm).

Chú ý: Polyline không bao cao độ Z. Nếu bạn muốn vẽ tại một cao độ nào đó, bạn phải Set Elevation

## The Spline Object

***Set SplineObject = Object.AddSpline(FitPoints, StartTangent, EndTangent)***

**NAME**

**DATA TYPE**

**DESCRIPTION**

FitPoints

Variant

Là mảng một chiều kiểu double chỉ ra danh sách các điểm mà SPLine sẽ đi qua bao gồm tọa độ X, Y và Z có dạng (i.e., p1x, p1y, p1z, p2x, p2y, p2z, etc.).

StartTangent

Variant

Mảng gồm 3 phần tử kiểu doubles xác định tiếp tuyến của spline tại điểm đầu.

EndTangent

Variant

Mảng gồm 3 phần tử kiểu doubles xác định tiếp tuyến của spline tại điểm đầu.

**Vẽ cung tròn (Arc)**

***Set ArcObject = Object.AddArc(CenterPoint, Radius, StartAngle, EndAngle)***

**NAME**

**DATA TYPE**

**DESCRIPTION**

CenterPoint

Variant

Mảng 3 phần tử kiểu Double là tọa độ 3D tâm cung tròn trong hệ tọa độ WCS

Radius

Double

Bán kính

StartAngle

Double

Góc bắt đầu của cung tròn (radians), là góc hợp với trục X trong hệ tọa độ WCS

EndAngle

Double

Góc kết thúc của cung tròn (radians), là góc hợp với trục X trong hệ tọa độ WCS

40

### **The Circle Object**

*Set CircleObject = Object.AddCircle(CenterPoint, Radius)*

**NAME**

**DATA TYPE**

**DESCRIPTION**

CenterPoint

Variant

Mảng 3 phần tử kiểu Double là tọa độ 3D tâm đường tròn trong hệ tọa độ WCS

Radius

Double

Bán kính đường tròn

### **The Ellipse Object**

*Set EllipseObject = Object.AddEllipse(CenterPoint, MajorAxis, RadiusRatio)*

**NAME**

**DATA TYPE**

**DESCRIPTION**

CenterPoint

Variant

Mảng 3 phần tử kiểu Double là tọa độ 3D tâm Ellipse trong hệ tọa độ WCS.

MajorAxis

Variant

Mảng 3 phần tử kiểu Double là tọa độ 3D một điểm mô tả vector của trục lớn Ellipse tính từ tâm.

RadiusRatio

Double

Tỷ lệ chiều dài của trục bé và trục lớn:  $0 < \text{RadiusRatio} \leq 1$ .

**3.**

**Tạo các khối đặc**

41

**The Solid Object**

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

**NAME**

**DATA TYPE**

**DESCRIPTION**

Point1

Point2

Variant

Mảng 3 phần tử kiểu Double là tọa độ 3D điểm đầu, điểm cuối của đường thứ nhất trong hệ tọa độ WCS.

Point3

Point4

Variant

Mảng 3 phần tử kiểu Double là tọa độ 3D điểm đầu, điểm cuối của đường thứ hai trong hệ tọa độ WCS.

Point 1Point 3Point 4Point 2

**4.**

**Tạo đối tượng hatch**

**The Hatch Object**

*Set HatchObject = Object.AddHatch(PatternType, PatternName, Associativity)*

**NAME**

**DATA TYPE**

**DESCRIPTION**

PatternType

Long

Loại của Pattern, có 3 giá trị được liệt kê ở bảng dưới.

PatternName

String

Tên của Hatch được định nghĩa trong file \*.pat.

Associativity

Boolean

True: associative hatch.

False: disassociative hatch.

### **Bảng các giá trị của PatternType**

**CONSTANT**

**VALUE**

**DESCRIPTION**

AcHatchPatternTypeUserDefined

0

Cho phép định nghĩa Pattern dựa trên kiểu Line hiện.

AcHatchPatternTypePredefined

1

Sử dụng tên pattern từ những Pattern đã định nghĩa trước trong file acad.pat.

AcHatchPatternTypeCustomDefined

2

Sử dụng tên pattern từ những Pattern đã định nghĩa trước trong file acad.pat.

Sau khi bạn tạo Hatch object, bạn phải xác định boundary or loop bằng cách sử dụng phương thức AppendOuterLoop. Có thể xác định thêm các đảo nhỏ bên trong miền bên trong miền lớn bằng cách sử dụng phương thức AppendInnerLoop. Phương thức này có cú pháp như sau:

***Object.AppendOuterLoop loop***

***Object.AppendInnerLoop loop***

**NAME**

**DATA TYPE**

**DESCRIPTION**

Object

AcadHatch

Đối tượng Hatch

42

#### **NAME DATA TYPE DESCRIPTION**

Loop

Array of AcadEntity

Là mảng thực thể. Chú ý là các thực thể này bắt buộc phải tạo thành chu trình khép kín, nếu không Autocad sẽ báo lỗi

OuterInner

Sau khi đã định nghĩa hatch, bạn cần dùng phương thức Object.Evaluate để vẽ hatch, trong đó: Object là đối tượng hatch thuộc kiểu AcadHatch.

5.

**Tạo đối tượng Region, các phép toán trên Region**

**The Region Object**

*RegionArray = Object.AddRegion(ObjectsArray)*

**NAME**

**DATA TYPE**

**DESCRIPTION**

ObjectsArray

Array of Entity

Là mảng các thực thể. Chú ý là các thực thể này bắt buộc phải tạo thành chu trình khép kín, nếu không Autocad sẽ báo lỗi.

**The Region Boolean**

*Object.Boolean(Operation, Object1)*

**NAME**

**DATA TYPE**

**DESCRIPTION**

Operation

AcBooleanType

acUnion: nối 2 Region

acIntersection: lấy phần giao nhau của 2 Region

acSubtraction: lấy phần không giao nhau.

Region mới vừa tạo thành sẽ được gán cho biến Object.

## II.

### Thêm Text vào bản vẽ

#### 1.

##### Tạo các TextStyle

TextStyle nằm trong classe Textstyles collection.

Thêm một style dùng phương thức ThisDrawing.TextStyles.Add (NameStyle). Bạn cũng có thể chỉnh sửa thuộc tính của TextStyle. Các thuộc tính gồm:

FontFile: File font.

BigFontFile: Font shape.

Height: Chiều cao của font.

Width: Tỷ lệ chiều rộng của chữ.

ObliqueAngle: Góc nghiêng của chữ.

TextGenerationFlag: Backward text, upsidedown text, or both.

#### 2.

### Chèn Text vào bản vẽ

#### The Text Object

*Set TextObject = Object.AddText(TextString, InsertionPoint, Height)*



43

**NAME**

**DATA TYPE**

**DESCRIPTION**

TextString

String

Nội dung text

InsertionPoint

Variant

Mảng 3 phần tử kiểu Double là tọa độ 3D điểm chèn Mtext trên bản vẽ.

Height

Double

Chiều cao chữ (phải là một số dương)

**The MText Object**

*Set MTextObject = Object.AddMText(InsertionPoint, Width, TextString)*

**NAME**

**DATA TYPE**

**DESCRIPTION**

InsertionPoint

Variant

Mảng 3 phần tử kiểu Double là tọa độ 3D điểm chèn Mtext trên bản vẽ.

Width

Double

Chiều rộng của Mtext

TextString

String

Nội dung text

**3.**

**Chèn các ký tự đặc biệt, các ký tự Unicode.**

**The Special characters**

Sử dụng %%nnn để chèn ký tự đặc biệt. Ví dụ:

Dim percent as Long

percent = ASC("%")

TextString = chr(percent) + chr(percent) + "nnn"

Các Code đặc biệt như sau

%%o Toggles overscore mode on and off

%%u Toggles underscore mode on and off

%%d Draws degree symbol

%%p Draws plus and minus tolerance symbol

%%c Draws diameter dimensioning symbol

%% Draw single percent sign

### **The Unicode characters**

\U+00B0 Degree symbol

\U+00B1 Plus/minus tolerance symbol

\U+2205 Diameter dimensioning symbol

### **III.**

#### **Sửa các đối tượng bản vẽ**

AutoCAD cung cấp cho bạn một số phương thức và thuộc tính giúp bạn có thể chỉnh sửa các thực thể đồ họa. Với các thuộc tính này bạn có thể:

Copy, delete, explode, highlight, mirror, move, offset, rotate, và scale objects

Làm việc với polar và rectangular arrays

Thay đổi color, layer, linetype, và visibility của các thực thể đồ họa

Khi bạn thay đổi một thực thể đồ họa bằng code, sự thay đổi đó không hiển thị ngay trên bản vẽ cho đến khi bạn sử dụng phương thức Update của Object, hoặc sử dụng phương thức Regen của Document Object. Trong một số trường hợp, AutoCAD sẽ update trên bản vẽ khi macro hoặc program của bạn đã hoàn thành. Trong mục này, ta thường xuyên sử dụng phương thức Update như sau: DrawingObject.Update

#### **1.**

#### **Các phép sửa đổi cơ bản**

##### **Copying Objects**

Khi sử dụng phương thức Copy, đối tượng mới được tạo ra có cùng vị trí với thực thể gốc và được vẽ lên trên thực thể gốc. Cú pháp của phương thức như sau:

44

**Set DrawingObject = DrawingObject.Copy**

### **Deleting Objects**

Để xóa một thực thể ra khỏi bản vẽ, ta dùng phương thức Delete. Phương thức có cú pháp như sau:

**Object.Delete**

Phương thức Erase tương tự như phương thức Delete. Như phương thức này chỉ áp dụng cho selectionset groups. Bạn không thể áp dụng nó để xóa một Object.

### **Exploding Objects**

Sử dụng phương thức Explode để tách một thực thể gộp thành các thực thể đơn lẻ. Phương thức này trả về một mảng các objects vừa được tách ra. Phương thức có cú pháp như sau:

**varObjectArray = Object.Explode**

Lưu ý: Từ phiên bản CAD 2004 trở đi, lệnh Explode sẽ tách MText thành các đối tượng Text riêng lẻ.

### **Highlighting Entities**

Phương thức Highlight sẽ làm cho đối tượng biến thành nét đứt như khi bạn chọn đối tượng đó. Cú pháp như sau:

**Object.Highlight Highlighted**

Tham số Highlighted kiểu Boolean, Nhận giá trị (True) khi muốn Object Highlight và (False) nếu muốn đối tượng đó trở về trạng thái bình thường.

### **Mirroring Objects**

Cú pháp như sau:

**Set DrawingObject = DrawingObject.Mirror(Point1, Point2)**

**NAME**

**DATA TYPE**

**DESCRIPTION**

Point1

Variant

Mảng 3 phần tử kiểu Double là tọa độ 3D điểm thứ nhất của trục đối xứng.

Point2

Variant

Mảng 3 phần tử kiểu Double là tọa độ 3D điểm thứ hai của trục đối xứng.

### **Moving Objects**

45

#### **DrawingObject.Move Point1, Point2**

**NAME**

**DATA TYPE**

**DESCRIPTION**

Point1

Variant

Mảng 3 phần tử kiểu Double là tọa độ 3D điểm thứ nhất của Vector dịch chuyển trong hệ tọa độ WCS.

Point2

Variant

Mảng 3 phần tử kiểu Double là tọa độ 3D điểm thứ hai của Vector dịch chuyển trong hệ tọa độ WCS.

### **Rotating Objects**

#### **DrawingObject.Rotate BasePoint, RotationAngle**

**NAME**

**DATA TYPE**

**DESCRIPTION**

BasePoint

Variant

Mảng 3 phần tử kiểu Double là tọa độ 3D điểm tâm quay trong hệ tọa độ WCS. Điểm này phải cùng tọa độ Z với thực thể gốc

RotationAngle

Double

Góc quay tính bằng radian.

### **Scaling Objects**

**DrawingObject.ScaleEntity BasePoint, ScaleFactor**

**NAME**

**DATA TYPE**

**DESCRIPTION**

46

**NAME DATA**

**TYPE**

**DESCRIPTION**

BasePoint

Variant

Mảng 3 phần tử kiểu Double là tọa độ 3D điểm tâm phóng trong hệ tọa độ WCS.

ScaleFactor

Double

Giá trị dương, là tỷ lệ tương đối kích thước của hình mới cho hình cũ.

2.

### **Các phép biến đổi nâng cao**

#### **Offsetting Objects**

Phương thức này có thể áp dụng cho các thực thể như Arc, Circle, Ellipse, Line, LightweightPolyline, Polyline, Spline, và Xline. Phương thức này trả về một mảng các thực thể mới được tạo.

**varObjectArray = Object.Offset(OffsetDistance)**

Tham số OffsetDistance kiểu double khác không thể hiện hướng và khoảng cách Offset. Giá trị âm nghĩa là offset tạo một Object bé hơn thực thể gốc. Đối với đường thẳng, giá trị âm thể hiện hướng của thực thể mới so với thực thể gốc.

## Object Arrays

Use the `ArrayPolar` and `ArrayRectangular` methods to create an array of objects based on an existing object. Both methods copy the base object into a regular pattern at a specified distance from one another.

47

### Creating a Polar Array of Objects

```
varObjectArray = DrawingObject.ArrayPolar (NumberOfObjects,  
AngleToFill, ArrayCenter)
```

**NAME**

**DATA TYPE**

**DESCRIPTION**

NumberOfObjects

Long

Số lượng các thực thể phải lớn hơn 1.

AngleToFill

Double

Giá trị khác không tính bằng radians.

ArrayCenter

Variant

Mảng 3 phần tử kiểu Double là tọa độ 3D điểm tâm array trong hệ tọa độ WCS.

### Creating a Rectangular Array of Objects

```
varObjectArray = DrawingObject.ArrayRectangular  
(NumberOfRows, NumberOfColumns, NumberOfLevels,  
DistBetweenRows, DistBetweenColumns, DistBetweenLevels)
```

**NAME**

**DATA TYPE**

**DESCRIPTION**

NumberOfRows

Long

Số dương, là số lượng dòng trong rectangular array. Nếu giá trị là 1, thì NumberOfColumns phải lớn hơn 1.

NumberOfColumns

Long

Số dương, là số lượng cột trong rectangular array. Nếu giá trị là 1, thì NumberOfRows phải lớn hơn 1.

NumberOfLevels

Long

Số dương, là số lượng các cấp theo trục Z trong rectangular 3D array.

DistBetweenRows

Double

Khoảng cách giữa các dòng. Giá trị này dương thì dòng sẽ mở rộng theo hướng lên trên. Ngược lại, nếu là giá trị âm, dòng sẽ được mở rộng xuống dưới. Nếu bằng không, đối tượng sẽ được vẽ lên trước thực thể gốc.

DistBetweenColumns

Double

Khoảng cách giữa các cột. Giá trị này dương thì cột sẽ mở rộng theo hướng sang trái. Ngược lại, nếu là giá trị âm, dòng sẽ được mở rộng sang phải. Nếu bằng không, đối tượng sẽ được vẽ lên trước thực thể

48

**NAME DATA**

**TYPE**

**DESCRIPTION**

gốc.

DistBetweenLevels

Double

Khoảng cách giữa các Level theo trục Z. Giá trị này dương thì dòng sẽ mở rộng theo hướng lên trên. Ngược lại, nếu là giá trị âm, dòng sẽ được mở rộng xuống dưới. Nếu bằng không, đối tượng sẽ được vẽ lên trước thực thể gốc.

Nếu bạn làm việc trong 2D, tốt nhất bạn đặt tham số NumberOfLevels bằng 1.

**Transform Objects**

Bạn có thể move, scale, hoặc rotate một object thông qua ma trận chuyển đổi  $4 \times 4$  bằng cách sử dụng phương thức TransformBy. Cú pháp của phương thức như sau:

### **anObj.TransformBy tMatrix**

Ma trận này có dạng như sau:

R00 R01 R02 T0

R10 R11 R12 T1

R20 R21 R22 T2

0 0 0 1

Trong đó R = Rotation và T = Translation:

Ví dụ: ma trận quay một thực thể một góc 90 độ sẽ có dạng như sau:

tMatrix(0,0) = 0.0 tMatrix(0,1) = 1.0 tMatrix(0,2) = 0.0

tMatrix(0,3) = 0.0 tMatrix(1,0) = 1.0 tMatrix(1,1) = 0.0

tMatrix(1,2) = 0.0 tMatrix(1,3) = 0.0 tMatrix(2,0) = 0.0

tMatrix(2,1) = 0.0 tMatrix(2,2) = 1.0 tMatrix(2,3) = 0.0

tMatrix(3,0) = 0.0 tMatrix(3,1) = 0.0 tMatrix(3,2) = 0.0

tMatrix(3,3) = 1.0

## **3.**

### **Chỉnh sửa PolyLine, SpLine**

#### **Edit Polylines**

49

Bạn có thể chỉnh sửa

Closed property: Opens hoặc closes một polyline.

Coordinates property: Thay đổi từng đỉnh của một polyline.

AddVertex method: Thêm một đỉnh vào polyline.



Sử dụng các phương thức sau để cập nhật chỗ lồi lõm và chiều rộng của một polyline:

**SetBulge:** Thiết lập độ cong của polyline.

**SetWidth:** Thiết lập bề rộng của điểm đầu và điểm kết thúc polyline.

### **Edit Splines**

Với splines bạn có thể thay đổi các thuộc tính sau:

**Closed:** Open hoặc close đường spline.

**ControlPoints:** Điều khiển các điểm của spline.

**EndTangent:** Điều khiển pháp tuyến điểm cuối của spline.

**FitPoints:** Specifies all the fit points of a spline.

**FitTolerance:** Thau đổi tolerance cho Spline.

**Knots:** Điều chỉnh nút vector cho spline.

**StartTangent:** Điều khiển pháp tuyến điểm đầu của spline.

Ngoài ra, bạn còn có thể thêm mới đỉnh,... như sau

**AddFitPoint:** Thêm một fit point vào Spline tại vị trí Index.

**DeleteFitPoint:** Xóa một fit point khỏi spline tại vị trí Index.

**GetFitPoint:** Lấy điểm fit point của spline tại vị trí Index.

**Reverse:** nghịch đảo hướng của Spline.

**SetControlPoint:** Thiết lập control point của spline tại vị trí index.

**SetFitPoint:** thiết lập fit point của spline tại vị trí index.

SetWeight: Thiết lập weight của control point tại vị trí index.

Các thuộc tính chỉ đọc:

Area: Diện tích khép kín của spline.

NumberOfControlPoints: số lượng các điểm điều khiển của spline.

NumberOfFitPoints: Số lượng các điểm Fit của spline.

#### **IV.**

#### **Block và thuộc tính của block**

Trong mục này chúng ta sẽ nghiên cứu

Tạo Block objects

Chèn Block object vào trong bản vẽ AutoCAD

Làm việc với external reference file

Tạo Attribute object

Chèn Block objects có chứa attribute

#### **1.**

#### **Blocks và Block References**

Block object đại diện cho một block definition, nó chứa tên và một tập hợp các thực thể đồ họa. Block objects bao gồm 2 loại:

Một Block definition là một kiểu dữ liệu trừu tượng định nghĩa các thực thể đồ họa nằm trong Block.

Một Block reference (hay block insertion), là những vị trí mà ta chèn Block vào bản vẽ.

Thay đổi một block definition tương đương với việc tắt hay đổi tất cả các block reference trên bản vẽ.

Có 3 loại block:

**Simple block:** Là những Block đơn giản được định nghĩa từ các thực thể có sẵn trên bản vẽ hoặc blok được định nghĩa bằng cách bạn insert một bản vẽ khác vào trong bản vẽ hiện hành.

50

**Externally referenced block:** đó là kiểu tham khảo ngoài, bạn có thể chỉnh sửa chúng bằng lệnh *REFEDIT*.

**Layout block:** . chứa dạng hình học của một Layout object.

Sử dụng thuộc tính *IsLayout* và *IsXRef* để nhận dạng kiểu của block definition. Nếu cả hai thuộc tính trên đều trả về giá trị *False*, thì Block object sẽ là simple block.

### **Truy cập Block Objects**

AutoCAD Document objects có Blocks collection chứa tất cả các Block definition objects trong bản vẽ. Ví dụ sau đây truy cập đến các block object:

```
Dim objBlocks As AcadBlocks
```

```
Set objBlocks = ThisDrawing.Blocks
```

```
MsgBox "There are " & objBlocks.Count & " Block objects"
```

Tham chiếu đến Block object có sẵn trong bản vẽ, ta sử dụng phương thức *Item*, đây cũng là phương thức mặc định của Blocks collection.

```
Dim objBlock As AcadBlock
```

```
Set objBlock = ThisDrawing.Blocks.Item(Index)
```

```
Set objBlock = ThisDrawing.Blocks.Item(NameBlock)
```

### **Duyệt qua các Block Object của Blocks Collection**

```
Public Sub ListBlocks()
```

```
Dim objBlock As AcadBlock
```

```
Dim strBlockList As String
```

```
strBlockList = "List of blocks: "
```

```
For Each objBlock In ThisDrawing.Blocks
```

```
strBlockList = strBlockList & vbCr & objBlock.Name
```

```
Next
```

```
MsgBox strBlockList
```

```
End Sub
```

## **Tạo Blocks**

### **Sử dụng phương thức Add**

Phương thức Add của Blocks collection dùng để thêm một Block object mới vào bản vẽ. Phương thức này sẽ trả về một *Simple block*. Phương thức có cú pháp như sau

```
Set BlockObject = BlocksCollection.Add(InsertionPoint, BlockName)
```

**NAME**

**DATA TYPE**

**DESCRIPTION**

InsertionPoint

Variant

Mảng 2 phần tử kiểu doubles mô tả tọa độ điểm chèn trong hệ tọa độ WCS.

BlockName

String

Tên của Block mới

### **AddXXX Methods**

Sử dụng phương thức AddXXX để thêm các thực thể đồ họa vào Block object giống như thêm các thực thể vào bản vẽ.

### **CopyObject Method**

Một cách khác để thêm một Object vào trong Block object đó là dùng phương thức Document object's CopyObject. Phương thức này sẽ nhân bản thêm một thực thể nữa. Cú pháp như sau:

51

```
varCopies = Owner.CopyObjects(Objects [, NewOwner] [, IdMap])
```

**NAME**

## **DATA TYPE**

## **DESCRIPTION**

Owner

Document, PaperSpace, ModelSpace or Block objects

Đối tượng chứa Object cần copy.

Objects

Variant

Mảng objects để copy. Các Object này phải thuộc Owner object.

NewOwner

Variant

Đích tới của phương thức copy. Nếu là null thì sẽ Copy vào Owner object.

IdMap

Variant

Xem IDPair objects.

### **Đổi tên Block Object**

Để đổi tên Block, bạn gán một chuỗi mới cho thuộc tính *Name*. Khi thay đổi tên thì tên của các Block reference sẽ tự động thay đổi theo.

Chú ý: Khi bạn thay đổi Layout Block hoặc khi bạn thay đổi những Block không có tên cụ thể (bắt đầu bằng dấu ‘\*’) có thể phá hỏng AutoCad.

### **Xóa Block Object**

#### **BlockObject.Delete**

Chú ý: Bạn không thể xóa Block khi:

**Có BlockReference object tham chiếu đến nó.** Nếu một BlockReference object tham chiếu đến block definition, bạn không thể xóa chúng. Để xóa Block definition này, bạn phải sử dụng phương thức Purge trong Document Object.

**Khi nó là một Xref.** Bạn cũng không cần thiết phải xóa nó vì thực chất AutoCAD không lưu trữ Xref trong bản vẽ.

### **The InsertBlock Method**

Như đã nói ở trên, một BlockReference object được tạo ra khi:

Bạn chèn một Block vào bản vẽ từ Block Definition

Bạn chèn một block vào bản vẽ từ một bản vẽ khác ngoài đĩa của bạn

## **Thao tác với Block**

### **The Item Method**

Sử dụng phương thức Item để truy cập cũng như duyệt qua các đối tượng trong block definition object. Phương thức có cú pháp như sau:

**Set objEntity = BlockObject.Item(Index)**

Index là vị trí của thực thể trong Block. Item là phương thức mặc định của BlockObject.

**Set objEntity = BlockObject(Index)**

### **The InsertBlock Method**

Sử dụng InsertBlock để thêm một BlockReference object vào bản vẽ hoặc vào một Block lồng. Cú pháp như sau:

**Set BlockReferenceObject = Object.InsertBlock(InsertionPoint, BlockName, \_**

**Xscale, Yscale, ZScale, RotationAngle)**

**NAME**

**DATA TYPE**

**DESCRIPTION**

52

**NAME DATA**

**TYPE**

**DESCRIPTION**

InsertionPoint

Variant

Mảng 3 phần tử kiểu Double là tọa độ điểm chèn vào đối tượng (đối tượng có thể là Block hay Document Object).

BlockName

String

Tên của Block object trong Blocks collection, hoặc đường dẫn và tên file của bản vẽ chèn vào bản vẽ hiện hành.

Xscale

Double

Là tỷ lệ phóng theo trục X, giá trị phải khác không. Nếu nhận giá trị âm thì sẽ lấy đối xứng theo trục đó tại điểm chèn.

Yscale

Double

Là tỷ lệ phóng theo trục Y, giá trị phải khác không. Nếu nhận giá trị âm thì sẽ lấy đối xứng theo trục đó tại điểm chèn.

Zscale

Double

Là tỷ lệ phóng theo trục Z, giá trị phải khác không. Nếu nhận giá trị âm thì sẽ lấy đối xứng theo trục đó tại điểm chèn.

RotationAngle

Double

Góc quay tương đối hợp với trục X trong hệ tọa độ WCS, tính bằng radian.

### **Deleting a Block Reference**

Giống như các Object khác, bạn sử dụng phương thức Delete để xóa block references. Phương thức chỉ xóa BlockReference object, không xóa Block definition object. Cú pháp như sau:

**BlockReferenceObject.Delete**

### **The Explode Method**

**varArray = BlockReferenceObject.Explode**

Chú ý:

Phương thức chỉ có ở BlockReferenceObject

Phương thức sẽ tạo ra một bản copy của block definition và vẫn để lại block reference chưa phá vỡ. Bạn phải xóa nó nếu bạn không muốn sử dụng chúng.

### **Ghi Block ra file (phương thức Wblock)**

Đối tượng Document object có phương thức ghi lại những gì có trong một SelectionSet object vào đĩa thành một bản vẽ mới. Bạn có thể nhập file này như một block definition bằng cách sử dụng phương thức InsertBlock. Cú pháp phương thức như sau:

**DocumentObject.WBlock FileName, SelectionSet**

**NAME**

## **DATA TYPE**

## **DESCRIPTION**

FileName

String

Tên của file sẽ được ghi. Bạn cũng không cần thiết phải ghi rõ cả tên phần mở rộng của file, phương thức sẽ sử dụng phần mở rộng là .dwg.

SelectionSet

SelectionSet object

selection set chứa các thực thể cần ghi ra file.

Phương thức mặc định sử dụng điểm gốc của bản vẽ mới là gốc tạo độ WCS của bản vẽ hiện hành. Bạn có thể sửa lại trước khi bạn sử dụng Wblock bằng cách sau:

```
Public Sub TestWBlock()
```

```
Dim objSS As AcadSelectionSet
```

```
Dim varBase As Variant
```

53

```
Dim dblOrigin(2) As Double
```

```
Dim objEnt As AcadEntity
```

```
Dim strFilename As String
```

```
'choose a selection set name that you only use as temporary storage and
```

```
'ensure that it does not currently exist
```

```
On Error Resume Next
```

```
ThisDrawing.SelectionSets("TempSSet").Delete
```

```
Set objSS = ThisDrawing.SelectionSets.Add("TempSSet")
```

```
objSS.SelectOnScreen
```

```
With ThisDrawing.Utility
```

```
.InitializeUserInput 1
```

```
strFilename = .GetString(True, vbCr & "Enter a filename: ")
```

```
.InitializeUserInput 1
```

```
varBase = .GetPoint(, vbCr & "Pick a base point: ")
```



```

End With
" WCS origin
dblOrigin(0) = 0: dblOrigin(1) = 0: dblOrigin(2) = 0
" move selection to the origin
For Each objEnt In objSS
objEnt.Move varBase, dblOrigin
Next
" wblock selection to filename
ThisDrawing.Wblock strFilename, objSS
" move selection back
For Each objEnt In objSS
objEnt.Move dblOrigin, varBase
Next
" clean up selection set
objSS.Delete
End Sub

```

### **Using MInsertBlock Objects**

**Set MInsertBlockObject = Object.AddMInsertBlock(InsertionPoint, BlockName, XScale, YScale, ZScale, RotationAngle, Rows, Columns, RowSpacing, ColumnSpacing)**

**NAME**

**DATA TYPE**

**DESCRIPTION**

InsertionPoint

Variant

Tọa độ điểm chèn.

BlockName

String

Tên của Block object trong Blocks collection, hoặc đường dẫn đầy đủ của bản vẽ có sẵn trên đĩa.

Xscale

Double

Tỉ lệ phóng theo phương X. Phải nhận giá trị khác 0. Giá trị âm sẽ lấy đối xứng theo phương X tại điểm chèn.

Yscale

Double

Tỉ lệ phóng theo phương Y. Phải nhận giá trị khác 0. Giá trị âm sẽ lấy đối xứng theo phương Y tại điểm chèn.

54

## **NAME DATA**

### **TYPE**

### **DESCRIPTION**

Zscale

Double

Tỉ lệ phóng theo phương Z. Phải nhận giá trị khác 0. Giá trị âm sẽ lấy đối xứng theo phương Z tại điểm chèn.

RotationAngle

Double

The rotation angle relative to the WCS Xaxis, expressed in radians.

Rows

Long

Số dương, là số lượng dòng.

Columns

Long

Số dương, là số lượng cột.

RowSpacing

Double

Số khác không, là khoảng cách giữa các dòng. Giá trị âm sẽ tạo ra các dòng theo hướng ngược với trục X.

ColumnSpacing

Double

Số khác không, là khoảng cách giữa các dòng. Giá trị âm sẽ tạo ra các dòng theo hướng ngược với trục Y.

## **External References**

External references, or *Xrefs*, are blocks that are not permanently loaded into the current drawing file. Instead, Xrefs refer to an external drawing file for their geometry (hence their name).

External references share many properties and methods with simple blocks, and for many purposes you can treat them as simple blocks. But sometimes you might also need to use external references' special capabilities. This section explains the following Xref methods:

Attaching and detaching

Loading and unloading

Binding

### **Attaching External References**

The `AttachExternalReference` method works much like `InsertBlock`, except that the resulting entity is an external reference instead of a block reference. Just like `InsertBlock`, the `PaperSpace`, `ModelSpace`, and `Block` objects expose this method and let you specify the insertion point, scale, and rotation angle in the drawing.

```
Set ExternalReferenceObject =  
Object.AttachExternalReference(FileName, _  
BlockName, InsertionPoint, Xscale, Yscale, Zscale, RotationAngle,  
Overlay)
```

[Table 137](#) explains this method's parameters.

Table 137: `AttachExternalReference` Method Parameters

#### **NAME**

#### **DATA TYPE**

#### **DESCRIPTION**

FileName

String

The external AutoCAD drawing file's name. You must specify the `.dwg` extension. Optionally, you can specify a path to the file. If you don't, AutoCAD tries to find the file in the system search path.

BlockName

String

A name for the internal `Block` object that will point to the external drawing file.

InsertionPoint

Variant

A threeelement array of doubles that specifies the 3D WCS coordinates where the Xref will be inserted into the Object.

Xscale

Double

A nonzero number representing the scaling factor for the Xref's X direction. Negative numbers mirror the insertion on this axis.

Yscale

Double

A nonzero number representing the scaling factor for the Xref's Y direction.

55

Table 137: AttachExternalReference Method Parameters

**NAME DATA**

**TYPE**

**DESCRIPTION**

Negative numbers mirror the insertion on this axis.

Zscale

Double

A nonzero number representing the scaling factor for the Xref's Z direction. Negative numbers mirror the insertion on this axis.

RotationAngle

Double

The rotation angle relative to the WCS Xaxis, expressed in radians.

Overlay

Boolean

Controls how the Xref is attached. If True, the Xref is brought in as an overlay. Overlay external references aren't visible if the current drawing is attached as an Xref to another drawing. In this way, overlay Xrefs can reduce the need to detach Xrefs before sharing drawings. If this parameter is False, the

Xref is an attachment.

The following example creates an Xref based on user input:

### **Detaching External References**

You can detach an external reference from the current drawing using the Block object's Detach method. It has this syntax:

```
BlockObject.Detach
```

Note

You detach an Xref's block definition, the method removes all associated ExternalReference objects from the drawing too. This includes linetypes, textstyles, dimstyles, nested block definitions, and layers.

### **Unloading External References**

You can also unload external references without detaching them from the current drawing. Use the Block object's Unload method. It has this syntax:

```
BlockObject.Unload
```

Though not visible, unloaded Xrefs are still associated with the current drawing. To regenerate them, reload them.

### **Reloading External References**

Use the Block object's Reload method to reload an external reference whenever you want, even if the Xref is already loaded. Reload an alreadyloaded Xref when you modify the underlying drawing and then want to update the inmemory copy in the current drawing. This method has the following syntax:

```
BlockObject.Reload
```

### **Binding External References**

Use the Block object's Bind method to convert external references to simple blocks. This operation builds an internal copy of the external drawing file in much the same way the InsertBlock method does using an external filename. Instead of referring to the external drawing database, Bind converts any former ExternalReference objects to simple block references. This method has the following syntax:

```
BlockObject.Bind(Merge)
```

This method has one parameter, Merge, a Boolean. When it's True, the method merges dependent symbol table entries in the external file with the current drawings entries. When it's False, the method prefixes them to avoid

collision with any other entry name in the current drawing. The prefix has the form BlockName\$X\$EntryName, where

BlockName is the block definition name for the current drawing's external reference

X is an automatically generated integer that makes the name unique in the current drawing

EntryName is the name of the symbol table entry in the externally referenced drawing file.

Note

If Merge is set to True and an entry is already present in the current drawing, the method maps the external entry to the current drawing entry. This is identical behavior to inserting block definitions that contain duplicate layers, linetypes, or textstyles in the current drawing.

The following example binds the specified external reference using either style

V.

**Chọn đối tượng**

1.

**Tạo Selectionset.**

Selectionset nằm trong class Selections collection.

**Thêm một Selectionset**

<b>Set</b>	<b>SelectionSetObject</b>	<b>=</b>
<b>SelectionSetsCollection.Add(SelectionSetName)</b>		

**NAME**

**DATA TYPE**

**DESCRIPTION**

SelectionSetName

String

Tên của Selectionset, tên này phải không được trùng với các Selectionset đã có.

**Truy suất đến collection**

Nếu bạn có nhiều Selectionset trong selections collection. Bạn có thể truy cập đến từng selectionset theo tên hoặc số thứ tự (Index) của nó. Ví dụ như sau:

```
Dim objSelections As AcadSelectionSets
Set objSelections = ThisDrawing.SelectionSets
Dim objSelection As AcadSelectionSet
Set objSelection = objSelections.Item(2)
Set objSelection = objSelections.Item("My SelectionSet")
```

Chỉ số của Selection nằm trong khoảng từ 0 đến (SelectionSets.Count - 1). Trong Autocad Selection collection thì phương thức Item là phương thức mặc định. Do vậy 2 ví dụ ở trên có thể viết lại như sau:

```
Set objSelection = objSelections(2)
Set objSelection = objSelections("My SelectionSet")
```

## 2.

### **Thêm đối tượng vào selection set**

57

### **Selecting Entities**

Khi bạn tạo mới một SelectionSet thì đó sẽ là một SelectionSet rỗng. Để thêm các đối tượng vào SelectionSet ta dùng phương thức SelectXXX. Giống như trong AutoCAD, ta có VBA cho phép ta chọn các đối tượng theo: chọn từng đối tượng (pick vào đối tượng), chọn theo Window, theo fence (ấn chữ f, và chọn 1 đường thẳng cắt qua các đối tượng được chọn) hoặc theo một đa giác (polygon).

#### **The Select Method**

Phương thức Select cho phép bạn thêm đối tượng vào selection set, ngoài ra nó cũng cho phép bạn sử dụng Last SelectionSet. Phương thức có cú pháp như sau:

```
SelectionSetObject.Select Mode [, Point1, Point2] [, FilterCodes, FilterValues]
```

**NAME**

**DATA TYPE**

**DESCRIPTION**

Mode

Long

Kiểu chọn, được liệt kê ở bảng dưới.

Point1

Variant

Mảng 3 phần tử kiểu Double là 3 tọa độ của góc đầu tiên của hình chữ nhật.

Point2

Variant

Mảng 3 phần tử kiểu Double là 3 tọa độ của góc đầu tiên của hình chữ nhật. Point1, Point2 thì phải dùng đồng thời với nhau.

FilterCodes

Variant

Dùng để lọc thực thể. Nghiên cứu trong mục sau.

FilterValues

Variant

Dùng để lọc thực thể. Nghiên cứu trong mục sau.

Bảng các kiểu chọn

**CONSTANT**

**VALUE**

**DESCRIPTION**

acSelectionSetWindow

0

Tất cả các thực thể nằm trong cửa sổ được chỉ ra bởi điểm Point1 và Point2 được chọn.

acSelectionSetCrossing

1

Tất cả các thực thể nằm trong hoặc cắt Window được chỉ ra bởi điểm Point1 và Point2 được chọn.

acSelectionSetPrevious

3

Selection set gần nhất sẽ được chọn. Point1 và Point2 không được sử dụng.

acSelectionSetLast

4

Thực thể được tạo cuối cùng tính đến thời điểm chọn sẽ được chọn.



Point1 và Point2 không được sử dụng.

```
acSelectionSetAll
```

```
5
```

Chọn tất cả các thực thể. Point1 và Point2 không được sử dụng.

Lưu ý là phương thức Select có thể chọn tất cả các thực thể thuộc tất cả các layer ngay cả khi nó được đóng bằng hay được khóa.

Ví dụ:

```
Public Sub TestSelectionSetFilter()
```

```
Dim objSS As AcadSelectionSet
```

```
Dim intCodes(0) As Integer
```

```
Dim varCodeValues(0) As Variant
```

```
Dim strName As String
```

```
On Error GoTo Done
```

```
58
```

```
With ThisDrawing.Utility
```

```
strName = .GetString(True, vbCr & "Layer name to filter: ")
```

```
If "" = strName Then Exit Sub
```

```
" create a new selectionset
```

```
Set objSS = ThisDrawing.SelectionSets.Add("TestSelectionSetFilter")
```

```
" set the code for layer
```

```
intCodes(0) = 8
```

```
" set the value specified by user
```

```
varCodeValues(0) = strName
```

```
" filter the objects
```

```
objSS.Select acSelectionSetAll, , , intCodes, varCodeValues
```

```
" highlight the selected entities
```

```
objSS.Highlight True
```

```
" pause for the user
```

```
.Prompt vbCr & objSS.Count & " entities selected"
```

```

.GetString False, vbLf & "Enter to continue "
" dehighlight the entities
objSS.Highlight False
End With
Done:
" if the selection was created, delete it
If Not objSS Is Nothing Then
objSS.Delete
End If
End Sub

```

### **The SelectOnScreen Method**

Đây là kiểu chọn chuẩn của Autocad như Window, Crossing và Last.

**SelectionSetObject.SelectOnScreen [, FilterCodes, FilterValues]**

### **The SelectAtPoint Method**

Cho phép chọn một thực thể bằng cách chọn 1 điểm.

**SelectionSet.SelectAtPoint Point [,FilterCodes, FilterValues]**

**NAME**

**DATA TYPE**

**DESCRIPTION**

Point

Variant

Mảng 3 phần tử kiểu Double là 3 tọa độ một điểm mà thực thể được chọn đi qua.

### **The SelectByPolygon Method**

**SelectionSetObject.SelectByPolygon Mode, Vertices [, FilterType, FilterData]**

59

**NAME**

**DATA TYPE**

## **DESCRIPTION**

Mode

Long

Kiểu chọn, được liệt kê ở bảng dưới.

Vertices

Variant

Là mảng một chiều kiểu double chỉ ra danh sách các điểm mà đa giác sẽ đi qua bao gồm tọa độ X, Y và Z trong WCS có dạng (i.e., p1x, p1y, p1z, p2x, p2y, p2z, etc.).

Bảng giá trị Modes

## **CONSTANT**

### **VALUE**

## **DESCRIPTION**

acSelectionSetFence

2

Vertices sẽ mô tả các đường thẳng liên tiếp nhau. Tất cả các thực thể cắt đường thẳng này đều được chọn. Vertices chứa ít nhất 2 điểm

acSelectionSetWindowPolygon

6

Vertices mô tả các đỉnh một đa giác. Tất cả các thực thể nằm trong đa giác này sẽ được chọn. Vertices chứa ít nhất 3 điểm.

acSelectionSetCrossingPolygon

7

Vertices mô tả các đỉnh một đa giác. Tất cả các thực thể nằm trong hoặc cắt qua đa giác này sẽ được chọn. Vertices chứa ít nhất 3 điểm.

## **Adding and Removing Items**

**SelectionSetObject.AddItem(Entities)**

**SelectionSetObject.RemoveItem(Entities)**

### **NAME**

### **DATA TYPE**

## **DESCRIPTION**

Entities

Array of AcadEntity objects

Mảng các thực thể thêm hay bớt đi khỏi tập chọn.

## The Clear, Delete, and Erase Methods

### SelectionSetObject.Clear

Phương thức *Clear* xóa tất cả các đối tượng trong SelectionSet object. Nhưng các thực thể nằm trong Selection set vẫn còn trong bản vẽ. Cú pháp như sau:

### SelectionSetObject.Delete

Phương thức *Delete* dùng để xóa một SelectionSet object trong SelectionSets collection. Thực thể nằm trong SelectionSet đó vẫn tồn tại trong bản vẽ.

### SelectionSetObject.Erase

Phương thức *Erase* xóa tất cả các thực thể nằm trong SelectionSet object và xóa khỏi bản vẽ. Các thực thể bị xóa khỏi bản vẽ, nhưng SelectionSet object vẫn tồn tại trong Selection Collection và ta có thể thêm các thực thể mới vào trong chúng.

## 3.

### Lọc đối tượng trong selection set

#### Selection Set Filters

Bạn có thể sử dụng 2 tham số FilterCodes and FilterValues để có một số điều kiện lọc. Ví dụ ta chỉ chọn các đối tượng có màu, layer hay litype,... theo ý muốn.

60

**FilterCodes** là một mảng kiểu integers, là Group DXF code. Bạn có thể xem thêm AutoCAD DXF. Sau đây một số code hay dùng nhất:

#### **DXF code**

#### **Filter type**

0

Object Type (String) như "Line," "Circle," "Arc," ...

2

Object Name (String)

8

Layer Name (String) như "Layer 0."

60

Object Visibility (Integer). Gồm 0 = visible, 1 = invisible.

62

Color Number (Integer) Là một số từ 0 tới 256. Giá trị 0:BYBLOCK. Giá trị 256 là BYLAYER. Giá trị âm là layer tắt.

67

Chỉ số Model/paper space (Integer). Giá trị 0 hoặc không gán = model space, 1 = paper space.

**FilterValues** là một mảng kiểu variant, Mỗi phần tử của mảng *FilterValues* phải tương ứng với một phần tử của mảng *FilterCodes*. Do vậy hai mảng này phải có cùng chiều dài.

Ví dụ:

```
FilterType(0) = 0
```

```
FilterData(0) = "Circle"
```

```
Sub Ch4_FilterMtext()
```

```
Dim sstext As AcadSelectionSet
```

```
Dim FilterType(0) As Integer
```

```
Dim FilterData(0) As Variant
```

```
Set sstext = ThisDrawing.SelectionSets.Add("SS2")
```

```
FilterType(0) = 0
```

```
FilterData(0) = "Circle"
```

```
sstext.SelectOnScreen FilterType, FilterData
```

```
End Sub
```

Khi bạn sử dụng nhiều hơn 2 điều kiện lọc. Bạn phải sử dụng các phép toán Logic

Đi kèm với các toán tử lọc là các group code đặc biệt là “-4”. Các cặp toán tử có thể sử dụng là.

**FILTER OPERATOR**

**START AND END VALUE**

**NUMBER OF OPERANDS**

"<AND"

... "AND>"

One or more

"<OR"

... "OR>"

One or more

"<XOR"

... "XOR>"

Exactly two

"<NOT"

... "NOT>"

Exactly one

Ví dụ:

```
Public Sub TestSelectionSetOperator()
```

```
Dim objSS As AcadSelectionSet
```

```
Dim intCodes() As Integer
```

```
Dim varCodeValues As Variant
```

```
Dim strName As String
```

```
On Error GoTo Done
```

```
With ThisDrawing.Utility
```

61

```
strName = .GetString(True, vbCr & "Layer name to exclude: ")
```

```
If "" = strName Then Exit Sub
```

```
" create a new selectionset
```

```
Set objSS = ThisDrawing.SelectionSets.Add("TestSelectionSetOperator")
```

```
" using 9 filters
```

```
ReDim intCodes(9): ReDim varCodeValues(9)
```

```
" set codes and values indented for clarity
```

```
intCodes(0) = 4: varCodeValues(0) = "<and"
```

```
intCodes(1) = 4: varCodeValues(1) = "<or"
```

```
intCodes(2) = 0: varCodeValues(2) = "line"
```

```
intCodes(3) = 0: varCodeValues(3) = "arc"
```

```
intCodes(4) = 0: varCodeValues(4) = "circle"
```

```

intCodes(5) = 4: varCodeValues(5) = "or>"
intCodes(6) = 4: varCodeValues(6) = "<not"
intCodes(7) = 8: varCodeValues(7) = strName
intCodes(8) = 4: varCodeValues(8) = "not>"
intCodes(9) = 4: varCodeValues(9) = "and>"
" filter the objects
objSS.Select acSelectionSetAll, , , intCodes, varCodeValues
" highlight the selected entities
objSS.Highlight True
" pause for the user
.Prompt vbCr & objSS.Count & " entities selected"
.GetString False, vbLf & "Enter to continue "
" dehighlight the entities
objSS.Highlight False
End With
Done:
" if the selection was created, delete it
If Not objSS Is Nothing Then
objSS.Delete
End If
End Sub

```

## **VI.**

### **Làm việc với Group**

Khi một group được tạo, theo mặc định khi ta chọn một thực thể trong group đó, tất cả các thực thể khác trong group đó sẽ được chọn. Nếu bạn muốn chọn từng thực thể trong group, hãy thay đổi biến hệ thống PICKSTYLE nhận giá trị 0 (mặc định nhận giá trị 1)

#### **1.**

#### **Tạo một Group Object**

Các group Object nằm trong Group Collection, ta dùng phương thức add để thêm một Group Object vào Group Collection, cú pháp như sau:

**Set GroupObject = GroupsCollection.Add(Name)**

**NAME**

**DATA TYPE**

## DESCRIPTION

Name

String

Tên của Group sẽ được tạo

2.

### Truy cập tới các Group Object

Truy cập đến group Collection

Dim objGroups As AcadGroups

Set objGroups = ThisDrawing.Groups

62

Để thiết lập tham chiếu đến group object, ta sử dụng phương thức Item của Groups collection:

Dim objGroup As AcadGroup

Set objGroup = objGroups.Item(Index)

Set objGroup = objGroups.Item(NameGroup)

Phương thức item là phương thức mặc định của Group collection, do vậy ta có thể duyệt qua các Group Object như sau:

Public Sub ListGroups()

Dim objGroup As AcadGroup

Dim strGroupList As String

For Each objGroup In ThisDrawing.Groups

strGroupList = strGroupList & vbCrLf & objGroup.Name

Next

MsgBox strGroupList, vbOKOnly, "List of Groups"

End Sub

3.

### Thêm bớt thực thể vào Group

Bạn có thể thêm hay bớt thực thể trong group bằng phương thức AppendItem và RemoveItem

**GroupObject.AppendItem(Entities)**



## **GroupObject.RemoveItem(Entities)**

**NAME**

**DATA TYPE**

**DESCRIPTION**

Entities

Array of AcadEntity objects

Mảng các thực thể sẽ bớt ra khỏi group

Chú ý: phương thức RemoveItem không xóa thực thể khỏi bản vẽ mà chỉ xóa liên kết thực thể với group.

**4.**

### **Xóa Group Object**

Xóa group Object khỏi group collection. Các thực thể trong group object sẽ không bị xóa, chúng sẽ được trả về bản vẽ.

## **GroupObject.Delete**

**VII.**

### **Sử dụng layer, color và linetype**

**1.**

#### **Sử dụng layer, color**

Trong mục này chúng ta sẽ nghiên cứu:

Truy cập đến Layers Collection và Layer Objects.

Kiểm tra sự tồn tại của một Layer.

Tạo một Layer và chuyển một layer thành layer hiện hành.

Thiết lập các thuộc tính của layer như: On /Off, Thawed /Frozen, Locked /Unlocked

Đổi tên, xóa một layer

Thiết lập hay lấy các thông số Color và Linetype của layer.

### **Làm việc với Layers**

AutoCAD cung cấp Layers collection chứa tất cả các đối tượng Layer có trong bản vẽ. Bạn có thể tạo ra các layer bằng các tên layer object vào Layers collection.

Bạn có thể truy suất vào Layers collection bằng cách như sau:

```
Dim objLayers As AcadLayers
```

```
Set objLayers = ThisDrawing.Layers
```

Để tham chiếu đến một Layer có sẵn trong bản vẽ. Bạn sử dụng phương thức Item của layer Collection như sau:

63

```
Dim objLayer As AcadLayer
```

```
Set objLayer = objLayers.Item(Index)
```

```
Set objLayer = objLayers.Item("Name of Layer")
```

Index là một số integer đại diện cho vị trí của layer trong Layers collection. "Name Layer là một chuỗi, là tên của Layer mà ta muốn truy cập đến nó. Nếu bạn sử dụng index thì nó phải nằm trong khoảng từ 0 đến ( Layers.Count 1).

Giống như các AutoCAD collections khác, Item là phương thức mặc định của Layer collection. Có nghĩa là tên phương thức này có thể không được viết vào nhưng cad vẫn hiểu đó là phương thức

```
Dim objLayer As AcadLayer
```

```
Set objLayer = objLayers(index)
```

```
Set objLayer = objLayers("Name of Layer")
```

### **Duyệt qua các Layers trong Layer Collection**

Sử dụng For ... Each loop để duyệt qua các layer trong Layer Collection:

```
Public Sub ListLayers()
```

```
Dim objLayer As AcadLayer
```

```
For Each objLayer In ThisDrawing.Layers
```

```
Debug.Print objLayer.Name
```

```
Next
```

```
End Sub
```

Ngoài ra ta có thể duyệt qua các Layer thông qua chỉ số Index. Sử dụng

thuộc tính objLayers.Count 1 như sau:

```
Public Sub ListLayersManually()  
Dim objLayers As AcadLayers  
Dim objLayer As AcadLayer  
Dim intI As Integer  
Set objLayers = ThisDrawing.Layers  
For intI = 0 To objLayers.Count - 1  
Set objLayer = objLayers(intI)  
Debug.Print objLayer.name  
Next  
End Sub
```

### **Kiểm tra sự tồn tại của một Layer**

```
Public Sub CheckForLayerByIteration()  
Dim objLayer As AcadLayer  
Dim strLayerName As String  
strLayername = InputBox("Enter a Layer name to search for: ")  
If "" = strLayername Then Exit Sub ' exit if no name entered  
For Each objLayer In ThisDrawing.Layers ' iterate layers  
If 0 = StrComp(objLayer.name, strLayername, vbTextCompare) Then  
MsgBox "Layer '" & strLayername & "' exists"  
Exit Sub ' exit after finding layer  
End If  
Next objLayer  
MsgBox "Layer '" & strLayername & "' does not exist"
```

64

End Sub

Ngoài ra bạn có thể kiểm tra sự tồn tại của một Layer bằng cách bẫy Layer

```
Public Sub CheckForLayerByException()
```

```

Dim strLayerName As String
Dim objLayer As AcadLayer
strLayerName = InputBox("Enter a Layer name to search for: ")
If "" = strLayerName Then Exit Sub ' exit if no name entered
On Error Resume Next ' handle exceptions inline
Set objLayer = ThisDrawing.Layers(strLayerName)
If objLayer Is Nothing Then ' check if obj has been set
MsgBox "Layer "" & strLayerName & "" does not exist"
Else
MsgBox "Layer "" & objLayer.Name & "" exists"
End If
End Sub

```

### **Creating a New Layer**

Để tạo mới một Layer, ta sử dụng phương thức Add trong Layer Collection như sau:

```
Set LayerObject = LayerCollection.Add(LayerName)
```

**NAME**

**DATA TYPE**

**DESCRIPTION**

LayerName

String

Nếu bạn Add một Layer có tên trùng với Layer đã có sẵn. AutoCAD sẽ sinh ra một lỗi.

```

Public Sub AddLayer()
Dim strLayerName As String
Dim objLayer As AcadLayer
strLayerName = InputBox("Name of Layer to add: ")
If "" = strLayerName Then Exit Sub ' exit if no name entered
On Error Resume Next ' handle exceptions inline
'check to see if layer already exists
Set objLayer = ThisDrawing.Layers(strLayerName)
If objLayer Is Nothing Then
Set objLayer = ThisDrawing.Layers.Add(strLayerName)

```

```

If objLayer Is Nothing Then ' check if obj has been set
MsgBox "Unable to Add " & strLayerName & ""
Else
MsgBox "Added Layer " & objLayer.Name & ""
End If
Else
MsgBox "Layer already existed"
End If
End Sub

```

**Chuyển Layer thành Layer hiện hành.**

65

Cú pháp như sau:

**DocumentObject.ActiveLayer = LayerObject**

Ví dụ: ThisDrawing.ActiveLayer = ThisDrawing.Layers("Walls")

Để lấy Layer hiện hành như sau:

**ThisDrawing.ActiveLayer.Name**

Ví dụ: If ThisDrawing.ActiveLayer.Name = "Walls" Then ...

**Turning a Layer On/Off**

**LayerObject.LayerOn = blnLayerOn**

Bạn cũng có thể kiểm tra layer đang bật bằng đoạn code sau:

If objLayer.LayerOn Then ... 'Thi hành nếu *Layer* là *On*.

**Thiết lập Layer: Frozen/Thawed**

Cú pháp như sau:

**objLayer.Freeze = True: Freeze**

**objLayer.Freeze = True: Thaw**

Kiểm tra tình trạng đóng băng của Layer:

**If objLayer.Freeze Then ...** 'Thi hành nếu layer đã đóng băng

**Thuộc tính Locking/Unlocking của Layer**

Bạn không thể chọn các thực thể đã bị khóa băng để chỉnh sửa. Tuy nhiên,

đôi tượng vẫn hiển thị nếu layer không bị đóng băng, và bạn vẫn có thể sử dụng các phương thức truy bắt điểm với chúng. Trong lúc layer bị khóa, bạn vẫn có thể thêm các thực thể đồ họa vào layer đó.

**objLayer.Lock = False: Unlock**

**objLayer.Lock = True: Lock**

Kiểm tra tình trạng của Layer:

**If objLayer.Lock Then ...** ' Thi hành nếu layer đang bị khóa.

**Thiết lập chế độ in cho Layer (Plottable or Not)**

Cú pháp như sau:

**objlayer.Plottable = False:** Không in khi bạn thực hiện lệnh in

**objlayer.Plottable = True:** Sẽ in khi bạn thực hiện lệnh in

Chú ý: Một số Layer tạo bởi ACIS hoặc ShapeManager solidmodeling engine như layer DEFPOINTS sẽ luôn luôn không được in ra.

**Đổi tên cho Layer (Plottable or Not)**

Cú pháp như sau:

**objLayer.Name = strLayerName**

**Deleting a Layer**

Cú pháp:

**LayerObject.Delete**

Phương thức Layer.Delete xóa một đối tượng Layer object ra khỏi Layers collection. Trong một số trường hợp sau, layer sẽ không được xóa:

66

Là layer hiện hành.

Là layer "0" (zero).

Là layer có chứa thực thể.

Là một Xrefdependent layer.

Nếu bạn cố tình xóa nó, Autocad sẽ sinh ra một lỗi.

### **Lấy Handle của một Layer**

AutoCAD gán cho mỗi object một giá trị handle hoặc ID duy nhất. Nó tồn tại và không thay đổi cùng với sự tồn tại Object đó. Bạn có thể truy cập đến handle của các Object thông qua phương thức Handle như sau:

```
Dim objLayer As AcadLayer
Dim strLayerHandle As String
Set objLayer = ThisDrawing.Layers("0")
strLayerHandle = objLayer.Handle
```

Handles được sử dụng rộng rãi khi bạn làm việc với extended entity data. *Extended entity data*, or *Xdata*, đó là những thông tin không phải là đồ họa được gán với Object bởi application program.

### **Layer Colors**

Mỗi layer có thuộc tính Color cung cấp màu cho tất cả các thực thể được vẽ trong layer nếu thuộc tính Color được thiết lập là ByLayer. Mặc định, màu của một layer mới sẽ là trắng hoặc đen tùy thuộc vào màu nền của AutoCAD. Ta có thể gán giá trị màu của Layer từ 0 đến 256. 9 trong số đó 9 được liệt kê trong AutoCAD VBA constant như sau:

```
objLayer.Color = acRed
```

Table 62: AutoCADDefined Color Constants

#### **CONSTANT**

#### **COLOR INDEX**

#### **COLOR**

acByBlock

0

#### **ByBlock**

acRed

1

Red

acYellow

2

Yellow

acGreen

3

Green

acCyan

4

Cyan

acBlue

5

Blue

acMagenta

6

Magenta

acWhite

7

White/Black (depending on the screen background color)

acByLayer

256

**ByLayer**

Lấy giá trị màu của một layer:

**intColor = objLayer.Color**

Tương tự bạn có thể thay đổi màu của từng thực thể với cú pháp như sau:  
**objLayer.Color = acColor**

**Layer Linetypes**

Mỗi Layer có một thuộc tính Linetype. Mặc định kiểu nét của các đối tượng được vẽ trong layer sẽ có kiểu nét là Linetype của layer. Trừ khi thực thể được người dùng thiết lập tới một kiểu nét khác.

67

Mặc định, kiểu đường của một layer mới là Continuous, tức là một solid line. Bạn có thể thay đổi chúng dựa trên thuộc tính **Layer.Linetype**

Ví dụ như sau:

```
Public Sub Layer0Linetype()
```

```
Dim objLayer As AcadLayer
```

```
Dim strLayerLinetype As String
```



```
Set objLayer = ThisDrawing.Layers("0")
objLayer.Linetype = "Continuous"
strLayerLinetype = objLayer.Linetype
End Sub
```

Lưu ý, khi ta sử dụng Linetype, bạn phải đảm bảo rằng Linetypes mà bạn sử dụng đã được Load vào trong bản vẽ.

### **Layer Lineweights**

Cú pháp của thuộc tính như sau: *objLayer.Lineweight*

2.

### **Sử dụng linetype**

Trong phần này chúng ta sẽ nghiên cứu những mục sau:

Truy cập Linetype collection và Linetype object.

Kiểm tra sự tồn tại của một linetype.

Tải một linetype vào trong bản vẽ và chuyển một Linetype thành dạng đường hiện hành.

Đổi tên và xóa một dạng đường.

Thiết lập và lấy các thông số về linetype's scale và description.

### **Truy cập Linetypes trong VBA**

```
Dim objLinetypes As AcadLineTypes
```

```
Set objLinetypes = ThisDrawing.Linetypes
```

Để tham chiếu đến một đối tượng LineTypeTo có sẵn, sử dụng phương thức Item:

```
Dim objLinetype As AcadLineType
```

```
Set objLinetype = objLinetypes.Item(Index)
```

```
Set objLinetype = objLinetypes.Item(NameLinetype)
```

NameLinetype kiểu string là tên dạng đường đã được load vào trong bản vẽ

Index kiểu interger, cũng giống như trong Layer Collection, Index nằm từ 0 đến (Linetypes.Count 1).

Giống như tất cả các collection khác trong AutoCAD, Phương thức Item là phương thức mặc định trong Linetypes collection.

### **Kiểm tra sự tồn tại của một Linetype**

```
Public Sub CheckForLinetypeByIteration()
```

```
Dim objLinetype As AcadLineType
```

```
Dim strLinetypeName As String
```

```
strLinetypeName = InputBox("Enter a Linetype name to search for: ")
```

```
If "" = strLinetypeName Then Exit Sub ' exit if no name entered
```

```
For Each objLinetype In ThisDrawing.Linetypes
```

```
If 0 = StrComp(objLinetype.Name, strLinetypeName, vbTextCompare)
```

Then

```
MsgBox "Linetype '" & strLinetypeName & "' exists"
```

```
Exit Sub ' exit after finding linetype
```

```
End If
```

```
Next objLinetype
```

68

```
MsgBox "Linetype '" & strLinetypeName & "' does not exist"
```

```
End Sub
```

Ngoài ra ta có thể bẫy lỗi để kiểm tra sự tồn tại của một LineType:

```
Public Sub CheckForLinetypeByException()
```

```
Dim strLinetypeName As String
```

```
Dim objLinetype As AcadLineType
```

```
strLinetypeName = InputBox("Enter a Linetype name to search for: ")
```

```
If "" = strLinetypeName Then Exit Sub ' exit if no name entered
```

```
On Error Resume Next ' handle exceptions inline
```

```
Set objLinetype = ThisDrawing.Linetypes(strLinetypeName)
```

```
If objLinetype Is Nothing Then ' check if obj has been set
```

```
MsgBox "Linetype '" & strLinetypeName & "' does not exist"
```

```
Else
```

```
MsgBox "Linetype '" & objLinetype.Name & "' exists"  
End If  
End Sub
```

### Tải một Linetype vào trong bản vẽ

Cú pháp Load lintype như sau:

```
Set LinetypeObject = LinetypesCollection.Load(LinetypeName,  
LinetypeFilename)
```

**NAME**

**DATA TYPE**

**DESCRIPTION**

LinetypeName

String

Tên của linetype

LinetypeFilename

String

Đường dẫn của file chứa LineType cần Load

Ví dụ: ThisDrawing.Linetypes.Load "Hidden", "acad.lin"

### Chuyển Linetype thành dạng đường hiện hành

Sử dụng phương thức ActiveLinetype như sau:

```
DocumentObject.ActiveLinetype = LinetypeObject
```

Ví dụ sau biến dạng đường "TRACKS" thành dạng đường hiện hành của bản vẽ hiện hành:

```
ThisDrawing.ActiveLinetype = ThisDrawing.Linetypes("TRACKS")
```

### Đổi tên Linetype

Sử dụng thuộc tính Linetype.Name property, bạn có thể thay đổi tên của một dạng đường.

### Deleting a Linetype

Phương thức Linetype.Delete cho phép bạn xóa một đối tượng Linetype từ Linetypes collection.

### LinetypeObject.Delete

Ta không thể xóa một dạng đường ra khỏi bản vẽ khi:

Nó là linetype hiện hành.

Nó là ByLayer, ByBlock, or Continuous linetype.

Nó là một Xrefdependent linetype.

### **Lấy Handle của Linetype**

69

```
Dim objLinetype As AcadLinetype
```

```
Dim strLinetypeHandle As String
```

```
Set objLinetype = ThisDrawing.Linetypes("Center")
```

```
strLinetypeHandle = objLinetype.Handle
```

### **Thay đổi Description của Linetype.**

AutoCAD Cho phép bạn read, add hoặc modify description của Linetype bằng cách sử dụng phương thức Description của Linetype object.

```
Dim strLineTypeDescription As String
```

```
objLineType.Description = "Linetype Description: ..."
```

```
strLineTypeDescription = objLineType.Description
```

The following example changes a Linetype description based on user input:

### **Scaling Linetypes**

Bạn có thể để sử dụng hai loại tỷ lệ phóng: global linetype scale (LTSCALE) và individual linetype scale (CELTSCALE).

#### **Global Scale**

```
Dim dblNewLTScale As Double
```

```
ThisDrawing.SetVariable "LTSCALE", 2#
```

```
dblNewLTScale = ThisDrawing.GetVariable("LTSCALE")
```

#### **Individual Scale**

```
Dim dblNewCELTScale As Double
```

```
ThisDrawing.SetVariable "CELTSCALE", 2#
```

```
dblNewCELTScale = ThisDrawing.GetVariable("CELTSCALE")
```

3.

### **Gán layer, color, linetype cho đối tượng**

Sử dụng các thuộc tính sau:

Object.Layer

Object.Color

Object.LineType

## **VIII.**

### **Làm việc với kích thước**

#### **1.**

#### **Làm việc với DimStyle**

##### **Thêm một DimStyle Object.**

DimStyle object thiết lập sự xuất hiện của một nhóm các kích thước và các leader. Các DimStyle objects nằm trong DimStyles collection, do vậy chúng ta có thể truy cập đến nó thông qua phương thức Item của DimStyles collection's.

Để tạo một DimStyle, chúng ta sử dụng phương thức Add như sau:

**Set DimStyleObject = DimStylesCollection.Add(DimStyleName)**

70

**NAME**

**DATA TYPE**

**DESCRIPTION**

DimStyleName

String

Tên của DimStyle

Ví dụ:

```
Dim objDimStyle As AcadDimStyle
```

```
Set objDimStyle = ThisDrawing.DimStyles.Add("NewDimStyle")
```

##### **Thiết lập một Dimension Style**

Để thiết lập một DimStyle trong VBA, bạn phải nắm vững tất cả các biến hệ thống mà thông qua nó bạn có thể điều khiển tất cả các TAB trong Dimension Style.

##### **Phương thức CopyFrom.**

Bạn có thể sử dụng phương thức CopyFrom để copy một DimStyle object, cú pháp như sau: **DimStyleObject.CopyFrom SourceObject**

Thông số SourceObject là đối tượng DimStyle object được copy. Kiểu Copy phụ thuộc vào nguồn Copy mà bạn sử dụng được liệt kê ra trong bảng dưới đây:

## **OBJECT**

### **STYLES COPIED**

Dimension, Tolerance, Leader

VBA sẽ copy tất cả các dữ liệu về Dimension style của kích thước, dung sai, dấu dẫn, kể cả các dữ liệu ghi đè (override) có trong các dimension này vào **DimStyleObject**

Document

Copy Dimension style và override của Dimension style hiện hành của bản vẽ được chỉ định trong *Document*

DimStyle

Copy Dimension style của bản vẽ hiện hành được chỉ định trong *DimStyle*

Ví dụ sau tạo một DimStyle mới, lấy tên là NewDimStyle. DimStyle này thừa hưởng tất cả các thuộc tính của DimStyle hiện hành, trừ màu của dimension line, extension line, và dimension text được đặt theo thứ tự là red, blue, and white.

```
Public Sub NewDimStyle
```

```
Dim objDimStyle As AcadDimStyle
```

```
Set objDimStyle = ThisDrawing.DimStyles.Add("NewDimStyle")
```

```
SetVariable "DIMCLRD", acRed
```

```
SetVariable "DIMCLRE", acBlue
```

```
SetVariable "DIMCLRT", acWhite
```

```
SetVariable "DIMLWD ", acLnWtByLwDefault
```

```
objDimStyle.CopyFrom ThisDrawing
```

```
End Sub
```

Giải thích như sau: khi thay đổi các thông số hệ thống của Dimension style, Cad sẽ tạo ra một kiểu kích thước ghi đè (override) dựa trên kiểu kích thước hiện hành. Đoạn mã objDimStyle.CopyFrom ThisDrawing sẽ copy toàn bộ thiết lập của Dimstyle hiện hành và override của nó vào *NewDimStyle*

## **Sử dụng Dimension Style**

### **Gán một Dimstyle cho một Dimension (một biến kích thước)**

```
Object.StyleName = DimStyleName
```

**NAME**

**DATA TYPE**

**DESCRIPTION**

Object

Dimension, Leader, or Tolerance object

Là object mà bạn muốn áp đặt Dimstyle cho nó.

71

**NAME DATA TYPE DESCRIPTION**

DimStyleName

String

Tên của Dimstyle

**Chuyển một DimStyle thán Dimstyle hiện hành**

Để chuyển một DimStyle thành DimStyle hiện hành, bạn sử dụng phương thức ActiveDimStyle như sau:

**Set DocumentObject.ActiveDimStyle = DimStyleObject**

**NAME**

**DATA TYPE**

**DESCRIPTION**

DimStyleObject

DimStyle object

Là biến kiểu Dimstyle mà bạn muốn chuyển nó thành Dimstyle hiện hành.

Ví dụ sau hiển thị kiểu kích thước hiện hành của bản vẽ hiện hành.

```
Msgbox "Kiểu kích thước hiện hành: " &  
ThisDrawing.ActiveDimStyle.Name
```

```
Dim objDimStyle As AcadDimStyle
```

Duyệt qua các Dimstyle có trong bản vẽ hiện hành.

```
For Each objDimStyle In ThisDrawing.DimStyles
```

```
Msgbox objDimStyle.Name
```

```
Next
```

## 2.

### Tạo các đường đo kích thước

#### Tạo các Dimensions

Cũng như các thực thể đồ họa khác, để tạo một đường đo kích thước, ta sử dụng phương thức AddDimXXX. Phương thức này có thể áp dụng trong ModelSpace, PaperSpace và Block objects.

Trong mục này, chúng ta sẽ nghiên cứu các loại kích thước sau:

Dim3PointAngular Object

DimAligned Object

DimAngular Object

DimDiametric Object

DimOrdinate Object

DimRadial Object

DimRotated Object

**Dim3PointAngular Object**

**Set Dim3PointAngularObject = Object.AddDim3PointAngular(VertexPoint, Point1, Point2, TextPosition)**

**NAME**

**DATA TYPE**

**DESCRIPTION**

VertexPoint

Variant

VertexPoint là mảng 3 phần tử kiểu Double mô tả tọa độ một đỉnh của góc cần đo trong hệ tọa độ WCS.

Point1

Variant

Point1 là mảng 3 phần tử kiểu Double mô tả tọa độ một trong 2 điểm cuối



trong hệ tọa độ WCS.

Point2

Variant

Point2 là mảng 3 phần tử kiểu Double mô tả tọa độ một trong 2 điểm cuối trong hệ tọa độ WCS.

TextPosition

Variant

TextPosition là mảng 3 phần tử kiểu Double mô tả tọa độ vị trí mà Text (góc đo được) sẽ hiển thị hệ tọa độ WCS.

72

### **DimAligned Object**

**Set DimAlignedObject = Object.AddDimAligned(Point1, Point2, TextPosition)**

NAME

DATA TYPE

DESCRIPTION

Point1

Variant

Point1 là mảng tọa độ 3 phần tử kiểu Double mô tả tọa độ một trong 2 điểm cuối trong hệ tọa độ WCS.

Point2

Variant

Point2 là mảng 3 phần tử kiểu Double mô tả tọa độ một trong 2 điểm cuối trong hệ tọa độ WCS.

TextPosition

Variant

TextPosition là mảng 3 phần tử kiểu Double mô tả tọa độ vị trí mà Text sẽ hiển thị hệ tọa độ WCS.

### **DimAngular Object**

**Set DimAngularObject = Object.AddDimAngular(Vertex, Point1, Point2, TextPosition)**

**NAME****DATA TYPE****DESCRIPTION**

Vertex

Variant

VertexPoint là mảng 3 phần tử kiểu Double mô tả tọa độ một đỉnh của góc cần đo trong hệ tọa độ WCS.

Point1

Variant

Point1 là mảng 3 phần tử kiểu Double mô tả tọa độ một trong 2 điểm cuối trong hệ tọa độ WCS.

Point2

Variant

Point2 là mảng 3 phần tử kiểu Double mô tả tọa độ một trong 2 điểm cuối trong hệ tọa độ WCS.

TextPosition

Variant

TextPosition là mảng 3 phần tử kiểu Double mô tả tọa

73

**NAME DATA****TYPE****DESCRIPTION**

độ vị trí mà Text (góc đo được) sẽ hiển thị hệ tọa độ WCS.

**DimDiametric Object**

**DimDiametricObject = Object.AddDimDiametric (ChordPoint1, ChordPoint2, LeaderLength)**

**NAME****DATA TYPE****DESCRIPTION**

ChordPoint1

Variant

ChordPoint1 là mảng 3 phần tử kiểu Double mô tả tọa độ một trong 2 điểm cuối của dây cung trong hệ tọa độ WCS.

ChordPoint2

Variant

ChordPoint2 là mảng 3 phần tử kiểu Double mô tả tọa độ một trong 2 điểm cuối của dây cung trong hệ tọa độ WCS.

LeaderLength

Double

Khoảng cách từ điểm cuối của dây cung ChordPoint1 tới vị trí của giá trị Text đo được.

**DimOrdinate Object**

**Set DimOrdinateObject = Object.AddDimOrdinate(DefinitionPoint, LeaderEndPoint, UseXAxis)**

**NAME**

**DATA TYPE**

**DESCRIPTION**

74

**NAME DATA**

**TYPE**

**DESCRIPTION**

DefinitionPoint

Variant

DefinitionPoint là mảng 3 phần tử kiểu Double mô tả tọa độ trong hệ tọa độ WCS.

LeaderEndPoint

Variant

DefinitionPoint là mảng 3 phần tử kiểu Double mô tả tọa độ vị trí của Text trong hệ tọa độ WCS.

UseXAxis

Boolean

Xác định DefinitionPoint được đo theo phương trục X hay trục Y. Nếu giá trị này là True thì, trục X được sử dụng như là một tham chiếu.

### **DimRadial Object**

**Set DimRadialObject = Object.AddDimRadial (CenterPoint, ChordPoint, LeaderLength)**

**NAME**

**DATA TYPE**

**DESCRIPTION**

CenterPoint

Variant

CenterPoint là mảng 3 phần tử kiểu Double mô tả tọa độ tâm của đường tròn hay cung tròn cần đo trong hệ tọa độ WCS.

ChordPoint

Variant

ChordPoint là mảng 3 phần tử kiểu Double mô tả tọa độ điểm nằm trên dây cung trong hệ tọa độ WCS.

LeaderLength

Double

Khoảng cách từ Text đến điểm ChordPoint

### **The DimRotated Object**

**Set DimRotatedObject = Object.AddDimRotated(Point1, Point2, DimLocationPoint, RotationAngle)**

75

**NAME**

**DATA TYPE**

**DESCRIPTION**

Point1

Variant

Point1 là mảng 3 phần tử kiểu Double mô tả tọa độ một trong 2 điểm cuối

của đoạn thẳng trong hệ tọa độ WCS.

Point2

Variant

Point2 là mảng 3 phần tử kiểu Double mô tả tọa độ một trong 2 điểm cuối của đoạn thẳng trong hệ tọa độ WCS.

DimLocationPoint

Variant

DimLocationPoint là mảng 3 phần tử kiểu Double mô tả tọa độ điểm mà tại đó CAD sẽ vẽ đường thẳng chứa mũi tên và Text trong hệ tọa độ WCS.

RotationAngle

Double

Góc hợp bởi đường ghi kích thước và trục X, đơn vị Radian.

**3.**

**Tạo các leader.**

76

## **CHƯƠNG 4: TÙY BIẾN MENUS VÀ TOOLBARS**

### **I.**

#### **Cơ bản về menu group và toolbar**

Trong mục này, chúng ta sẽ nghiên cứu những mục sau:

Load, save, và unload một menu group

Gán các phím tắt (accelerator key)

Thao tác với các menu bar

Tạo và chỉnh sửa menu

Tạo và chỉnh sửa toolbar

Toolbar động và chức năng Docking.

MenuGroups collection chứa các MenuGroup object, nó chứa tất cả các đối tượng thuộc ToolBars và PopupMenus collections. MenuBar collection sẽ chứa tất cả PopupMenu object đang xuất hiện trên AutoCAD menu bar.

1.

### **Menugroup collection**

#### **MenuGroups Collection**

Tất cả các menu được load trong session hiện tại của AutoCAD được lưu cất trong MenuGroup collection. Các menu này có thể được hiển thị hoặc không được hiển thị trên menu bar của Autocad. Người dùng có thể điều khiển sự hiển thị của các menu thông qua lệnh MenuLoad trong Autocad. Mỗi một menu MenuGroup object cung cấp tất cả các toolbar and popup menu có trong Group đó.

#### **Loading Menu Groups**

Phương thức Load của MenuGroup collection dùng để tải một menu group được chứa trong các file ( .mnc, .mns, or .mnu ) vào Autocad. Cú pháp của lệnh này như sau:

**Set MenuGroupObject = MenuGroupsCollection.Load  
(MenuFileName [,BaseMenu])**

**NAME**

**DATA TYPE**

**DESCRIPTION**

MenuFileName

String

Đường dẫn và tên File sẽ được tải vào AutoCad.

BaseMenu

Boolean

Thông số này xác định menu group được tải là menu cơ bản hay menu từng phần. Giá trị True là base menu. Ngược lại sẽ là partial menu. Mặc định là False.

Sử dụng tham số BaseMenu với giá trị True sẽ tương đương với việc thực hiện lệnh MENU trong AutoCAD (hoặc thi hành lệnh MENULOAD và check vào lựa chọn Replace All). Chỉ MenuGroup mới mới được Load và nó sẽ thay thế tất cả các menu cũ trong AutoCad.

Như một lựa chọn, sử dụng phương thức Load Với tham số BaseMenu là False, tương đương với bạn thi hành lệnh MENULOAD trong AutoCAD mà

không check vào lựa chọn. Replace All. Menu group được tải thêm vào AtutoCad.

## Kiểu của Menu Groups

77

Để lấy thông số kiểu của menu groups đã được tải vào trong AutoCAD, bạn có thể sử dụng thuộc tính *Type* cho mỗi MenuGroup object. Cú pháp như sau:

**IngMenuGroupType = MenuGroupObject.Type**

**CONSTANT**

**VALUE**

**DESCRIPTION**

AcBaseMenuGroup

0

Menu group là base menu group.

AcPartialMenuGroup

1

Menu group là partial menu group.

Ví dụ sau sẽ liệt kê tất cả các menu group đã được tải vào trong AutoCad và kiểu của chúng:

```
Public Sub ListMenuGroups()  
Dim objMenuGroup As AcadMenuGroup  
Dim strMenuGroupNames As String  
strMenuGroupNames = "The following menu groups are currently loaded,  
"  
  
For Each objMenuGroup In Application.MenuGroups  
If objMenuGroup.Type = acBaseMenuGroup Then  
strMenuGroupNames = strMenuGroupNames & vbCrLf & _  
objMenuGroup.Name & ": Base menu"  
Else  
strMenuGroupNames = strMenuGroupNames & vbCrLf & _
```

```
objMenuGroup.Name & ": Partial menu"  
End If  
Next  
MsgBox strMenuGroupNames  
End Sub  
Lưu ý:
```

MenuGroups collection không có phương thức Add. Tuy nhiên bạn có thể tạo một bản copy file .mns vào một file mới, sau đó tải và sửa chữa nó theo ý của bạn. Bạn cũng có thể tạo một file .mnu hoặc .mns mới và tải chúng từ VBA.

Bạn không thể chỉnh sửa được các menu hình ảnh, menu màn hình và các bảng số hóa. Tuy nhiên, bạn vẫn có thể Load và UnLoad chúng bằng VBA.

2.

## **Menugroup Object**

### **Thành phần của MenuGroup Object**

Bạn Load một MenuGroup vào trong AutoCAD, tương đương với một MenuGroup object được thêm vào MenuGroup collection. Mỗi MenuGroup object chứa hai collection là PopupMenus and Toolbars. Cú pháp để truy cập đến 2 collection trên như sau:

```
Set PopupMenusCollection = MenuGroupObject.Menus
```

```
Set ToolbarsCollection = MenuGroupObject.Toolbars
```

### **Saving Menu Groups**

Bạn Có hai phương thức để ghi lại: Save và SaveAs.

```
MenuGroupObject.Save MenuFileType
```

```
MenuGroupObject.SaveAs FileName, MenuFileType
```

**NAME**

**DATA TYPE**

**DESCRIPTION**

MenuFileType

Long

Xác định kiểu file sẽ được ghi: File nguồn hoặc file đã được biên dịch (chi tiết ở bảng dưới).



78

## **NAME DATA**

### **TYPE**

### **DESCRIPTION**

FileName

String

Đường dẫn đầy đủ và tên file sẽ ghi lại.

Các hằng số AcMenuFileType

### **CONSTANT**

#### **VALUE**

#### **DESCRIPTION**

acMenuFileCompiled

0

compiled menu file (.mnc extension)

acMenuFileSource

1

source menu file (.mns extension)

### **Unloading Menu Groups**

Tương đương với lệnh MENULOAD hoặc MENUUNLOAD trong AutoCad:

#### **MenuGroupObject.Unload**

## **II.**

### **Thay đổi menu bar**

Bạn có thể: Thêm, bớt, sắp xếp lại các menu trên menu bar.

#### **1.**

#### **Chèn thêm menu vào menubar**

#### **Phương thức InsertInMenuBar của PopupMenu object.**

#### **PopupMenuObject.InsertInMenuBar(Index)**

### **NAME**

### **DATA TYPE**

### **DESCRIPTION**

Index

Variant

Vị trí mà popup menu sẽ được thêm vào MenuBar. Index là kiểu Integer từ 0 đến N. N là số lượng object trên menu bar, hoặc kiểu string là tên của một menu đã tồn tại (bao gồm cả ký tự & của accelerator key), popup menu sẽ được thêm vào trước menu có tên như trong index. Nếu menu trong index không tồn tại thì một menu mới sẽ được thêm vào tại vị trí cuối cùng của menu bar.

**InsertMenuInMenuBar method of a PopupMenus collection.**

**PopupMenuCollection.InsertMenuInMenuBar MenuName, Index**

**NAME**

**DATA TYPE**

**DESCRIPTION**

MenuName

String

Tên của menu sẽ được thêm vào menu bar.

Index

Variant

Giống như trên.

**2.**

**Xoá bỏ menu trên menubar**

Giống như thêm menu vào menu bar, bạn cũng có 2 cách để xóa menu khỏi chúng:

**Phương thức RemoveFromMenuBar**

**PopupMenuObject.RemoveFromMenuBar**

Sub RemoveMenus()

Dim objMenu As AcadPopupMenu

For Each objMenu In ThisDrawing.Application.MenuBar

If MsgBox("Remove " & objMenu.Name & "?", vbYesNo) = vbYes Then

objMenu.RemoveFromMenuBar



Label

String

Label của menu item. Label có thể chứa ngôn ngữ DIESEL.

Macro

String

Macro sẽ thực hiện khi menu được click.

**3.**

### **Chèn một khoảng trống vào Pulldown menu**

**Set    PopupMenuItemObject    =**  
**PopupMenuObject.AddSeparator(Index)**

**NAME**

**DATA TYPE**

**DESCRIPTION**

Index

Variant

Giống như trong Phương thức InsertMenuInMenuBar. Nếu menu item được xác định bởi tham số index không tồn tại, thì khoảng trống sẽ được đặt ở cuối của menu.

Chú ý: Bạn không thể thêm khoảng trống vào đầu của Popup menu.

80

**4.**

### **Thêm một Menu Item vào Shortcut Menu**

Giá trị của thuộc tính ShortcutMenu của PopupMenu object nhận giá trị True nếu menu là shortcut menu. Từ đó, bạn có thể thêm một item vào trong shortcut menu như đối với dropdown menu.

**5.**

### **Tạo Submenu cho menu item**

**Set    PopupMenuObject    =    PopupMenuObject.AddSubMenu(Index, Label)**

**NAME**

## **DATA TYPE**

## **DESCRIPTION**

Index

Variant

Giống như trong Phương thức InsertMenuInMenuBar

Label

String

Label của menu item. Label có thể chứa ngôn ngữ DIESEL.

**6.**

### **Xóa bớt menu item**

Sử dụng phương thức Delete như sau:

**PopupMenuObject.Delete**

**IV.**

### **Tạo và chỉnh sửa Toolbars**

**1.**

#### **Tạo mới toolbar**

**Set ToolbarObject = ToolbarsCollection.Add(ToolbarName)**

**NAME**

**DATA TYPE**

**DESCRIPTION**

ToolbarName

String

Tên của Toolbar object sẽ được tạo

**2.**

#### **Thêm một nút chọn vào toolbar**

Một Toolbar button được đại diện bởi ToolbarItem object. Bạn có thể sử dụng phương thức AddToolbarButton method để thêm một Toolbar item vào một vị trí xác định trên toolbar.

**Set ToolbarItemObject = ToolbarObject.AddToolbarButton(Index, ButtonName, HelpString, Macro[, FlyoutButton])**

**NAME**

**DATA TYPE**

**DESCRIPTION**

Index

## Variant

Vị trí mà popup menu sẽ được thêm vào MenuBar. Index là kiểu Integer từ 0 đến N. N là số lượng object trên menu bar, hoặc kiểu string là tên của một Toolbar button đã tồn tại, button mới sẽ được thêm vào trước button có tên như trong index. Nếu menu trong index không tồn tại thì một button mới sẽ được thêm vào tại vị trí cuối cùng của toolbar.

ButtonName

String

Tên của toolbar button sẽ được tạo. (chỉ chấp nhận các ký tự: alphanumeric, dashes (), và underscores (\_)).

HelpString

String

Là chuỗi chú thích

Macro

String

Macro sẽ thi hành khi ta chọn nút lệnh.

FlyoutButton

Boolean

Nếu thiết lập là True thì sẽ tạo một flyout button. Mặc định là False.

Chú ý: Toolbar button cũng giống như menu item. Khi bạn muốn thay đổi chỉ số Index của chúng. Bạn không được phép thay đổi luôn giá trị Index. Mà đầu tiên, bạn phải xóa toolbar button. Sau đó bạn thêm nó vào Toolbar tại vị trí mới, vị trí mà bạn muốn thay đổi.

### 3.

#### **Định nghĩa hình ảnh cho các nút Toolbar Button.**

81

Để thiết lập hay tải về bitmap của icons gắn với nút lệnh của toolbar, bạn có thể sử dụng phương thức SetBitmaps và GetBitmaps. cả 2 đều có cú pháp tương tự nhau như sau:

**ToolbarItemObject.SetBitmaps SmallIconName, LargeIconName**

**ToolbarItemobject.GetBitmaps SmallIconName, LargeIconName**

**NAME**

**DATA TYPE**

**DESCRIPTION**

SmallIconName

String

Path và tên file của small bitmap (16×15 pixels)

LargeIconName

String

Path và tên file của large bitmap(24×22 pixels)

**4.**

**Thêm một khoảng trống vào toolbar**

**Set ToolbarItemObject = ToolbarObject.AddSeparator(Index)**

**NAME**

**DATA TYPE**

**DESCRIPTION**

Index

Variant

Vị trí của khoảng trống trên Toolbar

Chú ý: Bạn không thể thêm khoảng trống vào đầu của ToolBar

**5.**

**Floating và Docking Toolbars**

**Floating Toolbars**

**ToolbarObject.Float Top, Left, NumberOfRows**

**NAME**

**DATA TYPE**

**DESCRIPTION**

Top

Long

Vị trí biên trên của Toolbar (pixel) tính từ màn hình phía trên.

Left

Long

Vị trí biên dưới tính từ vị trí biên trên

NumberOfRows

Long

Số dòng mà toolbar buttons có thể phân bố trên đó. Nếu NumberOfRows nhiều hơn số lượng buttons có trong toolbar, tham số này sẽ bị bỏ qua.

Ví dụ:

```
Public Sub FloatDrawToolbar()
```

```
Dim objToolBarDraw As AcadToolbar
```

```
Set objToolBarDraw =
```

```
ThisDrawing.Application.MenuGroups.Item("ACAD").Toolbars.Item("Draw")
```

```
objToolBarDraw.Float 0, 0, 3
```

```
End Sub
```

### **Docking Toolbars**

82

#### **ToolbarObject.Dock DockStatus**

**NAME**

**DATA TYPE**

**DESCRIPTION**

DockStatus

Long

Giá trị của nó xem trong bảng dưới đây.

Hằng số: AcToolbarDockStatus

**CONSTANT**

**VALUE**

**DESCRIPTION**

acToolbarDockTop

0

Toolbar được bám vào phía trên của bản vẽ.

acToolbarDockBottom

1



Toolbar được bấm vào phía dưới của bản vẽ.

```
acToolbarDockLeft
```

2

Toolbar được bấm vào biên trái dưới của bản vẽ.

```
acToolbarDockRight
```

3

Toolbar được bấm vào biên phải của bản vẽ.

Ví dụ minh họa acToolbarDockBottom

**6.**

### **Tạo các flyout toolbar**

Sử dụng Set ToolbarItemObject = ToolbarObject.AddToolbarButton(Index, ButtonName, HelpString, Macro[, FlyoutButton]) với tham số FlyoutButton là True. Ví dụ sau tạo hai toolbar. Cái đầu tiên chứa flyout button. Toolbar thứ 2 được gắn vào flyout button của toolbar đầu tiên.

```
Sub Ch6_AddFlyoutButton()  
Dim currMenuGroup As AcadMenuGroup  
' Create the first toolbar  
Set currMenuGroup = ThisDrawing.Application.MenuGroups.Item(0)  
Dim FirstToolbar As AcadToolbar  
Set FirstToolbar = currMenuGroup.Toolbars.Add("FirstToolbar")  
' Add a flyout button to the first menu on the menu bar  
Dim FlyoutButton As AcadToolbarItem  
' Create the second toolbar. This will be attached to  
' the first toolbar via the flyout button.  
Set FlyoutButton = FirstToolbar.AddToolbarButton("", "Flyout",  
"Demonstrates a flyout button", "OPEN", True)  
Dim SecondToolbar As AcadToolbar  
' Add a button to the next toolbar  
Set SecondToolbar = currMenuGroup.Toolbars.Add("SecondToolbar")
```

83

```
Dim newButton As AcadToolbarItem
Dim openMacro As String
' Assign the macro the VB equivalent of "ESC ESC _open "
openMacro = Chr(vbKeyEscape) + Chr(vbKeyEscape) + "_open "
Set newButton = SecondToolbar.AddToolbarButton ("", "NewButton",
"Open a file.", openMacro)
' Attach the second toolbar to the flyout button on the first toolbar
FlyoutButton.AttachToolbarToFlyout
currMenuGroup.Name,SecondToolbar.Name
' Display the first toolbar, hide the second toolbar
FirstToolbar.Visible = True
SecondToolbar.Visible = False
End Sub
```

7.

### **Xóa Toolbar và Toolbar Button**

Bạn có thể xóa cả toolbars và toolbar buttons bằng phương thức Delete như sau:

**ToolbarObject.Delete**

**ToolbarItemObject.Delete**

V.

### **Tạo các macro**

1.

### **Các quy định về macro**

Bảng ký tự dùng trong Macro và mã ASCII tương đương.

#### **Character**

**ASCII**

**Mô tả**

;

chr(59)

ENTER

^M

chr(13)

ENTER

^|

chr(94) + chr(124)

TAB

SPACEBAR

chr(32)

Khoảng trắng trong Macro sẽ tương đương với bạn ấn phím SPACEBAR

\

chr(92)

Dừng lại để nhập số liệu

-

chr(95)

Chuyển lệnh đứng sau nó sang Tiếng Anh

+

chr(43)

Tiếp tục macro ở dòng lệnh tiếp theo(đặt ở cuối dòng lệnh trước)

=\*

chr(61) + chr(42)

Hiện thị hiện hành toplevel image, pulldown, và shortcut menu

\*^C^C

chr(42) + chr(3) + chr(3)

Dùng để lặp lại các lệnh đứng sau nó cho đến khi bạn ấn Esc

\$

chr(36)

Để tải một đoạn menu section hoặc thêm một đoạn biểu thức DIESEL

^B

chr(2)

Bật hoặc tắt Snap (CTRL+B)

^C

chr(3)

Cancels command (CTRL+C)

ESC

chr(3)

Cancels command (ESC)

**^D**

**chr(4)**

**Bật tắt Coords (CTRL+D)**

**^E**

**chr(5)**

**Thiết lập mặt phẳng cùng kích thước (CTRL+E)**

**^G**

**chr(7)**

**Bật tắt Grid (CTRL+G)**

**84**

**^H**

**chr(8)**

**Trả về phím backspace**

**^O**

**chr(15)**

**Bật tắt chế độ Ortho (CTRL+O)**

**^Q**

**chr(17)**

**Không hiển thị tất cả các thông báo, các dòng tính trạng làm và dữ liệu nhập vào (CTRL+Q)**

**^T**

**chr(20)**

**Bật tắt menu số hóa Tablet (CTRL+T)**

**^V**

**chr(22)**

**Changes current viewport (CTRL+V)**

**^Z**

**chr(26)**

**Null character đặt ở cuối Macro để xóa các ký tự phát sinh ở cuối của**

menu Item.

Chú ý về ký tự kết thúc macro như sau. Nếu bạn kết thúc macro mà không có các ký tự đặc biệt để kết thúc macro như “;” thì AutoCAD sẽ tự động thêm vào cuối macro một phím cách. Tuy nhiên, đối với các lệnh Text thì phím cách này sẽ trả về một ký tự trắng của text nhập vào. Do vậy ta cần nắm chắc quy tắc sau:

Khi có một dấu chấm phẩy trong macro, AutoCAD sẽ thay thế nó bằng phím ENTER.

Nếu dòng macro kết thúc bằng các ký tự (\), (+), hoặc (;), AutoCAD sẽ không thêm một khoảng trắng sau nó.

**2.**

**Các ví dụ về macro**

85

## **CHƯƠNG 5: PHÁT TRIỂN ỨNG DỤNG VỚI VBA**

**I.**

### **AutoCAD Events**

Event xuất hiện như một kết quả của một sự kiện xảy ra khi chương trình đang chạy. Autocad cung cấp 3 cấp Event: application, document, and object. Event handlers (bộ quản lý sự kiện) là một Sub procedures, thủ tục này sẽ được chạy một cách tự động khi các sự kiện gắn với nó xảy ra.

ApplicationLevel Events: Xuất hiện khi có sự thay đổi trong môi trường ứng dụng AutoCAD. Nó bao gồm các sự kiện mở bản vẽ, đóng bản vẽ, thực thi dòng lệnh, thay đổi các biến hệ thống và thay đổi của sổ ứng dụng AutoCAD.

DocumentLevel Events: Xuất hiện khi có sự thay đổi trong bản vẽ như thêm, sửa một đối tượng, tái sinh lại bản vẽ.

ObjectLevel Events: xuất hiện khi có sự thay đổi ở cấp thực thể trong bản vẽ, như sự kiện thay đổi (Modified).

## 1.

### **Applicationlevel events**

Applicationlevel events không mặc định có trong VBA khi bạn load một Project. Để sử dụng các Event này, bạn phải làm qua các bước sau:

Đầu tiên, bạn insert một class module, và khai báo một declare an object với kiểu AcadApplication với KeyWord Withevents. Ví dụ:

Bạn tạo một Class module có tên là EventClassModule

```
Public WithEvents objApp As AcadApplication
```

Sau khi bạn khai báo Object With event, và bạn có thể viết event procedures cho object mới trong class module. (khi bạn chọn object này trong Object box, tất cả các events hợp lệ sẽ được hiển thị trong Procedure dropdown list box như hình vẽ trên)

Sau đó bạn connect tới Application object

#### 1.

Trong cửa sổ chứa main module, bạn thêm các dòng khai báo sau:

```
Dim X As New EventClassModule
```

#### 2.

Cũng trong cửa sổ này, bạn thêm thủ tục sau:

```
Sub InitializeEvents()
```

```
Set X.ObjApp = ThisDrawing.Application
```

```
End Sub
```

#### 3.

Trong main module, bạn gọi thủ tục InitializeEvents:

```
Call InitializeEvents
```

Ví dụ sau:

```
Public WithEvents ACADApp As AcadApplication
```

```
Sub Example_AcadApplication_Events()
```

86

```
' This example initializes the public variable (ACADApp)
' which will be used to intercept AcadApplication Events
'
' Run this procedure FIRST!
' We could get the application from the ThisDocument
' object, but that would require having a drawing open,
' so we grab it from the system.
Set ACADApp = GetObject( "AutoCAD.Application.16")
End Sub
Private Sub ACADApp_BeginFileDrop _
(ByVal FileName As String, Cancel As Boolean)
' This example intercepts an Application BeginFileDrop event.
'
' This event is triggered when a drawing file is dragged
' into AutoCAD.
'
' To trigger this example event:
' 1) Make sure to run the example that initializes
' the public variable (named ACADApp) linked to this event.
'
' 2) Drag an AutoCAD drawing file into the AutoCAD
' application from either the Windows Desktop
' or Windows Explorer
' Use the "Cancel" variable to stop the loading of the
' dragged file, and the "FileName" variable to notify
' the user which file is about to be dragged in.
If MsgBox("AutoCAD is about to load " & FileName & vbCrLf _
& "Do you want to continue loading this file?", _
vbYesNoCancel + vbQuestion) <> vbYes Then
Cancel = True
End If
End Sub
```

Danh sách các Event trong application level:

AppActivate

AppDeactivate

ARXLoaded

ARXUnloaded

BeginCommand

BeginFileDrop

BeginLisp

BeginModal

BeginOpen

BeginPlot

BeginQuit

BeginSave

EndCommand

EndLisp

EndModal

EndOpen

EndPlot

87

EndSave

LispCancelled

NewDrawing

SysVarChanged

WindowChanged

WindowMovedOrResized

**2.**

### **Documentlevel events**

Không giống như applicationlevel events, documentlevel events mặc định có khi bạn tải Project vào bản vẽ. Nếu bạn chọn AcadDocument Object trong



Object List box của Thisdrawing module của AutoCAD project, the documentlevel events được liệt kê trong danh sách bên phải như hình vẽ dưới đây:

Các Event được sử dụng trong document level như sau:

Activate

BeginClose

BeginCommand

BeginDoubleClick

BeginLisp

BeginPlot

BeginRightClick

BeginSave

BeginShortcutMenuCommand

BeginShortcutMenuDefault

BeginShortcutMenuEdit

BeginShortcutMenuGrip

BeginShortcutMenuOsnap

Deactivate

EndCommand

EndLisp

EndPlot

EndSave

EndShortcutMenu

LayoutSwitched

LispCancelled

ObjectAdded

88

ObjectErased

ObjectModified

SelectionChanged

WindowChanged

WindowMovedOrResized

### 3.

#### **ObjectLevel Events**

Để sử dụng objectlevel events, đầu tiên bạn phải tạo mới một class module và khai báo biến để tham chiếu đến đối tượng mà bạn muốn bắt sự kiện Modified. Class module mới chứa khai báo đối tượng bằng cách sử dụng VBA keyword WithEvents, ví dụ:

```
Public WithEvents objLine As AcadLine
```

Đặt tên Class module đó là EventClassModule, sau đó viết Code cho Event:

#### 1.

Trong cửa sổ chứa main module, bạn thêm các dòng khai báo sau:

```
Dim X As New EventClassModule
```

#### 2.

Cũng trong cửa sổ này, bạn thêm thủ tục sau:

```
Public Sub InitializeEvent()
```

```
Dim dblStart(2) As Double
```

```
Dim dblEnd(2) As Double
```

```
dblEnd(0) = 1: dblEnd(1) = 1: dblEnd(2) = 0
```

```
Set X.objLine = ThisDrawing.ModelSpace.AddLine(dblStart, dblEnd)
```

```
End Sub
```

#### 3.

Trong main module, bạn gọi thủ tục InitsializeEvents:

```
Call InitializeEvents
```

Ví dụ sau đây tạo một polyline với Event

```
Public WithEvents PLine As AcadLWPolyline
```

```
Sub CreatePLineWithEvents()
```

```
' This example creates a light weight polyline
```

```
Dim points(0 To 9) As Double
```

```
points(0) = 1: points(1) = 1
```

```
points(2) = 1: points(3) = 2
```

```
points(4) = 2: points(5) = 2
```

```

points(6) = 3: points(7) = 3
points(8) = 3: points(9) = 2
Set PLine = ThisDrawing.ModelSpace. AddLightWeightPolyline(points)
PLine.Closed = True
ThisDrawing.Application.ZoomAll
End Sub
Private Sub PLine_Modified (ByVal pObject As AutoCAD.IAcadObject)
' This event is triggered when the polyline is resized.
' If the polyline is deleted the modified event is still
' triggered, so we use the error handler to avoid
' reading data from a deleted object.
On Error GoTo ERRORHANDLER
MsgBox "The area of " & pObject.ObjectName & " is: " _
& pObject.Area
Exit Sub
ERRORHANDLER:
MsgBox Err.Description

```

89

End Sub

## **II.**

### **Sử dụng Form**

#### **1.**

#### **Làm việc với Form và Macro**

**Tạo mới một form, chạy form trong chế độ run\_mode**

Giống hệ VB

#### **Chèn controls vào form**

Không khác VB

#### **Ẩn, hiển thị form**

Giống hệ VB

#### **Load và Unload form**

Giống y như VB

## **Sử dụng Modal form**

Đây là điểm khác của VBA với VB. Chế độ Modal form cho phép bạn cùng một lúc làm việc với cả Form và cả màn hình của CAD như việc Pick chuột, đánh lệnh,...

2.

## **Làm việc với module và macro**

### **Chạy macro từ toolbar và menu**

Bạn có thể chạy macro từ AutoCAD toolbar hoặc menu bằng cách thay đổi menu Macro property cho toolbar hoặc menu đó. Macro property có dạng:

**VBARUN filename.dvb!modulename.macroname**

Trong đó:

*filename* là tên của project file.

*modulename* là tên của module chứa macro cần chạy.

*macroname* là tên của macro.

Ví dụ: VBARUN Project1.dvb!module1.thongkethep

### **Tải Project một cách tự động**

Có 2 cách để Load Project một cách tự động:

1.

Khi VBA được tải, nó sẽ tìm kiếm trong AutoCAD một project có tên là *acad.dvb*. File này sẽ được tải một cách mặc định như một default project

2.

Cũng giống như VBA, Autolisp cũng có một file có tên là *acad.lsp* cũng được tải một cách tự động khi bản vẽ mới xuất hiện. Dòng lệnh sau trong file *acad.lsp* dùng để tải project *myproj.dvb* vào trong bản vẽ mỗi khi một bản vẽ mới được mở

```
(defun S::STARTUP()
```

```
(command "_VBALOAD" "myproj.dvb")
```

```
)
```

### **Chạy macro một cách tự động**

Có 2 cách để chạy macro một cách tự động:

1.

Bạn có thể chạy các macro một cách tự động, bằng cách soạn thảo AutoCAD startup của acad.lsp. Ví dụ, để tự động chạy Macro có tên drawline, đầu tiên bạn copy đoạn macro vào trong file project acad.dvb . Sau đó, mở notepad.exe và soạn đoạn sau:

```
(defun S::STARTUP()  
(command "_vbarun" "drawline")
```

90

)

2.

Trong Project *acad.dvb*, macro có tên AcadStartup sẽ tự động được chạy khi VBA load.

### **III.**

#### **Tương tác với các ứng dụng và các cơ sở dữ liệu khác**

1.

##### **Tương tác với Visual Lisp**

Tự đọc

2.

##### **Sử dụng cơ sở dữ liệu DAO**

Tự đọc, giống VB

3.

##### **Giao tiếp với các ứng dụng khác.**

Để trao đổi thông tin giữa các Activex Model, ta làm theo các bước sau:

1.

Tham chiếu đến ActiveX Object Model của ứng dụng cần giao tiếp với.

2.

Tạo một instance của ứng dụng

3.

Viết Code bằng cách sử dụng cả AutoCAD Object Model và Object Model của ứng dụng ngoài.

##### **Tham chiếu đến ActiveX Object Library của ứng dụng khác**

Việc đầu tiên khi bạn trao đổi thông tin với các ứng dụng ngoài là tham chiếu đến ActiveX Object Library, vì nó chứa tất cả các đối tượng, phương thức, thuộc tính, hằng số và sự kiện do ứng dụng đó định nghĩa

Thông thường, để kết nối với các Activex Library, bạn chọn Tools menu

- References. , sau đó một danh sách các object libraries VBA tìm thấy trong hệ thống của bạn sẽ được hiện lên. Sau đó bạn chọn những Library mà bạn cần trao đổi dữ liệu tới nó. Ví dụ, bạn thêm object library của Microsoft Excel, chọn *Microsoft Excel object library entry* trong danh sách.

Khi bạn đã tạo tham chiếu đến object library của ứng dụng ngoài, bạn có thể sử dụng VBA Object Browser để xem danh sách các object của ứng dụng mà bạn tham chiếu đến.

Cụ thể như sau:

1.

In the VBA IDE, open the Tools menu and select the References menu option.

2.

Find and select the entry in the list of Available References for the application you want to access.

3.

Select OK to close the dialog box with your changes.

### **Tạo một Instance của ứng dụng ngoài**

91

Sau khi tham chiếu đến Object library, bạn phải tạo một instance của ứng dụng đó. Các bước thực hiện như sau:

1.

Bạn khai báo một biến đại diện cho application. Ví dụ, khai báo một biến đại diện cho Excel.Application:

**Dim ExcelAppObj as Excel.Application**

2.

sau đó bạn tạo một instance cho application. Ví dụ:

**Set ExcelAppObj = New Excel.Application**

**Viết Code cho Objects từ ứng dụng ngoài**

Ví dụ sau đây điều khiển sự hiển thị của Excel sau khi đã tạo Instance cho nó:

**ExcelAppObj.Visible = TRUE**

Bạn có thể sử dụng VBA Object Browser để tìm các đối tượng và sử dụng help file để đọc thêm về bất kỳ một Object Model mà bạn tham chiếu đến.

### **Thoát khỏi ứng dụng**

Khi ứng dụng ngoài chạy, nó chiếm một bộ nhớ nhất định trong máy tính. Sau khi không sử dụng đến chương trình đó nữa, tốt nhất bạn nên tắt tất cả các ứng dụng mà bạn đã tạo Instance cho nó. Ví dụ, thoát session Excel như sau:

**ExcelAppObj.Application.Quit**

### **List AutoCAD attributes on an Excel spreadsheet**

Ví dụ: thủ tục sau tìm tất cả các block references trong bản vẽ hiện hành. Liệt kê tất cả các attributes của các block references đó:

```
Sub Ch12_Extract()  
Dim Excel As Excel.Application  
Dim ExcelSheet As Object  
Dim ExcelWorkbook As Object  
Dim RowNum As Integer  
Dim Header As Boolean  
Dim elem As AcadEntity  
Dim Array1 As Variant  
Dim Count As Integer  
' Launch Excel.  
Set Excel = New Excel.Application  
' Create a new workbook and find the active sheet.  
Set ExcelWorkbook = Excel.Workbooks.Add  
Set ExcelSheet = Excel.ActiveSheet  
ExcelWorkbook.SaveAs "Attribute.xls"  
RowNum = 1  
Header = False  
' Iterate through model space finding  
' all block references.  
For Each elem In ThisDrawing.ModelSpace  
With elem
```

```
' When a block reference has been found,  
' check it for attributes  
If StrComp(.EntityName, "AcDbBlockReference", 1) = 0 Then  
If .HasAttributes Then  
' Get the attributes  
Array1 = .GetAttributes
```

92

```
' Copy the Tagstrings for the  
' Attributes into Excel  
For Count = LBound(Array1) To UBound(Array1)  
If Header = False Then  
If StrComp(Array1(Count).EntityName, _  
"AcDbAttribute", 1) = 0 Then  
ExcelSheet.Cells(RowNum, Count + 1).value = _  
Array1(Count).TagString  
End If  
End If  
Next Count  
RowNum = RowNum + 1  
For Count = LBound(Array1) To UBound(Array1)  
ExcelSheet.Cells(RowNum, Count + 1).value = Array1(Count).textString  
Next Count  
Header = True  
End If  
End If  
End With  
Next elem  
Excel.Application.Quit  
End Sub
```



## **Sử dụng GetObject, và NewObject**

Như đã trình bày ở trường I “Cách lập trình AutoCAD VBA trong Excel”,  
Bạn cũng thể GetObject, và NewObject để connect tới các ứng dụng khác

Bảng lớp ứng dụng

### **APPLICATION CLASS**

#### **IDENTIFICATION**

Excel

Excel.Application.x (x là product version)

Word

Word.Application.x (x là version index)

Microsoft Product Versions

#### **PRODUCT**

#### **VERSION**

#### **EXAMPLE**

Office 95

7

Word.Application.7

Office 97

8

Excel.Application.8

Office 2000

9

Word.Application.9

Office XP

10

Powerpoint.Application.10

Office 2003

11

Outlook.Application.11

Ví dụ:

```
Public Sub StartExcel(App As Excel.Application, Visible As Boolean)
```

```
'handle errors inline
```

```
On Error Resume Next
```

```
Set App = GetObject( "Excel.Application") 'depends on application
'check to see if application is running
If Err Then
'no, application will need to be started
```

93

```
Err.Clear
Set App = CreateObject("Excel.Application") 'depends on application
'check to see if application was started
If Err Then
'no, application could not be started exit
Exit Sub
End If
End If
'set the application visibility
App.Visible = Visible
End Sub
Kiểm tra xem đã connect được với Excel hay chưa:
Public Sub Start()
Dim oExcel As Excel.Application
'attempt to start Excel
StartExcel oExcel, True
If Not oExcel Is Nothing Then
MsgBox "Success"
Else
MsgBox "Could not start Excel, exiting ...", vbCritical
Exit Sub
End If
End Sub
```

**IV.**

## **Làm việc với Xdata**

### **1.**

#### **Khái niệm về XData**

Xdata là dữ liệu đính kèm với các thực thể. Xdata có thể có hay không có. Người dùng thường dùng nó để ghi thêm các thông tin mở rộng về thực thể. Ví dụ, một số hiệu thép ghi trong bản vẽ bê tông thường có số hiệu thép, phi, khoảng cách giữa các thanh. Nhưng không có độ dài thanh đó. Ta có thể dùng Xdata để ghi thêm độ dài thép.

Extended data giới hạn 16K cho mỗi thực thể entity. Extended data gồm các group code DXF từ 1000 đến 1071. AutoCAD chứa các group và dữ liệu trong nó nhưng không sử dụng chúng. Bạn có thể tham khảo Group code và kiểu dữ liệu của từng Group code trong bảng DXF Code. Sau đây là một số Group code điển hình:

#### **String**

Group Code thứ 1000. Strings trong extended data có độ dài tối đa là 255 bytes (với byte thứ 256 dành cho null character).

#### **Application Name**

Group Code thứ 1001 (cũng là giá trị kiểu string). Application names có độ dài tối đa là 31 bytes (byte thứ 32 dành cho null character) và phải tuân theo quy tắc đánh tên (như tên của layer). Một application name có thể chứa các chữ cái như: số, và các ký tự đặc biệt như "\$", "'", và "\_". Nhưng không được phép chứa khoảng trống.

#### **Layer Name**

Group code thứ 1003. Tên của một layer gắn liền với Xdata.

#### **Database Handle**

Group code thứ 1005. Handle của entity trong drawing database.

#### **3D Point**

94

Group code thứ 1010. 3 giá trị kiểu Real, là tọa độ một điểm.

#### **Real**

Group code thứ 1040. Một số thực.

#### **Integer**

Group code thứ 1070. Một số 16bit integer.

### **Long**

Group code thứ 1071. Một số 32bit long. Nếu giá trị này được gán cho một số short integer hoặc real, thì nó sẽ được quy đổi về số kiểu long integer. Nếu giá trị không hợp lệ(ví dụ như kiểu string), nó được quy đổi về số long zero (0L).

### **Control String**

Group code thứ 1002. Xdata control string can be either "{" or "}". These braces enable the application to organize its data by subdividing it into lists. The left brace begins a list, and the right brace terminates the most recent list. Lists can be nested.

Note If a 1001 group appears within a list, it is treated as a string and does not begin a new application group.

### **Binary Data**

1004. Binary data that is organized into variablelength chunks, which can be handled in ObjectARX with the ads\_binary structure. The maximum length of each chunk is 127 bytes.

Note AutoLISP cannot directly handle binary chunks, so the same precautions that apply to long (1071) groups apply to binary groups as well.

### **World Space Position**

1011. Unlike a simple 3D point, the WCS coordinates are moved, scaled, rotated, and mirrored along with the parent entity to which the extended data belongs. The WCS position is also stretched when the STRETCH command is applied to the parent entity and when this point lies within the select window.

### **World Space Displacement**

1012. A 3D point that is scaled, rotated, or mirrored along with the parent, but not stretched or moved.

### **World Direction**

1013. A 3D point that is rotated or mirrored along with the parent, but not scaled, stretched, or moved. The WCS direction is a normalized displacement that always has a unit length.

### **Distance**

1041. A real value that is scaled along with the parent entity.

### **Scale Factor**

1042. Also a real value that is scaled along with the parent.

2.

**Sets the extended data (XData) associated with an object.**

Cú pháp như sau

**object.SetXData XDataType, XData**

Trong đó:

Object: Tất cả các thực thể bản vẽ , AttributeReference, Block, Dictionary, DimStyle, Group, Layer, Linetype, PlotConfigurations, RegisteredApplication, TextStyle, UCS, View, Viewport; Xrecord.

95

XdataType: Variant (array of short) inputonly

Xdata: Array of Variant; inputonly

**3.**

**Gets the extended data (XData) associated with an object.**

Cú pháp như sau

**object.GetXData AppName, XDataType, XDataValue**

Object: Tất cả các thực thể bản vẽ , AttributeReference, Block, Dictionary, DimStyle, Group, Layer, Linetype, PlotConfigurations, RegisteredApplication, TextStyle, UCS, View, Viewport; Xrecord.

AppName: String; inputonly, chuỗi rỗng sẽ trả về tất cả các dữ liệu được gán với đối tượng.

XdataType: Variant (array of short) Outputonly

Xdata: Array of Variant; Output only

**4.**

**Các Ví dụ**

**Ví dụ 1:**

```

Sub Ch10_AttachXDataToSelectionSetObjects()
' Create the selection set
Dim sset As Object
Set sset = ThisDrawing.SelectionSets.Add("SS1")
' Prompt the user to select objects
sset.SelectOnScreen
' Define the xdata
Dim appName As String, xdataStr As String
appName = "MY_APP"
xdataStr = "This is some xdata"
Dim xdataType(0 To 1) As Integer
Dim xdata(0 To 1) As Variant
' Define the values for each array 1001 indicates the appName
xdataType(0) = 1001
xdata(0) = appName
'1000 indicates a string value
xdataType(1) = 1000
xdata(1) = xdataStr
' Loop through all entities in the selection set and assign the xdata to each
entity
Dim ent As Object
For Each ent In sset
ent.SetXData xdataType, xdata
Next ent
End Sub

```

### **Ví dụ 2:**

```

Sub Example_SetXdata()
' Ví dụ sau tạo ra một đường line và gán extended data cho nó.
' Create the line
Dim lineObj As AcadLine
Dim startPt(0 To 2) As Double, endPt(0 To 2) As Double
startPt(0) = 1#: startPt(1) = 1#: startPt(2) = 0#

```

96

```
endPt(0) = 5#: endPt(1) = 5#: endPt(2) = 0#
```

```
Set lineObj = ThisDrawing.ModelSpace.AddLine(startPt, endPt)
```

```
ZoomAll
```

```
' Initialize all the xdata values. Note that first data in the list should be
```

```
' application name and first datatype code should be 1001
```

```
Dim DataType(0 To 9) As Integer
```

```
Dim Data(0 To 9) As Variant
```

```
Dim reals3(0 To 2) As Double
```

```
Dim worldPos(0 To 2) As Double
```

```
DataType(0) = 1001: Data(0) = "Test_Application"
```

```
DataType(1) = 1000: Data(1) = "This is a test for xdata"
```

```
DataType(2) = 1003: Data(2) = "0" ' layer
```

```
DataType(3) = 1040: Data(3) = 1.23479137438413E+40 ' real
```

```
DataType(4) = 1041: Data(4) = 1237324938 ' distance
```

```
DataType(5) = 1070: Data(5) = 32767 ' 16 bit Integer
```

```
DataType(6) = 1071: Data(6) = 32767 ' 32 bit Integer
```

```
DataType(7) = 1042: Data(7) = 10 ' scaleFactor
```

```
reals3(0) = 2.95: reals3(1) = 100: reals3(2) = 20
```

```
DataType(8) = 1010: Data(8) = reals3 ' real
```

```
worldPos(0) = 4: worldPos(1) = 400.99999999: worldPos(2) = 2.798989
```

```
DataType(9) = 1011: Data(9) = worldPos ' world space position
```

```
' Attach the xdata to the line
```

```
lineObj.SetXData DataType, Data
```

```
' Return the xdata for the line
```

```
Dim xdataOut As Variant
```

```
Dim xtypeOut As Variant
```

```
lineObj.GetXData "", xtypeOut, xdataOut
```

```
End Sub
```

**Ví dụ 3:**

Ví dụ sau hiển thị tất cả các **xdata** trong selection set

```
Sub Ch10_ViewXData()
```

```
' Find the selection created in previous example
```

```
Dim sset As Object
```

```
Set sset = ThisDrawing.SelectionSets.Item("SS1")
```

```
' Define the xdata variables to hold xdata information
```

```
Dim xdataType As Variant
```

```
Dim xdata As Variant
```

```
Dim xd As Variant
```

```
'Define index counter
```

```
Dim xdi As Integer
```

```
xdi = 0
```

```
' Loop through the objects in the selection set and retrieve the xdata for  
the object
```

```
Dim msgstr As String
```

```
Dim appName As String
```

```
Dim ent As AcadEntity
```

```
appName = "MY_APP"
```

97

```
For Each ent In sset
```

```
msgstr = ""
```

```
xdi = 0
```

```
' Retrieve the appName xdata type and value
```

```
ent.GetXData appName, xdataType, xdata
```

```
' If the xdataType variable is not initialized, there
```

```
' was no appName xdata to retrieve for that entity
```

```
If VarType(xdataType) <> vbEmpty Then
```

```
For Each xd In xdata
```

```
msgstr = msgstr & vbCrLf & xdataType(xdi) _
```



```

& ":" & xd
xdi = xdi + 1
Next xd
End If
' If the msgstr variable is NULL, there was no xdata
If msgstr = "" Then msgstr = vbCrLf & "NONE"
MsgBox appName & " xdata on " & ent.ObjectName &
":" & vbCrLf & msgstr
Next ent
End Sub

```

#### **Ví dụ 4:**

Ví dụ sau, lọc tất cả các vòng tròn chứa xdata của “MY\_APP” application:

```

Sub Ch4_FilterXdata()
Dim sstext As AcadSelectionSet
Dim mode As Integer
Dim pointsArray(0 To 11) As Double
mode = acSelectionSetWindowPolygon
pointsArray(0) = 12#: pointsArray(1) = 7#: pointsArray(2) = 0
pointsArray(3) = 12#: pointsArray(4) = 10#: pointsArray(5) = 0
pointsArray(6) = 10#: pointsArray(7) = 10#: pointsArray(8) = 0
pointsArray(9) = 10#: pointsArray(10) = 7#: pointsArray(11) = 0
Dim FilterType(1) As Integer
Dim FilterData(1) As Variant
Set sstext = ThisDrawing.SelectionSets.Add("SS9")
FilterType(0) = 0
FilterData(0) = "Circle"
FilterType(1) = 1001
FilterData(1) = "MY_APP"
sstext.SelectByPolygon mode, pointsArray, FilterType, FilterData

```

**V.**

#### **Làm việc với Xrecord**

**1.**

## Khái niệm về Xrecord

Xrecord là đối tượng thuộc Collection Dictionaries. Khái niệm về Xrecord cũng tương tự như Xdata, nhưng Xrecord không bị giới hạn về kích thước cũng như thứ tự và nó có thể chứa bất kỳ loại dữ liệu nào.

Khác với Xdata, XRecords chứa dữ liệu trong standard AutoCAD group codes, tức là không phải chứa trong phần mở rộng như Xdata. Các group code này nhỏ hơn 1000. Tất cả các standard

98

AutoCAD group codes đều được sử dụng để chứa dữ liệu. Một điều quan trọng nữa là ta có thể quản lý Xrecord bằng object IDs.

Ví dụ: Các bước AutoCAD ghi lại các thông tin của một Layer như sau:

- 

Tạo ra một từ điển mở rộng (Extension dictionary) trong layer collection.

- 

Tạo một Dictionary object lấy tên là ACAD\_LAYERSTATE trong extension dictionary.

- 

Lưu tất cả các thuộc tính của layer vào một XRecord object trong ACAD\_LAYERSTATE dictionary. AutoCAD lưu tất cả các thiết lập của các layer vào Xrecord nếu bạn chọn chức năng Save. Khi bạn khôi phục lại (restore the layer setting, AutoCAD sẽ lấy dữ liệu từ XRecord để khôi phục lại).

Như vậy, ta cũng có thể ghi lại các dữ liệu cần thiết trong quá trình lập trình xử lý bản vẽ bằng cách sử dụng Xrecord.

Những Group code sau có thể được sử dụng trong XRecord objects:

- 

100: Subclass marker (AcDbXrecord)

- 

1369: (except 5 and 105)

XRecord objects được ghi lại cùng với bản vẽ, và chúng có thể được truy cập một cách trực tiếp bởi ObjectARX và LISP. Do vậy, nếu bạn cần bảo mật dữ liệu, bạn phải mã hóa chúng.:

**VBA class name:**

AcadXRecord

**Create using:**

Dictionary.AddXRecord

**Access via:**

Dictionary.Item

2.

**Phương thức AddXRecord**

**RetVal = object.AddXRecord(Keyword)**

▪

Object: kiểu Dictionary, đối tượng sẽ chứa XRecord.

▪

Keyword: String, inputonly. Là tên của XRecord trong dictionary.

▪

RetVal: XRecord object

Sub Example\_AddXRecord()

' This example creates a new XRecord if one doesn't exist,

' appends data to the XRecord, and then reads it back. To see data being added,

' run the example more than once.

Dim TrackingDictionary As AcadDictionary, TrackingXRecord As AcadXRecord

Dim XRecordDataType As Variant, XRecordData As Variant

Dim ArraySize As Long, iCount As Long

Dim DataType As Integer, Data As String, msg As String

' Unique identifiers to distinguish this XRecordData from other XRecordData

Const TYPE\_STRING = 1

Const TAG\_DICTIONARY\_NAME = "ObjectTrackerDictionary"

99

Const TAG\_XRECORD\_NAME = "ObjectTrackerXRecord"

```

' Connect to the dictionary in which to store the XRecord
On Error GoTo CREATE
Set TrackingDictionary =
ThisDrawing.Dictionaries(TAG_DICTIONARY_NAME)
Set TrackingXRecord =
TrackingDictionary.GetObject(TAG_XRECORD_NAME)
On Error GoTo 0
' Get current XRecordData
TrackingXRecord.GetXRecordData XRecordDataType, XRecordData
' If there is no array yet then create one
If VarType(XRecordDataType) And vbArray = vbArray Then
ArraySize = UBound(XRecordDataType) + 1
' Get the size of the data elements returned
ArraySize = ArraySize + 1
' Increase to hold new data
ReDim Preserve XRecordDataType(0 To ArraySize)
ReDim Preserve XRecordData(0 To ArraySize)
Else
ArraySize = 0
ReDim XRecordDataType(0 To ArraySize) As Integer
ReDim XRecordData(0 To ArraySize) As Variant
End If
' Append new XRecord Data
' For this sample we only append the current time to the XRecord
XRecordDataType(ArraySize) = TYPE_STRING:
XRecordData(ArraySize) = _
CStr(Now)
TrackingXRecord.SetXRecordData XRecordDataType, XRecordData
' Read back all XRecordData entries
TrackingXRecord.GetXRecordData XRecordDataType, XRecordData
ArraySize = UBound(XRecordDataType)
' Retrieve and display stored XRecordData
For iCount = 0 To ArraySize
' Get information for this element

```

```

DataType = XRecordDataType(iCount)
Data = XRecordData(iCount)
If DataType = TYPE_STRING Then
msg = msg & Data & vbCrLf
End If
Next
MsgBox "The data in the XRecord is: " & vbCrLf & vbCrLf & msg,
vbInformation
Exit Sub
CREATE:
' Create the objects that hold this XRecordData
If TrackingDictionary Is Nothing Then ' Make sure to have tracking
object
Set TrackingDictionary = _
ThisDrawing.Dictionaries.Add(TAG_DICTIONARY_NAME)
Set TrackingXRecord = _

```

100

```
TrackingDictionary.AddXRecord(TAG_XRECORD_NAME)
```

```
End If
```

```
Resume
```

```
End Sub
```

**3.**

### **Phương thức SetXRecordData**

**object.SetXRecordData XRecordDataType, XRecordData**

▪

Object: kiểu XRecord Object, đối tượng sẽ chứa dữ liệu.

▪

XrecordDataType: Variant (array of short); inputonly.

The following group codes are common to all XRecord objects:

## Group codes

### Description

100

Subclass marker (AcDbXrecord)

1369 (except 5 and 105)

Giá trị có thể được sử dụng bởi bất kỳ ứng dụng nào.

4.

### Phương thức GetXRecordData

**object.GetXRecordData XRecordDataType, XRecordData Value**

▪

Object: kiểu XRecord Object, đối tượng sẽ chứa dữ liệu.

▪

XrecordDataType: Variant (array of short); Outputonly.

Sub Example\_SetXRecordData()

' This example creates a new XRecord if one doesn't exist,

' appends data to the XRecord, and reads it back. To see data being added,

' run the example more than once.

```
Dim TrackingDictionary As AcadDictionary, TrackingXRecord As AcadXRecord
```

```
Dim XRecordDataType As Variant, XRecordData As Variant
```

```
Dim ArraySize As Long, iCount As Long
```

```
Dim DataType As Integer, Data As String, msg As String
```

```
' Unique identifiers to distinguish our XRecordData from other XRecordData
```

```
Const TYPE_STRING = 1
```

```
Const TAG_DICTIONARY_NAME = "ObjectTrackerDictionary"
```

```
Const TAG_XRECORD_NAME = "ObjectTrackerXRecord"
```

```
' Connect to the dictionary in which the XRecord is stored
```

```
On Error GoTo CREATE
```

```
Set TrackingDictionary = ThisDrawing.Dictionaries(TAG_DICTIONARY_NAME)
```

```
Set TrackingXRecord = TrackingDictionary.GetObject(TAG_XRECORD_NAME)
```

```
On Error GoTo 0
```

```

' Get current XRecordData
TrackingXRecord.GetXRecordData XRecordDataType, XRecordData
' If there is no array already, create one
If VarType(XRecordDataType) And vbArray = vbArray Then
ArraySize = UBound(XRecordDataType) + 1
' Get the size of the data elements returned
ArraySize = ArraySize + 1

```

101

```

' Increase to hold new data
ReDim Preserve XRecordDataType(0 To ArraySize)
ReDim Preserve XRecordData(0 To ArraySize)
Else
ArraySize = 0
ReDim XRecordDataType(0 To ArraySize) As Integer
ReDim XRecordData(0 To ArraySize) As Variant
End If
' Append new XRecord Data
'
' For this sample, we only append the current time to the XRecord
XRecordDataType(ArraySize) = TYPE_STRING:
XRecordData(ArraySize) = _
CStr(Now)
TrackingXRecord.SetXRecordData XRecordDataType, XRecordData
' Read back all XRecordData entries
TrackingXRecord.GetXRecordData XRecordDataType, XRecordData
ArraySize = UBound(XRecordDataType)
' Retrieve and display stored XRecordData
For iCount = 0 To ArraySize
' Get information for this element

```

```
DataType = XRecordDataType(iCount)
Data = XRecordData(iCount)
If DataType = TYPE_STRING Then
msg = msg & Data & vbCrLf
End If
Next
MsgBox "The data in the XRecord is: " & vbCrLf & vbCrLf & msg,
vbInformation
Exit Sub
CREATE:
' Create the objects that hold the XRecordData
If TrackingDictionary Is Nothing Then ' Make sure the tracking object is
there
Set TrackingDictionary = _
ThisDrawing.Dictionaries.Add(TAG_DICTIONARY_NAME)
Set TrackingXRecord = _
TrackingDictionary.AddXRecord(TAG_XRECORD_NAME)
End If
Resume
End Sub
```